# STAT 547: Bayesian Workflow

Charles C. Margossian
University of British Columbia
Winter 2026
https://charlesm93.github.io/stat_547/

**DRAFT**

## 3 Prediction and model comparison

Even if we trust the inference, we need to check whether the fitted model "works."

Question: what does it mean for a fitted model to work?

- Accurate predictions
- Accurate recommendations, decisions, ...
- Accurate inference on an unknown quantity of interest.

Remark: In ML, it is common to check a trained model without checking the inference. Is this a good practice?

Most applications of a model imply we are making decent predictions. So let's at least check that! Let $y_{1:N}$ be our data and $\theta$ our model parameters. For simplicity, let's assume that, conditional on $\theta$, the observations are independent, that is

$$p(y_{1:N} \mid \theta) = \prod_{n=1}^{N} p(y_n \mid \theta). \tag{1}$$

Let's suppose we want to make a prediction on a new observation, $y_{N+1}$. There are several ways to make this prediction:

- **Point prediction.** For a fixed $\hat{\theta}$, compute an estimator of $\hat{y}_{N+1} = \mathbb{E}(\hat{y}_{N+1} \mid \hat{\theta})$.
- **Distribution prediction.** Instead of doing a point prediction, we may consider a distribution of predictions, using the likelihood, $p(\hat{y}_{N+1} \mid \hat{\theta})$.
- **Posterior distribution of predictions.** In a Bayesian setting, we want to examine the posterior distribution of the predictions,

$$\begin{aligned} p(y_{N+1} \mid y_{1:N}) &= \int p(y_{N+1}, \theta \mid y_{1:N}) \mathrm{d}\theta \\ &= \int p(y_{N+1} \mid \theta, y_{1:N}) p(\theta \mid y_{1:N}) \mathrm{d}\theta \\ &= \int p(y_{N+1} \mid \theta) p(\theta \mid y_{1:N}) \mathrm{d}\theta, \end{aligned} \tag{2}$$

  where we used conditional independence to get the last line.

A natural way to study this posterior distribution is by drawing samples from $p(y_{N+1} \mid y_{1:N})$. This is easy to do once we have (approximate) samples from $p(\theta \mid y_{1:N})$. In particular, we can use a two step process:

(i) $\theta^{(i)} \sim p(\theta \mid y_{1:N})$

(ii) $y_{N+1}^{(i)} \sim p(y_{N+1} \mid \theta^{(i)})$.

In Stan, we already know how to do (i). It remains to do (ii). This can be done in post-process or using the `generated quantities` block.

## 3.1 Posterior predictive check

(Sometimes aptly called *posterior retrodictive checks.*[1])

As a first step, we're going to replicate the training data. That is, we're going to sample $\tilde{y}_{1:N}$ and see how closely the model's simulation match the observations we used to train the model.

**Coding demo.** Posterior predictions for the SIR model with a Poisson likelihood.

Informally, we want to check that the simulated data captures all the meaningful characteristics in the data. One way to do this is to plot the observations along with the simulations. In our disease transmission example, we plot the posterior median and 90 % posterior interval of $p(\tilde{y}_{1:N} \mid \theta)$.

Some questions to ask:

- Does the posterior median capture the central tendency of the observations?
- Is the uncertainty well-calibrated? For example, do 90 % of the observations fall within the 90 % posterior interval? Is the model too confident? Not confident enough?
- Are there some problematic outliers that suggest our model is missing some important mechanisms in the data generating process?

To illustrate how insightful these simple visual checks can be, we're going to fit another model to the same influenza data set. Namely an SIR model with a negative Binomial likelihood.

How do the two posterior predictive checks compare? Does a model better capture the characteristics of the data?

## 3.2 Out-of-sample predictions

Reconstructing the observations used to train the model provides an important sanity check but it only tells us so much about a model's ability to make predictions.

A common practice is to split the data into a *training* and a *validation* set and see how well the model predicts data it has not "seen".

Question: How should we split the data into a training and a validation set?

[1] `https://betanalpha.github.io/assets/case_studies/principled_bayesian_workflow.html`

## 3.3 Prediction score

Visual checks are not always convenient, let alone feasible. Usually we also report a quantitative measure of how good the prediction is.

**Example 1. (Continuous observation)** Suppose we have a normal likelihood, with estimated mean $\hat{\mu}_n$ and standard deviation $\hat{\sigma}_n$, indexed by $n$. Our best prediction is

$$\hat{y}_n = \hat{\mu}_n. \tag{3}$$

Then, we might report the squared prediction error,

$$\text{Err} = (y_n - \hat{\mu}_n)^2 . \tag{4}$$

This is a valid measure for the quality of our prediction but it doesn't tell us anything about $\hat{\sigma}_n$ or how confident our model is in its prediction.

Instead, we can use the *point-estimate log predictive density*,

$$\begin{aligned} \text{p-lpd} &= \log p(y_n \mid \hat{\mu}_n, \hat{\sigma}_n^2) & (5) \\ &= k - \log \hat{\sigma}_n - \frac{1}{2\hat{\sigma}_n^2} (y_n - \hat{\mu}_n)^2 . & (6) \end{aligned}$$

**Example 2. (Binary observation)** Suppose we have a Bernoulli likelihood with estimated parameter $\hat{\pi}_n$. Our best prediction is

$$\hat{y}_n = \mathbb{I}(\hat{\pi}_n \geq 0.5). \tag{7}$$

The the prediction error is

$$\text{Err} = \mathbb{I}(\hat{y}_n = y_n). \tag{8}$$

Again, this doesn't really tell us whether the model is too confident or not. Here, the *point-estimate log predictive density* is

$$\begin{aligned} \text{p-lpd} &= \log p(y_n \mid \hat{\pi}_n) & (9) \\ &= y_n \log \hat{\pi}_n + (1 - y_n) \log(1 - \hat{\pi}_n). & (10) \end{aligned}$$

This is the cross-entropy loss function (derived from a probabilistic perspective)!

In a Bayesian setting, we consider the point-estimate log predictive density averaged over the posterior distribution. This leads to the *expected log predictive density*,

$$\text{elpd} = \log p(y_{N+1} \mid y_{1:N}) = \log \int p(y_{N+1} \mid \theta) p(\theta \mid y_{1:N}) \text{d}\theta. \tag{11}$$

Question: Equipped with this notion, how should we compare our two disease transmission models?

We need to agree on which test set to use. A common choice is to do cross-validation, where we average over multiple test-sets. In *leave-one-out* cross-validation, we only exclude one observation and train the model over all the other observations. Then we compute the elpd at each observation and take the average,

$$\text{elp}_{\text{loo}} = \frac{1}{N} \log p(y_n \mid y_{-n}). \tag{12}$$

But this requires training the model $N$ times, which is expensive...

Idea: We're going to find a fast approximation of $p(\theta \mid y_{-n})$, based on $p(\theta \mid y_{1:N})$.

## 3.4 Importance sampling estimator

Importance sampling (IS) provides a general framework to do Monte Carlo estimation when we have samples from a proposal distribution $q(z)$, rather than our target distribution $p(z)$.

The main relation we use is,

$$\int f(z)p(z)\text{d}z = \int f(z)\frac{p(z)}{q(z)}q(z)\text{d}z. \tag{13}$$

We denote $w(z) = p(z)/q(z)$ the *importance weights*. With draws from $q$, we can in principle construct an unbiased Monte Carlo estimator of $\mathbb{E}_p f$,

$$\hat{f}_{IS} = \frac{1}{S}\sum_{s=1}^{S} f\left(z^{(s)}\right) w\left(z^{(s)}\right). \tag{14}$$

The main challenge with importance sampling is that the importance weight may have an infinite variance, if they are regions where $q(z)$ is small and $p(z)$ is large.

When computing the elpd at $y_n$ , $p(\theta \mid y_{1:N})$ is our proposal distribution and $p(\theta \mid y_{-n})$ our target distribution. Then, applying Bayes' rule and using the conditional independence of the likelihood,

$$\begin{aligned} w(\theta) &= \frac{p(\theta \mid y_{-n})}{p(\theta \mid y_{1:N})} \\ &\propto \frac{p(\theta)p(y_{-n} \mid \theta)}{p(\theta) \mid p(y_{1:N} \mid \theta)} \\ &= \frac{\prod_{j \neq n} p(y_j \mid \theta)}{\prod_{j=1}^{N} p(y_j \mid \theta)} \\ &= \frac{1}{p(y_n \mid \theta)} =: r_n(\theta). \end{aligned} \tag{15}$$

To be explicit, we denote $c$ the missing constant, so that $w(\theta) = cr_n(\theta)$. Then, we have the following importance sampling estimator of $\log p(y_n \mid y_{-n})$,

$$\widehat{\text{elpd}}_n = \frac{1}{S}\sum_{s=1}^{S} cr_n\left(\theta^{(s)}\right) \log p\left(y_n \mid \theta^{(s)}\right). \tag{16}$$

We need to find a way to deal with the unknown constant $c$. To do so, we consider the Monte Carlo estimator,

$$\frac{1}{S}\sum_{s=1}^{S} cr\left(\theta^{(s)}\right) = \frac{1}{S}\sum_{s=1}^{S} w\left(\theta^{(s)}\right) \longrightarrow \int \frac{p(\theta \mid y_{-n})}{p(\theta \mid y_{1:N})} p(\theta \mid y_{1:N}) \mathrm{d}\theta = \int p(\theta \mid y_{-n}) \mathrm{d}\theta = 1. \quad (17)$$

This suggests a consistent estimator, sometimes called the *self-normalized IS estimator*,

$$\begin{aligned} \widehat{\text{elpd}}_n &= \frac{1}{S}\frac{\sum_{s=1}^{S} cr_n\left(\theta^{(s)}\right)\log p\left(y_n \mid \theta^{(s)}\right)}{\frac{1}{S}\sum_{s=1}^{S} cr_n\left(\theta^{(s)}\right)} \\ &= \frac{\sum_{s=1}^{S} r_n\left(\theta^{(s)}\right)\log p\left(y_n \mid \theta^{(s)}\right)}{\sum_{s=1}^{S} r_n\left(\theta^{(s)}\right)}, \end{aligned} \quad (18)$$

where, bearing an abuse of notation, we denote $\widehat{\text{elpd}}$ the above estimator. Crucially, this estimator does not depend on $c$ and can be computed, provided we can evaluate the likelihood $p(y_n \mid \theta)$.

## 3.5 Trunctated and Pareto-smooth IS estimator

Unfortunately, IS estimators as in eq. (18) can suffer from a large and even unbounded variance. We'll briefly review some strategies to control the variance of IS estimators.

One approach is to bound the ratios via a truncation rule,

$$\tilde{r}_n = \min(r_n, \sqrt{S}\bar{r}). \quad (19)$$

It can be shown that, provided $r_n$ has a finite first moment ($\mathbb{E}(|r_n|) < \infty$), plugging $\tilde{r}_n$ into eq. (18) bounds the variance of $\widehat{\text{elpd}}_n$ but introduces a bias that can be large.

A better strategy is to use Pareto-smoothing. This topic is a bit of a rabbit hole and here we'll only cover it at a high level. The main idea is to fit the tail distribution of $r_n$,

$$p(r_n \mid r_n > u), \quad (20)$$

for some threshold $u$. In practice, this distribution is often well approximated by a generalized Pareto distribution,

$$p(y \mid \mu, \sigma, k) = \begin{cases} \frac{1}{\sigma}\left(1 + k\left(\frac{y-\mu}{\sigma}\right)\right)^{-\frac{1}{k}-1} & , k \neq 0, \\ \frac{1}{\sigma}\exp\left(\frac{y-\mu}{\sigma}\right) & , k = 0. \end{cases} \quad (21)$$

We fit the distribution to the tails of the IS weights using estimators $\hat{k}$, $\hat{\mu}$ and $\hat{\sigma}$. A key characteristic of the Pareto distribution is that it has $1/k$ finite moments. That is,

$$\mathbb{E}|x^p| \begin{cases} < \infty & , \text{ if } p \leq 1/k \\ = \infty & , \text{ if } p > 1/k. \end{cases} \quad (22)$$

Hence, the $\hat{k}$ estimator can tell us whether the tail weights have a finite variance or not. Specifically, the variance is finite if $\hat{k} \leq 0.5$. Sometimes, we can get away with $\hat{k} > 0.5$, because even if the tail weights have an unbounded variance, the $\widehat{\text{elpd}}_n$ can still have a finite variance.[2]

[2] For this, we can invoke generalizations of the central limit theorem, with subefficient convergence rates.

One heuristic is to check that $\hat{k} < 0.7$ for all $\widehat{\text{elpd}}_n$. If for a particular $n$, $\hat{k} > 0.7$ we may need to refit the model with the $n^{\text{th}}$ data point excluded or come up with a different estimation strategy.

The Pareto smoothed Importance Sampling (PSIS) estimator can be computed using the R package `loo`.

**Coding demo:** Use loo-cv to compare the SIR model with a Poisson likelihood and a negative binomial likelihood.

For a more detailed discussion on the topic, see the excellent paper by Vehtari et al. [2017].

## References

A. Vehtari, A. Gelman, and J. Gabry. Practical bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27(5):1413–1432, 2017.