INF367: Spring 2020
Project 1: Bayesian linear regression with ARD priors

Deadline: April 3rd, 23.59
Deliver here: `https://mitt.uib.no/courses/23594/assignments/28325`

Deliverables. You should return **exactly two** files:

1. A pdf file containing a report; see Section 2 for more details
2. A zip file containing code

Grading: Grading will be based on the following criteria:

- Correctness (your answers/code are correct and clear)
- Experimental design and insight in the simulation study
- Reporting (thoroughness and clarity of the report)
- Clarity of code (documentation, naming of variables, logical formatting)

This project will count 15% towards your final grade.

Late submission policy: All late submissions will get a deduction of 2 points. In addition, there is a 2-point deduction for every starting 12-hour period. That is, a project submitted at 00.01 on April 4th will get a 4-point deduction and a project submitted at 12.01 on the same day will get a 6-point deduction (and so on). All projects submitted on April 6th or later are automatically failed. (Executive summary: Submit your project on time.) There will be no possibility to resubmit failed projects so start working early.

If you are sick during the project period, the deadline will be extended accordingly.

1 Bayesian linear regression with automatic relevance determination (ARD)

Overfitting is a fundamental challenge in machine learning. Linear models can overfit in many scenarios. For example, if there are lots of irrelevant features or

if one uses too complicated basis functions. Typically, one trains several models with varying complexity and uses model selection to choose a model that fits best to the data.

One way to avoid overfitting is to do dimensionality reduction as a preprocessing step. Automatic relevance determination (ARD) priors try to avoid overfitting by doing feature selection implicitly by pushing some weights to zero effectively selecting a simpler model.

Let us consider now consider a "standard" Bayesian linear regression and the ARD version.

1.1 "Standard" Bayesian linear regression

Let us begin with a standard Bayesian linear regression model (same as in Exercise 9). Assume that we have observed n pairs $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i$ are the feature values and y_i is the label. Let $\mathbf{w} \in \mathbb{R}^d$ be our regression weights. The likelihood is

$$P(y \,|\, \mathbf{x}, w, \beta) = \prod_{i=1}^{n} N(y_i \,|\, \mathbf{w}^T \mathbf{x}_i, \beta^{-1}).$$

The parameter β has a Gamma prior

$$P(\beta) = Gamma(\beta \,|\, a_0, b_0)$$

where a_0 and b_0 are user-defined hyperparameters. The regression weights have a Gaussian prior

$$P(\mathbf{w}) = N(\mathbf{w} \,|\, \mathbf{0}, \alpha^{-1}\mathbf{I})$$

where α (prior precision) is a user-defined hyperparameter. Intuitively, α determines the amount of regularization. The larger α, the more regularization.

The goal is to compute the posterior

$$P(\mathbf{w}, \beta \,|\, \mathbf{x}, y).$$

1.2 Automatic relevance determination (ARD)

The standard linear regression regularizes all features equally. Thus, adding more regularization to avoid overfitting may push also the weights of the important weights towards zero and lead to underfitting. The idea of automatic relevance determination is to use a flexible prior to push weights of the irrelevant features to zero while not penalizing the relevant features.

Assume that we have observed n pairs $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i$ are the feature values and y_i is the label. Let $\mathbf{w} \in \mathbb{R}^d$ be our regression weights. The likelihood is

$$P(y \,|\, \mathbf{x}, w, \beta) = \prod_{i=1}^{n} N(y_i \,|\, \mathbf{w}^T \mathbf{x}_i, \beta^{-1}).$$

For the noise precision β, we use a Gamma prior

$$P(\beta) = Gamma(\beta \,|\, a_0, b_0)$$

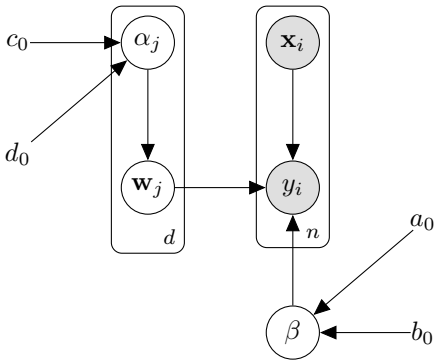


Figure 1: Plate diagram of the regression model with ARD prior. Variables $\mathbf{x}_i$, and y_i are observed; We are interested in the posterior distribution of parameters $\mathbf{w}$, α and β. The constants a_0, b_0, c_0, and d_0 are user-defined hyperparameters

where a_0 and b_0 are user-defined hyperparameters.

The prior for the weights $\mathbf{w}$ is a Gaussian distribution

$$P(\mathbf{w} \,|\, \alpha_1, \dots \alpha_d) = \prod_{j=1}^{d} N(w_j \,|\, 0, \alpha_j^{-1}).$$

Notice that the difference compared to the standard case is that instead of having one precision parameter α that is same for all dimensions, we have a separate prior precision α_j for each dimension. Furthermore, α_j is not assumed to be known but we place a prior on it. The prior for α_j is a Gamma distribution

$$P(\alpha_j) = Gamma(\alpha_j \,|\, c_0, d_0) \quad \forall j = 1, \dots, d$$

where c_0 and d_0 are user-defined hyperparameters. This allows the model to learn automatically which features should be regularized strongly and which not.

The posterior is

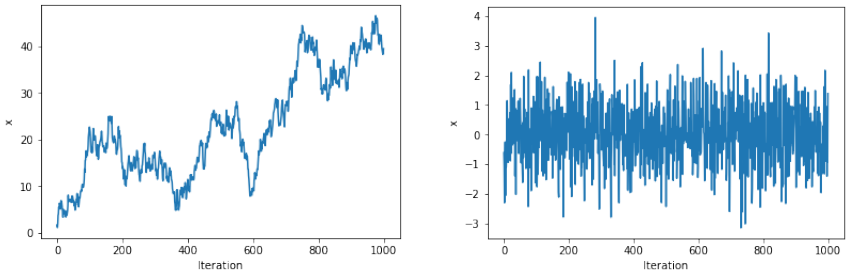
$$P(\mathbf{w}, \alpha, \beta \,|\, y, \mathbf{x}) \propto \prod_{i=1}^{n} P(y_i \,|\, \mathbf{w}, \mathbf{x}_i, \beta) \prod_{j=1}^{d} [P(w_j \,|\, \alpha_j) P(\alpha_j)] P(\beta).$$

A plate diagram visualizing the ARD model is shown in Figure 1.

1.3 Tasks

1.3.1 Deriving the formulas and implementation

Derive a Gibbs sampler for the ARD regression model. To this end, you have to derive the full conditional distributions $P(\mathbf{w} \,|\, y, \mathbf{x}, \alpha, \beta)$, $P(\alpha \,|\, y, \mathbf{x}, \mathbf{w}, \beta)$, and $P(\beta \,|\, y, \mathbf{x}, \mathbf{w}, \alpha)$. Here we use a shorthand $\alpha = (\alpha_1, \dots, \alpha_d)$



(a) (b)

Figure 2: Two traceplots. Traceplot (a) shows that the chain has not converged and/or is not mixing well. Traceplot (b) shows good mixing.

Implement the Gibbs sampler. You should also compute the unnormalized log-posterior

$$\sum_{i=1}^{n} \log P(y_i \,|\, \mathbf{w}, \mathbf{x}_i, \beta) + \log P(\mathbf{w} \,|\, \alpha) + \log P(\alpha) + \log P(\beta)$$

after each iteration.

Hints:

When deriving full conditional distributions, drop all terms that do not involve the variable in question (those terms are constants). All updates involve conjugate priors and thus you should get closed-form solutions.

1.3.2 Simulation study

Conduct a small simulation study to verify whether the ARD priors can mitigate overfitting. We want to answer the following questions: When does ARD work? When doesn't it work?

Start by generate data sets: That is, choose a function f, generate random values for $\mathbf{x}_i$ and then generate the label $y_i = f(\mathbf{x}_i) + \epsilon_i$ where ϵ_i is random noise (typically Gaussian). Vary the function f, the number of dimensions, the number of samples, the amount of noise (variance of the Gaussian). Note: we are especially interested in scenarios where there are lots of irrelevant features (feature x_i does not have effect on the label y).

Then learn models using both ARD and the standard model. Compare results. Use appropriate measures to assess performance. Visualize the results.

Assess convergence of the Gibbs sampling. It is ok to do this visually by looking at the traceplots. If you are interested, you can also use various other convergence analysis methods (see, for example, `https://arxiv.org/pdf/1909.11827.pdf`.

Hints:

You can plot functions from the posterior distribution (especially when $\mathbf{x}$ is

one-dimensional). Just use the weights from different samples. As the subsequent samples are correlated, use thinning.

Predictive distributions can be approximated using the procedure on slide 36 of the lectures slides about sampling.

You can visualize chains using traceplots; see Figure 2 for examples. Marginal distributions can be visualized using histograms.

1.3.3 Real data

Load the data set `Lung_cancer_small.csv` [1]. We are going to predict the survival time of the patients[2]. That is, we predict the variable y given all the other variables. The original data has 945 features but our data contains only the first 100 features.

Learn the linear regression model using this data. Assess convergence and mixing. Do your results make sense? What variables affect the survival time most?

2 Report

The report should contain a clear and thorough description of what you have done and what conclusions you have made. Specifically, you should answer at least the following questions:

What are the full conditional distributions of your Gibbs sampler and how did you derive them? How does your Gibbs sampler work?

Explain your simulation study. What are the specific questions that each test tries to answer? How did you generate the data? How did you measure performance? What do you conclude from the results? Remember to visualize your results!

How did the ARD model work on real data?

[1] More information about the data can be found here: `https://rdrr.io/cran/PRIMsrc/man/Real.2-data.html`.

[2] The results are going to be slightly pessimistic because the data is *censored*, that is, some patients were alive when the study ended and in their cases, the variable y records the time when the study ended (these are the cases with delta=1). The proper way would be to use so-called censored regression methods to correct this, but here we do the slightly misleading analysis to keep things simple.