Assignment 8, 2023

LOO-CV model comparison

Aki Vehtari et al.

1 General information

This is for BDA 2023

The maximum amount of points from this assignment is 6.

We have prepared a **quarto template specific to this assignment (html, qmd, pdf)** to help you get started.

Setup

We recommend Aalto students use jupyter.cs.aalto.fi, for all others we also provide a docker container.

Tip

Reading instructions:

- **The reading instructions for BDA3 Chapter 6** (posterior predictive checking).
- **The reading instructions for BDA3 Chapter 7** (predictive performance).
- The 'loo' package vignette on the basics of LOO shows an example of how to modify Stan code and use the package with Stan models.
- Also read about PSIS-LOO in the PSIS-LOO paper.
- CV-FAQ includes a lot of informative answers to frequent questions and misconceptions.

Grading instructions:
The grading will be done in peergrade. All grading questions and evaluations for this assignment are contained within this document in the collapsible **Rubric** blocks.
Installing and using `CmdStanR`:
See the Stan demos on how to use Stan in R (or Python). Aalto JupyterHub has working R and CmdStanR/RStan environment and is probably the easiest way to use Stan. * To use CmdStanR in Aalto JupyterHub: `library(cmdstanr) set_cmdstan_path('/coursedata/cmdstan')`
The Aalto Ubuntu desktops also have the necessary libraries installed.
To install Stan on your laptop, run 'install.packages("cmdstanr", repos = c("https://mc-stan.org/r-packages/", getOption("repos")))' in R. If you encounter problems, see additional answers in **FAQ**. For Aalto students, if you don't succeed in short amount of time, it is probably easier to use Aalto JupyterHub.

If you use `Aalto JupyterHub`, all necessary packages have been pre-installed. In your laptop, install package `cmdstanr`. Installation instructions on Linux, Mac and Windows can be found at https://mc-stan.org/cmdstanr/. Additional useful packages are `loo`, `bayesplot` and `posterior` (but you don't need these in this assignment). For Python users, `PyStan`, `CmdStanPy`, and `ArviZ` packages are useful.
Stan manual can be found at https://mc-stan.org/users/documentation/. From this website, you can also find a lot of other useful material about Stan.
If you edit files ending `.stan` in RStudio, you can click "Check" in the editor toolbar to make syntax check. This can significantly speed-up writing a working Stan model.

! Reporting accuracy

For posterior statistics of interest, only report digits that are not completely random based on the Monte Carlo standard error (MCSE).
Example: If you estimate $E(\mu) \approx 1.234$ with $\text{MCSE}(E(\mu)) = 0.01$, then the true expectation is likely to be between 1.204 and 1.264, it makes sense to report $E(\mu) \approx 1.2$.
See Lecture video 4.1, the chapter notes, and a case study for more information.

Further information

- The recommended tool in this course is R (with the IDE RStudio).
- Instead of installing R and RStudio on you own computer, see **how to use R and RStudio remotely**.
- If you want to install R and RStudio locally, download R and RStudio.
- There are tons of tutorials, videos and introductions to R and RStudio online. You can find some initial hints from **RStudio Education pages**.
- When working with R, we recommend writing the report using `quarto` and the provided template. The template includes the formatting instructions and how to include code and figures.
- Instead of `quarto`, you can use other software to make the PDF report, but the the same instructions for formatting should be used.
- Report all results in a single, **anonymous** *.pdf -file and submit it in **peergrade.io**.
- The course has its own R package `aaltobda` with data and functionality to simplify coding. The package is pre-installed in JupyterHub. To install the package on your own system, run the following code (upgrade="never" skips question about updating other packages):

```
install.packages("aaltobda", repos = c("https://avehtari.github.io/BDA_course_Aalto/", getOption("re
```

- Many of the exercises can be checked automatically using the R package `markmyassignment` (pre-installed in JupyterHub). Information on how to install and use the package can be found in the `markmyassignment` documentation. There is no need to include `markmyassignment` results in the report.

- Recommended additional self study exercises for each chapter in BDA3 are listed in the course web page. These will help to gain deeper understanding of the topic.
- Common questions and answers regarding installation and technical problems can be found in Frequently Asked Questions (FAQ).
- Deadlines for all assignments can be found on the course web page and in Peergrade. You can set email alerts for the deadlines in Peergrade settings.
- You are allowed to discuss assignments with your friends, but it is not allowed to copy solutions directly from other students or from internet.
- You can copy, e.g., plotting code from the course demos, but really try to solve the actual assignment problems with your own code and explanations.
- Do not share your answers publicly.
- Do not copy answers from the internet or from previous years. We compare the answers to the answers from previous years and to the answers from other students this year.
- Use of AI is allowed on the course, but the most of the work needs to by the student, and you need to report whether you used AI and in which way you used them (See points 5 and 6 in Aalto guidelines for use of AI in teaching).
- All suspected plagiarism will be reported and investigated. See more about the **Aalto University Code of Academic Integrity and Handling Violations Thereof**.
- Do not submit empty PDFs, almost empty PDFs, copy of the questions, nonsense generated by yourself or AI, as these are just harming the other students as they can't do peergrading for the empty or nonsense submissions. Violations of this rule will be reported and investigated in the same way was plagiarism.
- If you have any suggestions or improvements to the course material, please post in the course chat feedback channel, create an issue, or submit a pull request to the public repository!

Rubric

- Can you open the PDF and it's not blank nor nonsense? If the pdf is blank, nonsense, or something like only a copy of the questions, 1) report it as problematic in Peergrade-interface to get another report to review, and 2) send a message to TAs.
- Is the report anonymous?

This is the template for assignment 8. You can download the qmd-file or copy the code from this rendered document after clicking on `</> Code` in the top right corner.

Please replace the instructions in this template by your own text, explaining what you are doing in each exercise.

Setup

The following loads several needed packages:

```
library(bayesplot)
```

```
This is bayesplot version 1.11.1.9000

- 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(cmdstanr)
```

```
This is cmdstanr version 0.8.1.9000

- CmdStanR documentation and vignettes: mc-stan.org/cmdstanr

- CmdStan path: /u/77/ave/unix/.cmdstan/cmdstan-2.35.0-rc3

- CmdStan version: 2.35.0
```

```
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(ggdist) # for stat_dotsinterval
library(posterior)
```

```
This is posterior version 1.6.0

Attaching package: 'posterior'

The following object is masked from 'package:bayesplot':

    rhat

The following objects are masked from 'package:stats':

    mad, sd, var

The following objects are masked from 'package:base':

    %in%, match
```

```
library(brms)
```

```
Loading required package: Rcpp
```

```
Loading 'brms' package (version 2.21.6). Useful instructions
can be found by typing help('brms'). A more detailed introduction
to the package is available through vignette('brms_overview').
```

```
Attaching package: 'brms'
```

```
The following objects are masked from 'package:ggdist':

    dstudent_t, pstudent_t, qstudent_t, rstudent_t
```

```
The following object is masked from 'package:bayesplot':

    rhat
```

```
The following object is masked from 'package:stats':

    ar
```

```
# Globally specfiy cmdstan backend for brms
options(brms.backend="cmdstanr")
# Tell brms to cache results if possible
options(brms.file_refit="on_change")

# Set more readable themes with bigger font for plotting packages
ggplot2::theme_set(theme_minimal(base_size = 14))
bayesplot::bayesplot_theme_set(theme_minimal(base_size = 14))
```

2 A hierarchical model for chicken weight time series

2.1 Exploratory data analysis

In the first part of this assignment, you will explore the dataset `ChickWeight`. In particular, you will see what information is recorded in the dataset, and how you can use visualisation to learn more about the dataset. More information can be found on the corresponding page of the R documentation.

```
head(ChickWeight, 10)
```

```
Grouped Data: weight ~ Time | Chick
   weight Time Chick Diet
1      42    0     1    1
2      51    2     1    1
3      59    4     1    1
4      64    6     1    1
5      76    8     1    1
6      93   10     1    1
```

```
7      106    12      1     1
8      125    14      1     1
9      149    16      1     1
10     171    18      1     1
```

Subtask 2.a)

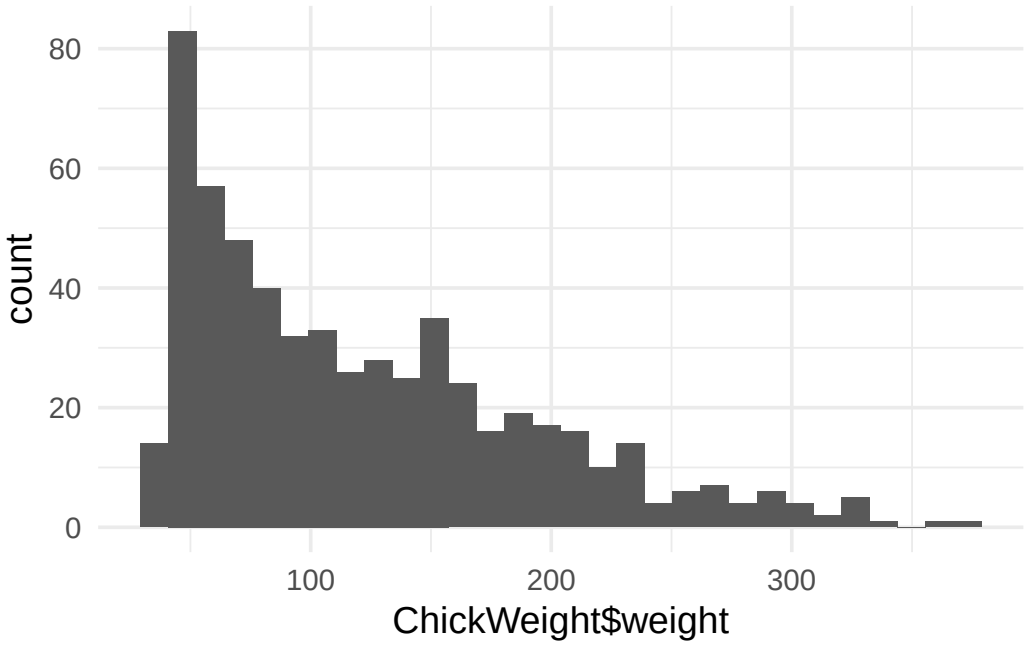
Create a histogram to explore the range of chicken weights. Describe what you see in the plot. What is the qualitative range of the data?

```
# Useful functions: ggplot, aes(x=...), geom_histogram

### {.content-hidden when-profile="public"}
ggplot(data = ChickWeight, aes(x=ChickWeight$weight)) +
  geom_histogram()
```

```
Warning: Use of `ChickWeight$weight` is discouraged.
i Use `weight` instead.

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




```
###
```

Rubric

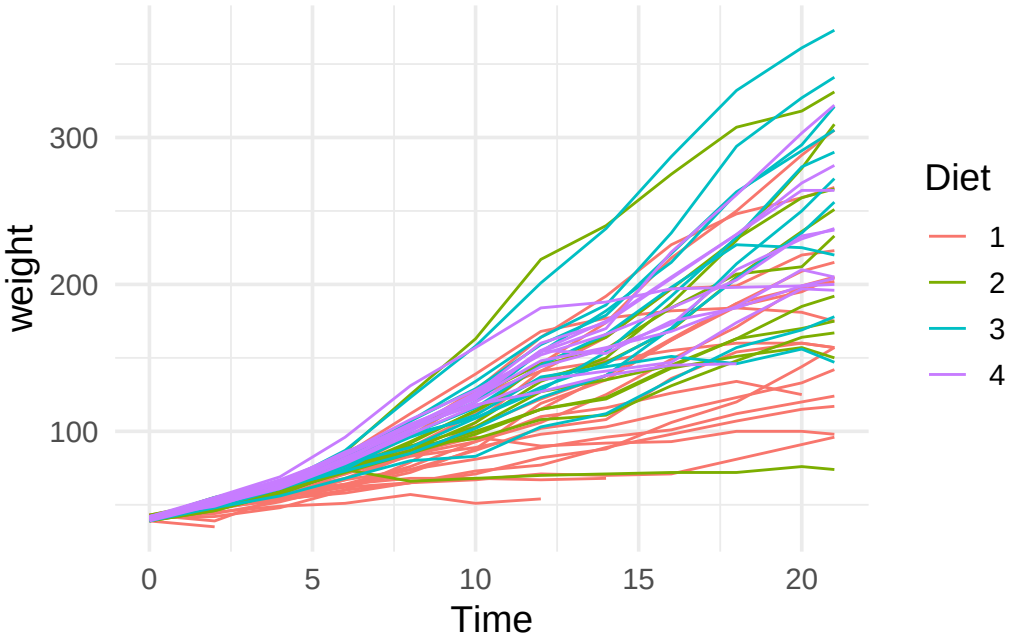
- Does the plot look correct and is it readable?
- Has it been stated that the data takes on only values which are ?

Subtask 2.b

Plot the weight of each chicken over time in a line plot. Add colours based on the diet. Describe what you see in the plot.

```
# Useful functions: ggplot, aes(x=...,y=...,group=...,color=...), geom_line

### {.content-hidden when-profile="public"}
ggplot(data = ChickWeight, aes(x=Time, y=weight, group = Chick, color=Diet)) +
  geom_line()
```




```
###
```

Rubric

- Does the plot look correct and is it readable?

2.2 Linear regression

In this section, you will build a model that predicts the weight of a chicken over time and depending on the diet. After sampling from the posteriors, you will use posterior predictive checks to see how well the predictions match the observations. Then you will adjust the model by adding more complexity, and check again.

Subtask 2.c

Using `brms`, implement a pooled linear regression with a normal model and `weight` as the predicted variable using `Diet` and `Time` as predictors. Try to use weakly informative priors.

Tip

For the prior on `Time`, consider how much the weight of a chicken (in grams) could possibly change each day. For the priors on the effects of different diets, consider how much average weight difference would be possible between diets. Note that as `Diet` is a categorical variable, the priors need to be specified for each category (apart from `Diet1` which is taken to be the baseline).

In `brms`, a regression can be specified as below, see also below (#m) or the last template. Fill in the appropriate variables, data, and likelihood family. Specify the priors, then run the model (by removing `#| eval: false` below).

```
priors <- c(
  prior(normal(0, <value>), coef = "Time"),
  prior(normal(0, <value>), coef = "Diet2"),
  prior(normal(0, <value>), coef = "Diet3"),
  prior(normal(0, <value>), coef = "Diet4")
)

f1 <- brms::brm(
  # This specifies the formula
  <OUTCOME> ~ 1 + <PREDICTOR> + <PREDICTOR>,
  # This specifies the dataset
  data = <data>,
  # This specifies the observation model family
  family = <observation_family>,
  # This passes the priors specified above to brms
  prior = priors,
  # This causes brms to cache the results
  file = "additional_files/assignment8/f1"
)

### {.content-hidden when-profile="public"}
priors <- c(
  prior(normal(0, 10), coef = "Time"),
  prior(normal(0, 50), coef = "Diet2"),
  prior(normal(0, 50), coef = "Diet3"),
  prior(normal(0, 50), coef = "Diet4")
)

f1 <- brms::brm(
  weight ~ 1 + Diet + Time,
  data = ChickWeight,
  family = "gaussian",
  prior = priors
)
###
```

Rubric

- Is the brms-formula ?
- Is the family ?
- Are the prior standard deviations reasonable, e.g. around ?

Next, you can use the `bayesplot` package to check the posterior predictions in relation to the observed data using the `pp_check` function. The function plots the y values, which are the observed data, and the y_{rep} values, which are replicated data sets from the posterior predictive distribution.

Subtask 2.d

Perform the posterior predictive check with the default arguments. What do you observe? Based on the plot, do the posterior predictions encapsulate the main features of the observed data? Point out any major differences between the predictions and the observed data. Answer the following questions:

- Are there qualitative differences between the observed data and the predicted data?
- Do the observed data seem quantitatively similar?

```
# Useful functions: brms::pp_check

### {.content-hidden when-profile="public"}
brms::pp_check(f1, plotfun="hist")
```

```
Error in eval(expr, envir, enclos): object 'f1' not found
```

```
###
```

Rubric

- Does the plot look correct and is it readable?
- Has it been recognized that the predicted data include ?
- Has it been recognized that the observed and predicted data ?

The default density plot is not always informative, but `bayesplot` has different settings that can be used to create plots more appropriate for specific data.

Subtask 2.e

Create another plot with grouping to the PPC plot using the arguments `type = "intervals_grouped"` and `group = "Diet"`. What do you observe? Point out any major differences between the predictions and the observed data. Based on your visualisations, how could the model be improved?

```
# Useful functions: brms::pp_check(..., type = ..., group=...)

### {.content-hidden when-profile="public"}
brms::pp_check(..., type = "intervals_grouped", group="Diet")
```

```
Error in eval(expr, envir, enclos): '...' used in an incorrect context
```

```
###
```

Rubric

- Does the plot look correct and is it readable?
- Is there at least one reasonable way to improve the model, e.g. ?

2.3 Log-normal linear regression

Based on the identified issues from the posterior predictive check, the model can be improved. It is advisable to change only one or a few things about a model at once. At this stage, focus on changing the observation model family to better account for the observed data.

One option is to use the lognormal observation model, which only allows positive values. In `brms` you can change the observation model family to this by setting the argument `family = "lognormal"`. Note that when using the log-normal observation model, the regression coefficients represent the change in the log weight of a chicken. The priors have been adjusted accordingly in the template.

> Subtask 2.f
>
> Adjust the model, sample from the posterior and create the same two posterior predictive check plots. Comment on your observations. Does the new model better capture some aspects of the data?

> Tip

```
log_priors <- c(
  prior(normal(0, log(3)), coef = "Time"),
  prior(normal(0, log(5)), coef = "Diet2"),
  prior(normal(0, log(5)), coef = "Diet3"),
  prior(normal(0, log(5)), coef = "Diet4")
)

### {.content-hidden when-profile="public"}
f2 <- brm(
  weight ~ Diet + Time,
  data = ChickWeight,
  family = "lognormal",
  prior = priors
)
```

```
Error in eval(expr, envir, enclos): object 'priors' not found
```

```
# criticism with ppc density (not grouped)
brms::pp_check(f2)
```

```
Error in eval(expr, envir, enclos): object 'f2' not found
```

```
# information for each chicken
brms::pp_check(f2, type = "intervals_grouped", group = "Diet")
```

```
Error in eval(expr, envir, enclos): object 'f2' not found
```

```
###
```

Rubric

- Do the plots look correct and are they readable?
- Has it been recognized that the fit to data is ?

2.4 Hierarchical log-normal linear regression

The model can further be improved by directly considering potential differences in growth rate for individual chicken. Some chickens may innately grow faster than others, and this difference can be included by including both population and group level effects in to the model.

To include a group effect in `brms`, the code `+ (predictor|group)` can be added to the model formula. In this case, the predictor is `Time` and the group is `Chick`.

Subtask 2.g

Create the same two plots as for the previous models. Comment on what you see. Do the predictions seem to better capture the observed data? Are there remaining discrepancies between the predictions and observed data that could be addressed?

2.4.1

```
priors <- c(
  prior(normal(0, log(3)), coef = "Time"),
  prior(normal(0, log(5)), coef = "Diet2"),
  prior(normal(0, log(5)), coef = "Diet3"),
  prior(normal(0, log(5)), coef = "Diet4")
)

f3 <-  brm(
  weight ~ Diet + Time + (Time|Chick),
  data = ChickWeight,
  family = "lognormal",
  prior = priors
)
```

```
Start sampling

Running MCMC with 4 sequential chains...

Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1 Iteration:  100 / 2000 [  5%]  (Warmup)
Chain 1 Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1 Iteration:  300 / 2000 [ 15%]  (Warmup)
Chain 1 Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1 Iteration:  500 / 2000 [ 25%]  (Warmup)
Chain 1 Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1 Iteration:  700 / 2000 [ 35%]  (Warmup)
Chain 1 Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1 Iteration:  900 / 2000 [ 45%]  (Warmup)
Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
```

```
Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1 Iteration: 1100 / 2000 [ 55%]  (Sampling)
Chain 1 Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 1 Iteration: 1300 / 2000 [ 65%]  (Sampling)
Chain 1 Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 1 Iteration: 1500 / 2000 [ 75%]  (Sampling)
Chain 1 Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 1 Iteration: 1700 / 2000 [ 85%]  (Sampling)
Chain 1 Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 1 Iteration: 1900 / 2000 [ 95%]  (Sampling)
Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1 finished in 9.1 seconds.
Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2 Iteration:  100 / 2000 [  5%]  (Warmup)
Chain 2 Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 2 Iteration:  300 / 2000 [ 15%]  (Warmup)
Chain 2 Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 2 Iteration:  500 / 2000 [ 25%]  (Warmup)
Chain 2 Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 2 Iteration:  700 / 2000 [ 35%]  (Warmup)
Chain 2 Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 2 Iteration:  900 / 2000 [ 45%]  (Warmup)
Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2 Iteration: 1100 / 2000 [ 55%]  (Sampling)
Chain 2 Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 2 Iteration: 1300 / 2000 [ 65%]  (Sampling)
Chain 2 Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 2 Iteration: 1500 / 2000 [ 75%]  (Sampling)
Chain 2 Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 2 Iteration: 1700 / 2000 [ 85%]  (Sampling)
Chain 2 Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 2 Iteration: 1900 / 2000 [ 95%]  (Sampling)
Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2 finished in 8.3 seconds.
Chain 3 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 3 Iteration:  100 / 2000 [  5%]  (Warmup)
Chain 3 Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 3 Iteration:  300 / 2000 [ 15%]  (Warmup)
Chain 3 Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 3 Iteration:  500 / 2000 [ 25%]  (Warmup)
Chain 3 Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 3 Iteration:  700 / 2000 [ 35%]  (Warmup)
Chain 3 Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 3 Iteration:  900 / 2000 [ 45%]  (Warmup)
Chain 3 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 3 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 3 Iteration: 1100 / 2000 [ 55%]  (Sampling)
Chain 3 Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 3 Iteration: 1300 / 2000 [ 65%]  (Sampling)
Chain 3 Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 3 Iteration: 1500 / 2000 [ 75%]  (Sampling)
Chain 3 Iteration: 1600 / 2000 [ 80%]  (Sampling)
```

```
Chain 3 Iteration: 1700 / 2000 [ 85%]  (Sampling)
Chain 3 Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 3 Iteration: 1900 / 2000 [ 95%]  (Sampling)
Chain 3 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 3 finished in 11.3 seconds.
Chain 4 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 4 Iteration:  100 / 2000 [  5%]  (Warmup)
Chain 4 Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 4 Iteration:  300 / 2000 [ 15%]  (Warmup)
Chain 4 Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 4 Iteration:  500 / 2000 [ 25%]  (Warmup)
Chain 4 Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 4 Iteration:  700 / 2000 [ 35%]  (Warmup)
Chain 4 Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 4 Iteration:  900 / 2000 [ 45%]  (Warmup)
Chain 4 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 4 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 4 Iteration: 1100 / 2000 [ 55%]  (Sampling)
Chain 4 Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 4 Iteration: 1300 / 2000 [ 65%]  (Sampling)
Chain 4 Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 4 Iteration: 1500 / 2000 [ 75%]  (Sampling)
Chain 4 Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 4 Iteration: 1700 / 2000 [ 85%]  (Sampling)
Chain 4 Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 4 Iteration: 1900 / 2000 [ 95%]  (Sampling)
Chain 4 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 4 finished in 7.8 seconds.

All 4 chains finished successfully.
Mean chain execution time: 9.2 seconds.
Total execution time: 37.0 seconds.

Loading required package: rstan

Loading required package: StanHeaders

rstan version 2.35.0.9000 (Stan version 2.35.0)

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 within-chain threading using `reduce_sum()` or `map_rect()` Stan functions,
change `threads_per_chain` option:
rstan_options(threads_per_chain = 1)

Attaching package: 'rstan'

The following objects are masked from 'package:posterior':

    ess_bulk, ess_tail
```

2.4.2

Rubric

- Do the plots look correct and are they readable?
- Has it been recognized that the fit to data is ?

Subtask 2.h

Have you encountered any convergence issues in the above models? Report and comment.

Rubric

- Has there been a potentially brief discussion of the standard convergence criteria (Rhat, ESS, divergent transitions) for all models?

2.5 Model comparison using the ELPD

There are many ways of comparing models[1]. Commonly, we evaluate point predictions, such as the mean of the predictive distribution[2], or accuracy of the whole posterior predictive. Whether we prioritise point or density predictive accuracy may serve different purposes and lead to different outcomes for model choice [3]. It is common, however, to report predictive accuracy via log-scores and point-predictive accuracy via root-mean-squared-error based on the empirical average of the predictive distribution. To cross-validate both metrics on left out observations without need to sample from each leave-one-out posterior, we use Pareto-smoothed importance sampling as discussed in the course materials (see Lecture 8).

We start comparing models based on the log-score. Use `loo::loo()` and `loo::loo_compare()` to quantify the differences in predictive performance.

Subtask 2.i

Answer the following questions using `loo`/`loo_compare`:

- Which model has the best predictive performance?
- Does the uncertainty influence the decision of which model is best?

```
# Useful functions: loo, loo_compare

### {.content-hidden when-profile="public"}
loo_f1 <- loo(f1)
```

```
Error in h(simpleError(msg, call)): error in evaluating the argument 'x' in selecting a method for fun
```

[1] In principle, when comparing models based on accuracy in predictions or parameter estimation (if true parameter values are available to you, as e.g. in simulation studies), we want to use so called strictly proper scoring rules that will always indicate when a "better" model is better and the score reaches its uniquely defined best value at the "true" model, if it is also well defined. See **Gneiting and Raftery, (2007)** for an in depth treatment of this topic.

[2] NOT predictions based on the mean of the posterior parameters, but first generating the predictive distribution and then computing an average.

[3] For instance, a unimodal and bimodal predictive density may have the same expected value, but very different areas of high posterior density and therefore very different log-scores.

```
loo_f1
```

```
Error in eval(expr, envir, enclos): object 'loo_f1' not found
```

```
loo_f2 <- loo(f2)
```

```
Error in h(simpleError(msg, call)): error in evaluating the argument 'x' in selecting a method for fun
```

```
loo_f2
```

```
Error in eval(expr, envir, enclos): object 'loo_f2' not found
```

```
loo_f3 <- loo(f3)
```

```
Warning: Found 3 observations with a pareto_k > 0.7 in model 'f3'. We recommend
to set 'moment_match = TRUE' in order to perform moment matching for
problematic observations.
```

```
loo_f3
```

```
Computed from 4000 by 578 log-likelihood matrix.

         Estimate   SE
elpd_loo  -2254.2 27.0
p_loo        78.4  6.6
looic      4508.5 54.1
------
MCSE of elpd_loo is NA.
MCSE and ESS estimates assume MCMC draws (r_eff in [0.3, 1.5]).

Pareto k diagnostic values:
                         Count Pct.    Min. ESS
(-Inf, 0.7]   (good)     575   99.5%   150
   (0.7, 1]   (bad)        3    0.5%   <NA>
   (1, Inf)   (very bad)   0    0.0%   <NA>
See help('pareto-k-diagnostic') for details.
```

```
loo_compare(loo_f1, loo_f2, loo_f3)
```

```
Error in eval(expr, envir, enclos): object 'loo_f1' not found
```

```
###
```

Rubric

- Do the results look correct and have they been presented in a readable way? They should be roughly .
- Has it been recognized that the best model is ?

Subtask 2.j

Assess whether the approximation to the LOO-CV distributions are reliable. Consult the $\hat{k}$ statistic which informs on the reliability of PSIS computation in PSIS-LOO. Plot the $\hat{k}$ values for each model against the data point ID and discuss. Are they as expected?

Tip

For hierarchical models, it may be more important to think about how well the individual group is predicted and how many observations there are in a group compared to the number of parameters estimated. Also check out CV-FAQ on high Pareto-$\hat{k}$ values.

```
# Useful functions: plot(loo(...), label_points = TRUE)
### {.content-hidden when-profile="public"}
plot(loo(f1), label_points = TRUE)
```

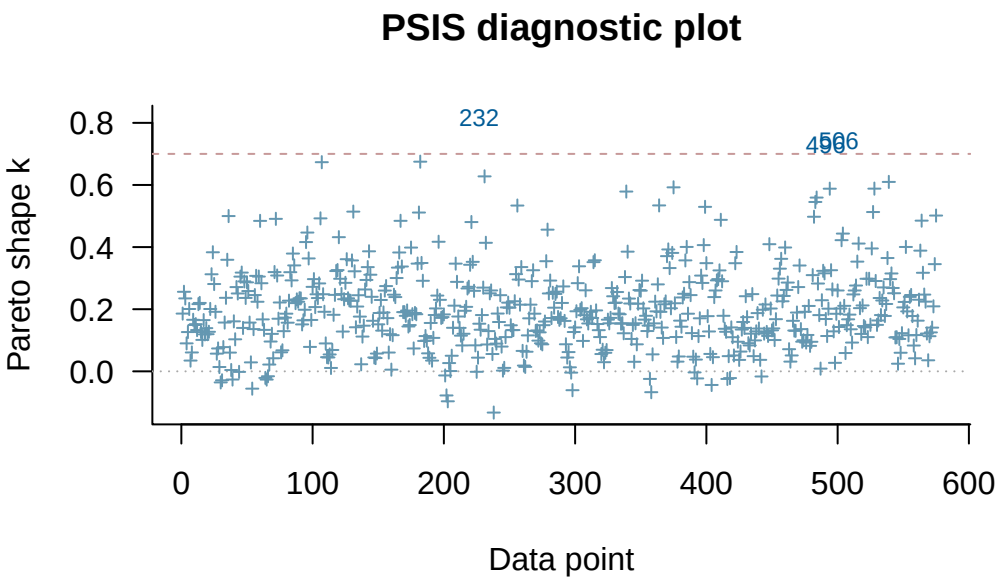
```
Error in h(simpleError(msg, call)): error in evaluating the argument 'x' in selecting a method for fun
```

```
plot(loo(f2), label_points = TRUE)
```

```
Error in h(simpleError(msg, call)): error in evaluating the argument 'x' in selecting a method for fun
```

```
plot(loo(f3), label_points = TRUE)
```

```
Warning: Found 3 observations with a pareto_k > 0.7 in model 'f3'. We recommend
to set 'moment_match = TRUE' in order to perform moment matching for
problematic observations.
```




```
###
```

Rubric

- Do the plots look correct and are they readable?
- Has it been explained why the $\hat{k}$ values are highest for the .

Subtask 2.k

Perform a PPC for the hierarchical model for

- a few of the chickens with the highest $\hat{k}$ values and
- a few of the chickens with the lowest $\hat{k}$ values

using the code in the template. What do you observe?

⚠ Creating a dummy example plot

Creating a dummy fit just to be able to generate an example plot below. Generate a similar plot for your hierarchical model.

```
# The brms-formula (weights ~ ...) below is not one that you should be using in your models!
dummy_fit <- brms::brm(
  weight ~ 1 + Time + Chick,
  data = ChickWeight,
  file="additional_files/assignment8/dummy_fit"
)
# Adjust the chicken_idxs variable to select appropriate chickens
chicken_idxs = c(1,3,11,43)
# Create this plot for your hierarchical model for selected chickens
brms::pp_check(
  dummy_fit, type = "intervals_grouped", group = "Chick",
  newdata=ChickWeight |> filter(Chick %in% chicken_idxs)
)
```

Rubric

- Does the plot look correct and is it readable?
- Has it been recognized that the chickens with high $\hat{k}$ values ?

2.6 Model comparison using the RMSE

Subtask 2.l

Use the function in the template to compare the RMSE and the LOO-RMSE for the three models. Explain the difference between the RMSE and the LOO-RMSE in 1–3 sentences. Is one generally lower than the other? Why?

⚠ `rmse` function implementation

The below function takes a brms fit object and computes either the root-mean-square error (RMSE) or the PSIS-LOO-RMSE, i.e. the RMSE using LOO-CV estimated using PSIS-LOO.

```r
# Compute RMSE or LOO-RMSE
rmse <- function(fit, use_loo=FALSE){
  mean_y_pred <- if(use_loo){
    brms::loo_predict(fit)
  }else{
    colMeans(brms::posterior_predict(fit))
  }
  sqrt(mean(
    (mean_y_pred - brms::get_y(fit))^2
  ))
}
```

Rubric

- Do the results look correct and have they been presented in a readable way? They should be roughly: .
- Has it been recognized that the RMSE is ?

3 Overall quality of the report

Rubric

- Does the report include comment on whether AI was used, and if AI was used, explanation on how it was used?
  - No
  - Yes
- Does the report follow the formatting instructions?
  - Not at all
  - Little
  - Mostly
  - Yes
- In case the report doesn't fully follow the general and formatting instructions, specify the instructions that have not been followed. If applicable, specify the page of the report, where this difference is visible. This will help the other student to improve their reports so that they are easier to read and review. If applicable, specify the page of the report, where this difference in formatting is visible.
- Please also provide feedback on the presentation (e.g. text, layout, flow of the responses, figures, figure captions). Part of the course is practicing making data analysis reports. By providing feedback on the report presentation, other students can learn what they can improve or what they already did well. You should be able to provide constructive or positive feedback for all non-empty and non-nonsense reports. If you think the report is perfect, and you can't come up with any suggestions how to improve, you can provide feedback on what you liked and why you think some part of the report is better than yours.