MLSP 2014 Schizophrenia Classification Challenge: Winning Model Documentation

Arno Solin
arno.solin@aalto.fi
Doctoral student at Aalto University
Espoo, Finland

Simo Särkkä (instructor)
simo.sarkka@aalto.fi
Aalto University
Espoo, Finland

Abstract

This technical note presents the idea and methods behind the winning solution for the MLSP 2014 Schizophrenia Classification Challenge organized on Kaggle. This challenge took place between June 5 and July 20, 2014, and 341 teams submitted solutions. The winning model 'Solution Draft' was based on a Bayesian machine learning paradigm known as Gaussian process (GP) classification.

1 Summary

The goal of the competition [1] was to automatically diagnose subjects with schizophrenia based on multimodal features derived from their magnetic resonance imaging (MRI) brain scans. The winning proposition was based on a Gaussian process (GP, [2]) classifier, where the observations are considered to be drawn from a Bernoulli distribution. The probability is related to the latent function via a sigmoid function that transforms it to a unit interval. A GP prior with a covariance function as a sum of a constant, linear, and Matérn kernel was placed over the latent functions. The model was trained by sampling using the GPSTUFF toolbox [3].

2 Data and preprocessing

Data collection (partially described in [4]) was performed at the Mind Research Network, and funded by a Center of Biomedical Research Excellence (COBRE) grant 5P20RR021938/P20GM103472 from the NIH to Dr. Vince Calhoun. Both the training and test data are available on Kaggle [1].

The data consist of two sets of information collected by different imaging modalities: *Functional Network Connectivity* (FNC, [5]) and *Source-Based Morphometry* (SBM, [6]) loadings. The FNC were derived form functional magnetic resonance imaging (fMRI) scans, and can be seen as a functional modality feature describing the subject's overall level of 'synchronicity' between brain areas. SBM loadings are derived from structural MRI scans, and they indicate the concentration of grey matter in different regions of the subject's brain.

We denote the training data as $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$. The training data consist of $n = 86$ subjects, where $\mathbf{x}_i \in \mathbb{R}^{410}$ (378 from the FNC and 32 from the SBM, ignoring the constant first terms). The test data $\mathcal{D}_* = \{(\mathbf{x}_{*,i}, y_{*,i})\}_{i=1}^{n_*}$ consists of $n_* = 119{,}748$ rows (subjects) with unknown labels y_*. As a preprocessing step, we normalize each dimension in the inputs $\mathbf{x}_i$ and $\mathbf{x}_{*,i}$ by dividing them by the standard deviations from training inputs. The labels were transformed to $y_i \in \{-1, 1\}$.

3 Modeling techniques and training

The winning model was based on Gaussian process classification [2], where the latent functions are assumed to be realizations of a Gaussian process prior. In binary GP classification with observations,

$y_i \in \{-1, 1\}, i = 1, \ldots, n$, associated with inputs $\{\mathbf{x}\}_{i=1}^n$, the observations are considered to be drawn from a Bernoulli distribution with a success probability $p(y_i = 1 \mid \mathbf{x}_i)$. The probability is related to the latent function via a sigmoid function that transforms it to a unit interval. We use a probit transformation that defines the likelihood model

$$p(y_i \mid f(\mathbf{x}_i)) = \Phi(y_i f(\mathbf{x}_i)) = \int_{-\infty}^{y_i f(\mathbf{x}_i)} \mathcal{N}(z \mid 0, 1)\,\mathrm{d}z, \tag{1}$$

where $\Phi(\cdot)$ is the Gaussian cumulative distribution function. We use a Gaussian process prior to define a prior distribution over the latent functions

$$f \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}')). \tag{2}$$

The latent Gaussian process model is characterized by its covariance function (kernel) $k(\cdot, \cdot)$. We want to account for any linear structure plus some additional short-scale non-linearities in the latent space. Therefore we set up the covariance function as a linear combination of three separate covariance functions:

$$k(\mathbf{x}, \mathbf{x}') = k_{\text{const.}}(\mathbf{x}, \mathbf{x}') + k_{\text{linear}}(\mathbf{x}, \mathbf{x}') + k_{\text{Matérn}}^{\nu=5/2}(\mathbf{x}, \mathbf{x}'), \tag{3}$$

where the individual covariance functions were defined as (see [2] for a similar parametrization):

$$k_{\text{const.}}(\mathbf{x}, \mathbf{x}') = \theta_1, \tag{4}$$

$$k_{\text{linear}}(\mathbf{x}, \mathbf{x}') = \theta_2\, \mathbf{x}^{\mathsf{T}} \mathbf{x}', \quad \text{and} \tag{5}$$

$$k_{\text{Matérn}}^{\nu=5/2}(\mathbf{x}, \mathbf{x}') = \theta_3 \left(1 + \frac{\sqrt{5}r}{\theta_4} + \frac{5r}{3\theta_4^2}\right) \exp\!\left(-\frac{\sqrt{5}r}{\theta_4}\right), \tag{6}$$

where $r = \|\mathbf{x} - \mathbf{x}'\|$. The Matérn class has previously turned out to be suitable for spatio-temporal GP modeling in fMRI applications (see, *e.g.*, [7–9]).

The hyperparameters $\boldsymbol{\theta} = \{\theta_1, \theta_2, \theta_3, \theta_4\}$ of the model were given the following hyper-priors: $\theta_1, \theta_2, \theta_3 \sim$ Log-Uniform, and $\theta_4 \sim t_4(0, 1)$. The hyperparameters were initialized as $\boldsymbol{\theta} = \{1, 1, 1, 0.01\}$.

The training was started by running a Laplace approximation scheme on the model until convergence (see the codes), and then the final training was performed by sampling (1000 samples, 91 after removing burn-in and thinning). We used *elliptical slice sampling* [10] for the latent functions, and the *surrogate slice sampler* [11] for the hyperparameters. These samplers are the defaults in GPSTUFF [3], and they do not require any parameter tuning. The class label probabilities $p(y_{*,i} = 1 \mid \mathcal{D}, \mathbf{x}_{*,i})$ for the test set can now be predicted by the trained model by integrating over the latent functions. For more information and discussion on the methods, see the toolbox manual [12].

4 Code description

The codes that were used for training and prediction can be found at:

- `http://github.com/asolin/MLSP2014-kaggle-challenge`

All files are written in Mathworks Matlab, and running the scripts require installation of the GPSTUFF toolbox (see Sec. 5). The following files are provided:

- `settings.m` (Matlab)
    a. Specifies the path to the training data (`TRAIN_DATA_PATH`), test data (`TEST_DATA_PATH`), model (`MODEL_PATH`), and submission output directories (`SUBMISSION_PATH`). This is the only place that specifies the paths to these directories.
    b. The GPSTUFF toolbox is added to the Matlab path with appropriate initializations.
- `train.m` (Matlab)

   a. Read training data from `TRAIN_DATA_PATH` (specified in `settings.m`).
   b. Do the normalization steps described in Section 2.
   c. Set up and train the GP classifier (Note that the random number generator seed is not specified).
   d. Save the model under `MODEL_PATH` (specified in `settings.m`).

- `predict.m` (Matlab)
   a. Read the training and test data from `TRAIN_DATA_PATH` and `TEST_DATA_PATH`, and do the normalization steps described in Section 2.
   b. Load the model from `MODEL_PATH`.
   c. Use the model to make predictions on new samples.
   d. Save the predictions to `SUBMISSION_PATH`.

5 Dependencies

This solution builds heavily upon the GPSTUFF toolbox [3, 13] for Mathworks Matlab (and Octave). It is our in-house-developed software package for Gaussian process modeling. All codes were tested in Matlab 8.2.0.701 (R2013b), and GPSTUFF version 4.5 (release 2014-07-22, available online [13], and distributed under the GNU General Public License) in Ubuntu Linux.

6 How to generate the solution

The following steps should be taken to replicate the model training procedure:

1. Download and unpack the GPSTUFF toolbox [13].
2. Modify (to set the paths) and run `setup.m` in Matlab.
3. Run `train.m` in Matlab to train the GP classifier (note that the random seed is not fixed). The model is saved under the path specified in `setup.m`.
4. Run `predict.m` in Matlab to predict using the GP classifier. The model output is stored under the path specified in `setup.m`.

The winning model (serialized and saved) and submission CSV file are stored under `./model/` and `./submission/`, respectively.

7 Additional comments and observations

This particular GP classifier model was chosen by trying out a couple of models and comparing their performance by leave-one-out cross-validation (LOOCV). This model did show promising performance using LOOCV, but the score (AUC) on the public leaderboard (calculated on approximately 52% of the data) on Kaggle was only 0.70536, discouraging any further tuning of the model. However, the final private leaderboard score (AUC) turned out as 0.92821 (topping the list). There is a huge discrepancy between the scores. My firm belief is that tuning this model a bit further would probably be beneficial for a more real-life application.

8 References

[1] MLSP 2014 Schizophrenia classification challenge, 2014. URL `https://www.kaggle.com/c/mlsp-2014-mri`.

[2] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[3] J. Vanhatalo, J. Riihimäki, J. Hartikainen, P. Jylänki, V. Tolvanen, and A. Vehtari. GPstuff: Bayesian modeling with Gaussian processes. *The Journal of Machine Learning Research*, 14 (1):1175–1179, 2013.

[4] M. Cetin, F. Christensen, C. Abbott, J. Stephen, A. Mayer, J. Canive, J. Bustillo, G. Pearlson, and V. D. Calhoun. Thalamus and posterior temporal lobe show greater inter-network connectivity at rest and across varying sensory loads in schizophrenia. *NeuroImage*, in press.

[5] E. A. Allen, E. Damaraju, S. M. Plis, E. B. Erhardt, T. Eichele, and V. D. Calhoun. Tracking whole-brain connectivity dynamics in the resting state. *Cerebral Cortex*, pages 663–676, 2012.

[6] J. M. Segall, E. A. Allen, R. E. Jung, E. B. Erhardt, S. K. Arja, K. A. Kiehl, and V. D. Calhoun. Correspondence between structure and function in the human brain at rest. *Frontiers in Neuroinformatics*, 6(10), 2012.

[7] S. Särkkä, A. Solin, A. Nummenmaa, A. Vehtari, T. Auranen, S. Vanni, and F.-H. Lin. Dynamical retrospective filtering of physiological noise in BOLD fMRI: DRIFTER. *NeuroImage*, 60 (2):1517–1527, 2012.

[8] S. Särkkä, A. Solin, and J. Hartikainen. Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing. *IEEE Signal Processing Magazine*, 30(4):51–61, 2013.

[9] A. Solin and S. Särkkä. Infinite-dimensional Bayesian filtering for detection of quasiperiodic phenomena in spatiotemporal data. *Physical Review E*, 88:052909, 2013.

[10] I. Murray, R. P. Adams, and D. Mackay. Elliptical slice sampling. In *International Conference on Artificial Intelligence and Statistics*, pages 541–548, 2010.

[11] I. Murray and R. P. Adams. Slice sampling covariance hyperparameters of latent Gaussian models. In *Advances in Neural Information Processing Systems*, pages 1732–1740, 2010.

[12] J. Vanhatalo, J. Riihimäki, J. Hartikainen, P. Jylänki, V. Tolvanen, and A. Vehtari. Bayesian modeling with Gaussian processes using the GPstuff toolbox. *arXiv preprint arXiv:1206.5754*, 2012.

[13] A. Vehtari (corresponding author). GPstuff – Gaussian process models for Bayesian analysis. URL `http://becs.aalto.fi/en/research/bayes/gpstuff/`. Software toolbox for Mathworks Matlab.