Project 2

Two main template files are available; modify these but keep their names:

- A template RMarkdown document, `report.Rmd`, that you should expand with your answers and solutions for this project assignment,
- A template `code.R` file where you should place function definitions needed by `report.Rmd`, with associated documentation.

Instructions:

- Include your name and student number in both `report.Rmd` and `code.R`.
- Code that displays results (e.g., as tables or figures) must be placed as code chunks in `report.Rmd`.
- Code chunks that load the function definitions from `code.R` are included in `report.Rmd` (your own function definitions).
- You must upload the following three files as part of this assignment: `code.R`, `report.Rmd` and a generated `html` version of the report with the file name `report.html` and you you will be marked based on the `report.html` file.
- Appendix code chunks display the code from `code.R` without running it.
- Use the `styler` package Addin for restyling code for better and consistent readability works for both `.R` and `.Rmd` files.
- Use `echo=TRUE` for analysis code chunks and `echo=FALSE` for table display and plot-generating code chunks.
- The main text in your report should be a coherent presentation of theory and discussion of methods and results, showing code chunks that perform computations and analysis.

The project will be marked between 0 and 40, using the marking guide CWmarking posted on Learn.

Submission should be done through Gradescope.

Part 1: 3D printer

Here, you will use the `filament1` dataset from Project 1, which contains information about a 3D printer that uses rolls of *filament* that get heated and squeezed through a moving nozzle, gradually building objects. The objects are designed using a CAD program (Computer Aided Design) that estimates how much material will be required to print them. The data file `"filament1.rda"` contains information about one 3D-printed object per row. The columns are

- `Index`: an observation index
- `Date`: printing dates
- `Material`: the printing material, identified by its colour
- `CAD_Weight`: the object weight (in grams) that the CAD software calculated
- `Actual_Weight`: the actual weight of the object (in grams) after printing

Consider two linear models, A and B, for capturing the relationship between `CAD_Weight` and `Actual_Weight`. Denote the `CAD_weight` for observation i x_i, and the corresponding `Actual_Weight` y_i. As in Project 1, the two models are defined by

- Model A: $y_i \sim \mathsf{Normal}[\beta_1 + \beta_2 x_i, \exp(\beta_3 + \beta_4 x_i)]$
- Model B: $y_i \sim \mathsf{Normal}[\beta_1 + \beta_2 x_i, \exp(\beta_3) + \exp(\beta_4) x_i^2]$

Recall that in Project 1, you built functions to estimate the parameters from Models A and B. In exercises 1-5 described below, you will build predictive distributions for models A and B and access the performance of both models to predict new data.

1. Create a function `filament1_predict` with similar behavior to the `R` built in `lm.predict` that computes predictive distributions and 95% prediction intervals for a new dataset. Your function should return a `data.frame` with variables `mean`, `sd`, `lwr`, and `upr`, summarizing the predictive distribution for each row of the new data. The function `filament1_aux_EV` in `code.R` evaluates the expectation and variance for model A and B and can be used to help constructing the predictive distributions in `filament1_predict`. The arguments `theta` and `Sigma_theta` needed in `filament1_aux_EV` can be obtained by first estimating models A and B using the `filament1_estimate` and `neg_log_lik` functions in `code.R` to retrieve the estimated expected value and variance for each model (recall from Project 1). Next, use your function to compute probabilistic predictions of `Actual_Weight` using the two estimated models and the `filament1` as new data. Use a level of significance of 5% for computing the prediction intervals. Note that in this exercise, the data for estimating and predicting the models are the same. Inspect the predictions visually.

2. Write functions to evaluate the squared error (ES) and Dawid-Sebastiani (DS) scores.

3. Here, you will use the functions created in the previous exercises to perform leave-one-out cross-validation. Implement a function `leave1out` that performs leave-one-out cross-validation for the selected model for each observation $i = 1, \dots, N$. In this function, you should first estimate the model parameters using $\{(x_j, y_j), j \neq i\}$, compute the prediction information based on x_i for prediction model F_i, and then compute the required scores $S(F_i, y_i)$. The output should be a `data.frame` that extends the original data frame with four additional columns `mean`, `sd`, `se` and `ds` of leave-one-out prediction means, standard deviations, and prediction scores. Use `leave1out` to obtain average leave-one-out scores $S(F_i, y_i)$ for each model and type of score for `filament1` data. Present the results and interpret them.

4. Construct a Monte Carlo estimate of the p-value to test the exchangeability between model predictions from A and B against the alternative hypothesis that B is better than A. Compute Monte Carlo standard errors, draw conclusions, and investigate whether one model is better at predicting than the other.

Part 2: Archaeology in the Baltic sea

In 1361, the Danish king Valdemar Atterdag conquered Gotland. A plunder of Visby followed the conquest. Most of the defenders (primarily local farmers who could not take shelter inside the city walls) were killed in the attack and were buried in a field outside of the walls of Visby. In the 1920s, the gravesite was subject to several archaeological excavations. A total of 493 femurs (256 left and 237 right) were found. One question that arises is the number of people that were buried at the gravesite. It must reasonably have been at least 256, but how many more?

To build a simple model for this problem, assume that the number of left ($y_1 = 256$) and right ($y_2 = 237$) femur are two independent observations from a $\mathsf{Bin}(N, \phi)$ distribution. Here N is the total number of people buried and ϕ is the probability of finding a femur, left or right, and both N and ϕ are unknown parameters.

The probability function for a single observation $y \sim \mathsf{Bin}(N, \phi)$ is

$$p(y|N, \phi) = \binom{N}{y} \phi^y (1-\phi)^{N-y}.$$

The combined *log-likelihood* $l(y_1, y_2|N, \phi) = \log p(y_1, y_2|N, \phi)$ for the data set $\{y_1, y_2\}$ is then given by

$$\begin{aligned} l(y_1, y_2|N, \phi) = &- \log\Gamma(y_1+1) - \log\Gamma(y_2+1) \\ &- \log\Gamma(N-y_1+1) - \log\Gamma(N-y_2+1) + 2\log\Gamma(N+1) \\ &+ (y_1+y_2)\log(\phi) + (2N - y_1 - y_2)\log(1-\phi) \end{aligned}$$

Before the excavation took place, the archaeologist believed that around 1000 individuals were buried, and that they would find around half on the femurs. To encode that belief in the Bayesian analysis, set $\xi = 1/(1+1000)$, which corresponds to an expected total count of 1000, and $a = b = 2$, which makes ϕ more likely to be close to 1/2 than to 0 or 1.

The posterior probability function for $(N, \phi|\boldsymbol{y})$ is

$$p_{N,\phi|\boldsymbol{y}}(n, \phi|\boldsymbol{y}) = \frac{p_{N,\phi,\boldsymbol{y}}(n, \phi, \boldsymbol{y})}{p_{\boldsymbol{y}}(\boldsymbol{y})} = \frac{p_N(n)p_\phi(\phi)p(\boldsymbol{y}|n, \phi)}{p_{\boldsymbol{y}}(\boldsymbol{y})}.$$

Let N have a $\mathsf{Geom}(\xi)$, $\xi > 0$, prior distribution, and let ϕ have a $\mathsf{Beta}(a, b)$, $a, b > 0$, prior distribution:

$$\begin{aligned} p_N(n) &= \mathsf{P}(N = n) = \xi\,(1-\xi)^n, \quad n = 0, 1, 2, 3, \ldots, \\ p_\phi(\phi) &= \frac{\phi^{a-1}(1-\phi)^{b-1}}{B(a,b)}, \quad \phi \in [0, 1]. \end{aligned}$$

Monte Carlo integration can be used to estimate the posterior expectations of N and ϕ. Consider prior distributions $N \sim \mathsf{Geom}(\xi)$, $0 < \xi < 1$, and $\phi \sim \mathsf{Beta}(a, b)$, $a, b > 0$. Furthermore, consider the prior distributions as sampling distributions for the Monte Carlo integration, such that samples $n^{[k]} \sim \mathsf{Geom}(\xi)$ and $\phi^{[k]} \sim \mathsf{Beta}(a, b)$, cand be used to compute Monte Carlo estimates:

$$\begin{aligned} \widehat{p}_{\boldsymbol{y}}(\boldsymbol{y}) &= \frac{1}{K}\sum_{k=1}^{K} p(\boldsymbol{y}|n^{[k]}, \phi^{[k]}) \\ \widehat{\mathsf{E}}(N|\boldsymbol{y}) &= \frac{1}{K\widehat{p}_{\boldsymbol{y}}(\boldsymbol{y})}\sum_{k=1}^{K} n^{[k]} p(\boldsymbol{y}|n^{[k]}, \phi^{[k]}) \\ \widehat{\mathsf{E}}(\phi|\boldsymbol{y}) &= \frac{1}{K\widehat{p}_{\boldsymbol{y}}(\boldsymbol{y})}\sum_{k=1}^{K} \phi^{[k]} p(\boldsymbol{y}|n^{[k]}, \phi^{[k]}) \end{aligned}$$

1. Write a function `arch_loglike` that evaluates the combined log-likelihood $\log[p(\boldsymbol{y}|N, \phi)]$ for a collection $\boldsymbol{y}$ of y-observations. If a `data.frame` with columns `N` and `phi` is provided, the log-likelihood for each row-pair (N, ϕ) shoule be returned.
2. Write a function `estimate` that takes inputs `y`, `xi`, `a`, `b`, and `K` and implements the above Monte Carlo integration method to approximate $p_{\boldsymbol{y}}(\boldsymbol{y})$, $\mathsf{E}(N|\boldsymbol{y})$, and $\mathsf{E}(\phi|\boldsymbol{y})$. Test the function by running `estimate(y=c(237,256), xi=1/1001, a=0.5, b=0.5, K=10000)` and comment on the results.