Laplace Inference for Multi-fidelity Gaussian Process Classification

Nikita Klyuchnikov and Evgeny Burnaev
Skolkovo Institute of Science and Technology
Moscow, Russia
nikita.klyuchnikov@skolkovotech.ru
e.burnaev@skoltech.ru

Abstract

In this paper we address a classification problem where two sources of labels with different levels of fidelity are available. Our approach is to combine data from both sources by applying a co-kriging schema on latent functions, which allows the model to account item-dependent labeling discrepancy. We provide an extension for Laplace inference for Gaussian process classification, that takes into account multi-fidelity data. We evaluate the proposed method on real and synthetic datasets and show that it is more resistant to different levels of discrepancy between sources, than other approaches for data fusion. Our method can provide accuracy/cost trade-off for a number of practical tasks such as crowd-sourced data annotation and feasibility regions construction in engineering design.

1 Introduction

The problem of multi-fidelity modeling [13] arises in the broad range of applied disciplines, such as engineering design, medical diagnostics, and even product development, when an object of interest can be approximated with a cheaper, yet typically less reliable alternative. The main motivation behind multi-fidelity modeling is that low-fidelity data can bring additional benefits in terms of accuracy/cost trade-off, when it is used properly along with high-fidelity data [23][22]. For example, article [21] demonstrates that high-quality linguistic annotation results can be achieved with much lower expenses when non-expert annotators i.e. low-fidelity data are employed. The authors concluded that four non-experts per item were enough on average to achieve an expert-level annotation quality for their tasks, although this requirement can be relaxed further, by requiring multiple annotations only for a fraction of the dataset. Similarly in engineering design [12], a high-fidelity source of data can be a physical experiment, whereas a low-fidelity can be a mathematical model or a computer simulation.

Multi-fidelity modeling based on Gaussian processes (GPs) [15] is a reasonable approach for applications discussed above, because of the Bayesian formulation, which allows incorporation of the prior knowledge about the task into the prediction and makes learning on small samples more robust. The latter is especially important, since high-fidelity data typically contains just a few examples. In addition, Gaussian processes are based on kernel functions, whose hyperparameters can be selected via marginal likelihood maximization instead of grid search with cross-validation.

Gaussian process regression for multi-fidelity data has been thoroughly studied in recent years [5][25], however multi-fidelity classification based on Gaussian processes has been left behind.
Moreover, as C. Dribusch points out in his thesis about feasibility regions for aerelastic stability modeling [4], multi-fidelity methods have been limited to continuous response models. Although discrete response models can also be approximated with continuous ones, in some extreme cases, such as binary classification, continuous approximations seem as weird as using Linear regression instead of Logistic regression. There are problems in engineering design with discrete responses, for instance, report [7] points out the problem of reality gap in robotic simulators and argues the importance of their ability to estimate reliable regions, where accomplishment of actions is accurately predicted by the simulator. This problem has binary responses i.e. success or fail; simulated outcomes of robot’s actions are low-fidelity data, whereas observations of real execution are high-fidelity data. Discrete responses are even more common and convenient when the object of interest is a human. For example, users say they either like a new feature of the product or not during A/B testing, which gives a direct evidence of their attitude i.e. high-fidelity data, or users are just asked to imagine the feature and express their preferences during interviewing i.e. low-fidelity data.

In this work, we propose a co-kriging model for latent low- and high- fidelity functions and extend the Laplace inference algorithm for Gaussian process classification to handle this case. We evaluate the proposed method on three groups of datasets: artificially generated under the model assumptions, real benchmark datasets with simulated noise for low-fidelity labels and real datasets with true noise.

2 Related work

A comprehensive introduction into GPs in the context of machine learning has been done in the book [15]. We were guided by that book during derivations of our algorithm. More detailed study of methods for approximate binary classification inference based on GPs is in the paper [10]. The paper demonstrates that Laplace approximation is the fastest inference method with moderate accuracy, whereas Expectation Propagation is the most accurate, but runs approximately 10 times slower.

Supervised classification in the presence of noise in labels has been studied in the paper [9], where authors consider class-conditional random Bernoulli noise and provide theoretical justification of learnability of such classification problems.

Multi-fidelity regression based on GPs is described in the paper [5]. Authors use a co-kriging setup for fidelities and provide an exact inference schema for their regression. In our work we adopt co-kriging for the classification problem by applying this setup on latent functions. Note that there is no exact inference schema for GP classification for single-fidelity case, nor for multi-fidelity one.

There is a large branch of research on learning from multiple annotators [27], which partially intersects with the applications of our method. Early works in this direction started with different strategies of feature-agnostic labels integration and active learning for optimizing annotation costs [20, 24, 8, 3]. Paper [23] proposed a generative probabilistic model to estimate annotators expertise along with items annotation difficulty, yet features are not observable for the model. Paper [5] studies the problem of pruning low-quality annotators in order to improve the quality of the training set for binary classification problem. The same authors also build an algorithm on top of the SVM, that decreases influence of low-quality entries [2]. Several state-of-the-art works [16, 17, 18] model annotations as random Bernoulli labels dependent on the true class, which in turn is generated via latent Gaussian process; these works have similar setups and provide Variational Bayes and Expectation Propagation inferences for them. Overall, all these works deal with cases when many annotators are available, since otherwise their expertise (fidelity in our
where \( \rho \) is an indicator function; \( \in \) is an integration symbol; \( \mathbb{I} \) is an indicator function; \( \mathbb{R} \) is the set of real numbers. Using the Bayesian approach we formally express this assumption with the following model:

\[
\begin{align*}
\theta & = \mathbb{I}[f_H(x) > 0], \\
p(y_H^i = 1|f_H(x_i^H)) = \sigma(f_H(x_i^H)) \quad \text{and} \quad p(y_L^i = 1|f_L(x_i^L)) = \sigma(f_L(x_i^L)),
\end{align*}
\]

\( (1) \)

where \( I \) is an indicator function; \( \sigma(z) = \frac{1}{1 + \exp(-z)} \) is a logistic; \( f_L \) and \( f_H \) are Gaussian processes on \( \Omega \). In our model we assume \( \sigma \) processes have the dependency called co-kriging \( [5] \):

\[
f_H(x_i^H) = \rho f_L(x_i^L) + \delta(x_i^H),
\]

\( (2) \)

where \( \rho \in \mathbb{R} \) is a linear coefficient and \( \delta \) is a residual Gaussian process independent of \( f_L \). Processes \( f_L \) and \( \delta \) have prior kernels \( k_l \) and \( k_d \) with hyper-parameters \( \theta_l \) and \( \theta_d \) respectively.

### 3 Problem statement

There is a binary function \( c: \Omega \to \{0, 1\} \) defined on the measurable set \( \Omega \subseteq \mathbb{R}^d \). The goal is to train a classifier \( \hat{c} \) that estimates \( c \) using the information from the two samples:

\[
D_H = \{(x_1^H, y_1^H)\}_{i=1}^{n_H} \quad \text{and} \quad D_L = \{(x_1^L, y_1^L)\}_{i=1}^{n_L}
\]

where \( x_i^H, x_i^L \in \Omega \) and \( y_i^L, y_i^H \in \{0, 1\} \). Let us also denote \( X_L = \{x_i^L\}_{i=1}^{n_L} \) and \( X_H = \{x_i^H\}_{i=1}^{n_H} \).

Sample \( D_H \) contains high-fidelity data, that is, it has much more reliable labels than \( D_L \), which contains low-fidelity data respectively. Using the Bayesian approach we formally express this assumption with the following model:

\[
c(x) = \mathbb{I}[f_H(x) > 0], \\
p(y_H^i = 1|f_H(x_i^H)) = \sigma(f_H(x_i^H)) \quad \text{and} \quad p(y_L^i = 1|f_L(x_i^L)) = \sigma(f_L(x_i^L)),
\]

\( (1) \)

where \( I \) is an indicator function; \( \sigma(z) = \frac{1}{1 + \exp(-z)} \) is a logistic; \( f_L \) and \( f_H \) are Gaussian processes on \( \Omega \). In our model we assume \( \sigma \) processes have the dependency called co-kriging \( [5] \):

\[
f_H(x_i^H) = \rho f_L(x_i^L) + \delta(x_i^H),
\]

\( (2) \)

where \( \rho \in \mathbb{R} \) is a linear coefficient and \( \delta \) is a residual Gaussian process independent of \( f_L \). Processes \( f_L \) and \( \delta \) have prior kernels \( k_l \) and \( k_d \) with hyper-parameters \( \theta_l \) and \( \theta_d \) respectively.

### 4 Solution

For simplicity of notation we omit specifying hyper-parameters \( (\rho \) and parameters of kernels \( \theta_l, \theta_d) \) as conditions of probabilities in formulas below.

The predictive distribution of \( f_H \) at \( x_* \in \Omega \) is:

\[
p(f_*^H|D_L, D_H, x_*) = \int \int p(f_*^H|f^L, \delta, X_L, X_H, x_*)p(f^L, \delta|D_L, D_H)d\mathcal{F}^Ld\delta,
\]

\( (3) \)

where \( f^L = (f^L(x_1^L), ..., f^L(x_{n_L}^L), f^L(x_1^H), ..., f^L(x_{n_H}^H))^T \); \( \delta = (\delta(x_1^H), ..., \delta(x_{n_H}^H))^T \).

The probability of \( c \) to be 1 at point \( x_* \) can be expressed by marginalization of the predictive distribution:

\[
p(c(x_*) = 1|D_L, D_H, x_*) = \int \sigma(f_*^H)p(f_*^H|D_L, D_H, x_*)df_*^H.
\]

\( (4) \)

Integrals \( (3) \) and \( (4) \) don’t have analytic solutions, therefore they have to be numerically integrated or approximated analytically. In this work we use Laplace approximation method to handle the former, whereas the predicted class based on the latter integral can be easily calculated in binary case once the predictive distribution is known or estimated \( [1] \):

\[
\hat{c}(x_*) = \mathbb{I}\left[ \int \sigma(f_*^H)p(f_*^H|D_L, D_H, x_*)df_*^H > \frac{1}{2} \right] = \mathbb{I}\left[ \int f_*^Hp(f_*^H|D_L, D_H, x_*)df_*^H > 0 \right].
\]

3
4.1 Laplace approximation

The derivation of Laplace approximation for multi-fidelity case is based on that for single-fidelity, provided in section 3.4 of book [15]. We will adjust our solution to fit the algorithmic framework developed in the book, in particular, algorithms 3.1, 3.2, and 5.1.

The posterior distribution in integral (3) is approximated with Gaussian distribution \( q(\cdot) \):

\[
p(t^L, \delta | D_L, D_H) \approx q(t^L, \delta | D_L, D_H) = \mathcal{N}\left( \xi = \begin{bmatrix} t^L \\ \delta \end{bmatrix}, \Sigma^{-1} \right),
\]

where \( \xi = \arg\max_{\xi} p(\xi | D_L, D_H) \) and \( \Sigma = -\nabla \nabla \log p(\xi | D_L, D_H) |_{\xi = \xi} \). Thus, for obtaining approximate posterior distribution we need to calculate these parameters.

According to Bayes formula and monotonic increase of log function, the problem of finding \( \hat{\xi} \) is equivalent to:

\[
\arg\max_{\xi} p(\xi | D_L, D_H) = \arg\max_{\xi} \left[ \log p(y^L, y^H | \xi) + \log p(\xi | X_L, X_H) \right],
\]

where \( y^L = (y^L_1, ..., y^L_{n_L})^T \) and \( y^H = (y^H_1, ..., y^H_{n_H})^T \). Note that the probability of evidence is omitted, since it is independent of the argument.

By analogy with the book we define \( \Psi(\xi) \overset{\Delta}{=} \log p(y^L, y^H | \xi) + \log p(\xi | X_L, X_H) \). Now let’s look at its components in more detail.

Let us denote \( X = X_L \cup X_H \). The prior distribution of \( \xi \) is normal:

\[
p(\xi | X_L, X_H) \sim \mathcal{N}\left( 0, K = \begin{bmatrix} k_l(X, X) & 0 \\ 0 & k_d(X_H, X_H) \end{bmatrix} \right).
\]

Log-likelihood is:

\[
\lambda \overset{\Delta}{=} \log p(y^L, y^H | \xi) = \sum_{i=1..n_l} \log \sigma(\hat{y}^L_i f^L(x_i^L)) + \sum_{i=1..n_h} \log \sigma(\hat{y}^H_i (\rho f^L(x_i^H) + \delta_i)),
\]

where for simplicity of notation \( \delta_i = \delta(x_i^H) \), \( \hat{y}^L_i = (2y^L_i - 1) \), and \( \hat{y}^H_i = (2y^H_i - 1) \).

4.2 Correctness of the method

Optimization problem \( [6] \) has a unique solution if \( \Psi \) is concave. We prove it by showing that Hessian of \( \Psi \) is negative semi-definite. The Hessian is

\[
\nabla \nabla \Psi(\xi) = -W - K^{-1},
\]

where \( W \overset{\Delta}{=} -\nabla \nabla \log p(y^L, y^H | \xi) \) and \( K \) is defined in (7). The latter is positive semi-definite since it is a kernel matrix. \( W \) has the following structure:

\[
W = \begin{bmatrix}
A & 0 & 0 \\
0 & \rho^2 D & \rho D \\
0 & \rho D & D
\end{bmatrix}
\]

with \( A = \nabla \nabla f^L(x_L) \lambda = \text{diag} \left( \omega \left( f^L(x_i^L) \right)_{i=1..n_l} \right) \),

\[
D = \nabla \nabla \delta \lambda = \text{diag} \left( \omega \left( (\rho f^L(x_i^H) + \delta_i) \right)_{i=1..n_h} \right),
\]

where \( \omega(z) = \sigma'(z) = \sigma(z)(1 - \sigma(z)) \).

Matrices \( A \) and \( D \) are positive semi-definite because their diagonal elements are non-negative. The block of \( W \) that contains \( D \) can be represented via Kronecker product:

\[
\begin{bmatrix}
\rho^2 D & \rho D \\
\rho D & D
\end{bmatrix} = \begin{bmatrix}
\rho^2 & \rho \\
\rho & 1
\end{bmatrix} \otimes D.
\]
Both multiplicands in (9) are positive semi-definite, thus, their Kronecker product is also positive semi-definite [19]. Hence, matrix $W$ is positive semi-definite, because it factorizes into two positive semi-definite blocks. Finally, the Hessian is negative semi-definite as a negation of sum of two positive semi-definite matrices.

4.3 Model selection

Model selection implies finding hyper-parameters $\rho$, $\theta_l$, and $\theta_d$ that maximize the approximate marginal log-likelihood [15]:

$$L \triangleq \log q(y^L, y^H | X_L, X_H, \rho, \theta_l, \theta_d) = -\frac{1}{2} \xi^T K^{-1} \xi + \lambda - \frac{1}{2} \log |B|,$$

where $B = I + W^{\frac{1}{2}} K W^{\frac{1}{2}}$.

Unlike single-fidelity case, $W$ in multi-fidelity case is non-diagonal, so computation of its square root is not straightforward. We have derived the exact formula for its fast and numerically stable calculation:

$$W^{\frac{1}{2}} = \begin{bmatrix} A^{\frac{1}{2}} & 0 \\ 0 & \frac{1}{\sqrt{\rho^2 + 1}} \begin{bmatrix} 0 & \rho \\ \rho & 1 \end{bmatrix} \otimes D^{\frac{1}{2}} \end{bmatrix},$$

note that matrices $A$ and $D$ are diagonal, so their square roots are easily calculated.

The algorithm 5.1 from [15] that optimizes (10) requires its partial derivatives w.r.t. hyper-parameters.

Derivatives of $L$ and $\hat{\xi}$ w.r.t. kernel hyper-parameters $\theta_l$ and $\theta_d$ are almost identical to formulas provided in the book, section 5.5.1 [15], except the final step in formula 5.23, that doesn’t take place for multi-fidelity case, since $W$ is not diagonal. Instead, we propose the following modification of that formula:

$$\frac{\partial L}{\partial \hat{\xi}_i} = -\frac{1}{2} \text{tr} \left( (K^{-1} + W)^{-1} \frac{\partial W}{\partial \hat{\xi}_i} \right) = -\frac{1}{2} \sum_{\text{all elements}} \left( (K^{-1} + W)^{-1} \circ \frac{\partial W}{\partial \hat{\xi}_i} \right),$$

where $\circ$ is an Hadamard (entrywise) product. Note that $\frac{\partial W}{\partial \hat{\xi}_i}$ is a sparse matrix that has at most 4 non-zero elements, therefore computation time of the derivatives remains linear.

Derivative of $L$ w.r.t to $\rho$ is:

$$\frac{\partial \xi}{\partial \rho} = -\xi^T K^{-1} \frac{\partial \xi}{\partial \rho} + \frac{\partial \lambda}{\partial \rho} - \frac{1}{2} \frac{\partial \log |B|}{\partial \rho}.$$  

Note that in our setup $K$ doesn’t depend on $\rho$. Now let’s look into components of (13) in more detail. By analogue with formula 5.24 from the book [15], we differentiate by $\rho$ a necessary condition of the maximum $\nabla \Psi(\xi)|_{\xi=\hat{\xi}} = 0$, where $\nabla \Psi(\xi) = \nabla \xi \lambda - K^{-1} \xi$, obtaining an equation on $\xi$:

$$\frac{\partial \hat{\xi}}{\partial \rho} = K \left( -W \frac{\partial \hat{\xi}}{\partial \rho} + \frac{\partial \nabla \xi \lambda}{\partial \rho} |_{\xi=\hat{\xi}} \right) \Rightarrow \frac{\partial \hat{\xi}}{\partial \rho} = (I + KW)^{-1} K \left( \frac{\partial \nabla \xi \lambda}{\partial \rho} |_{\xi=\hat{\xi}} \right),$$

where the components of the explicit term in formula (14) and derivatives of $\lambda$ w.r.t. components of $\xi$ are provided in Table [1].
Table 1: Components of $\xi$, corresponding derivatives of $\lambda$ and the explicit term in [14]; here $f_i^L = f^L(x_i^L)$ and $f_i^H = \rho f^L(x_i^H) + \delta(x_i^H)$.

| components of $\xi$ | components of $\nabla \xi \lambda$ | components of $\frac{\partial \nabla \xi \lambda}{\partial \rho}$ | explicit |
|----------------------|-----------------------------------|----------------------------------|---------|
| $f^L(X_L)$           | $y^L_i - \sigma(f^L_i)$          | $0$                              |         |
| $f^L(X_H)$           | $\rho(y^H_i - \sigma(f^H_i))$   | $y^H_i - \sigma(f^H_i) - \rho f^L(x^H_i)\omega(f^H_i)$ |         |
| $\delta(X_H)$       | $\tilde{y}^H_i - \sigma(f^H_i)$ | $-f^L(x^H_i)\omega(f^H_i)$       |         |

Next component of (13) is:

$$\frac{\partial \lambda}{\partial \rho} = \sum_{i=1..n_\lambda} \tilde{y}^H_i f^L(x^H_i) (1 - \sigma(\tilde{y}^H_i (\rho f^L(x^H_i) + \delta(x^H_i)))) + \sum \frac{\partial \lambda}{\partial \xi_i} \frac{\partial \xi_i}{\partial \rho}. \quad (15)$$

The last component is:

$$\frac{\partial \log |B|}{\partial \rho} = \sum_{\text{all elements}} \left((K^{-1} + W)^{-1} \frac{\partial W}{\partial \rho}\right), \quad (16)$$

where $\frac{\partial W}{\partial \rho} = \begin{bmatrix} 0 & 0 \\ 0 & \rho^2 \rho \otimes \frac{\partial D}{\partial \rho} \bigg|_{\text{explicit}}^{\text{explicit}} + \begin{bmatrix} 2\rho & 1 \\ 1 & 0 \end{bmatrix} \otimes D + \sum \frac{\partial W}{\partial \omega} \frac{\partial \omega}{\partial \rho};$ 

$$\frac{\partial D}{\partial \rho} \bigg|_{\text{explicit}} = \text{diag}\left(f^L(x^H_i)\zeta(f^H_i)\right|_{i=1..n_\lambda}) \quad \text{and} \quad \zeta(x) = \sigma''(x) = \sigma(x)(1-\sigma(x))(1-2\sigma(x)).$$

4.4 Prediction

Once we know estimates of parameters and hyper-parameters, we can use an ordinary schema of exact multi-fidelity posterior from [5] to obtain MAP predictions:

$$E[f_*|D_L, D_H, x_*] \approx \mathbb{E}_q[f_*|D_L, D_H, x_*] = \tilde{k}_*^T \tilde{K}^{-1} \tilde{f}, \quad (17)$$

where $\tilde{k}_*^T = \begin{bmatrix} k_l(x_*, X_L) & \rho k_l(x_*, X_H) + k_d(x_*, X_H) \end{bmatrix}; \quad \tilde{\xi} = \begin{bmatrix} f^L(X_L) & f^L(X_H) & \delta(X_H) \end{bmatrix}^T; \quad \tilde{K} = \begin{bmatrix} k_l(X_L, X_L) & \rho k_l(X_L, X_H) \\ \rho k_l(X_H, X_L) & \rho^2 k_l(X_H, X_H) + k_d(X_H, X_H) \end{bmatrix}; \quad \text{and} \quad \tilde{f} = \begin{bmatrix} f^L(X_L) \\ f^L(X_H) \end{bmatrix}.$

5 Experiments

We compared our model with a number of baseline approaches. The baselines are built upon ordinary Gaussian Process Classifier (gpc), Logistic Regression (logit) and Gradient Boosting Classifier (xgb). We trained those baselines in three modes: 1) training only on high-fidelity data (no prefix), 2) training on concatenated high- and low- fidelity data (with prefix C), 3) stacking low-fidelity predictions, that is, predictions of a classifier trained on low-fidelity data were used as additional features for training the classifier on high-fidelity data (with prefix S). All GPs-based methods used isotropic RBF kernel.

We evaluated our model on three groups of datasets:

1. Artificial datasets [1] we constructed datasets by virtue of the model [1] and [2]. Latent functions $f_L$ and $\delta$ were generated as instances of Gaussian processes, linear coefficients $\rho$ were adjusted to the desired discrepancy (noise level) between low- and high- fidelities. We used input dimensions 2, 5, 10, and 20. For each of them we generated 10 datasets.

\[\text{We published them in this repository } \text{https://github.com/user525/mfgpc}\]
Figure 1: Comparison of predicted class probabilities with multi-fidelity MCMC and Laplace inference on datasets from group 2: typical cases of correlations.

|        | dbts gpc Laplace | grmn gpc Laplace | stmg gpc MCMC | splc gpc MCMC | spmb gpc MCMC | hpth gpc MCMC | wvfr gpc MCMC |
|--------|------------------|------------------|--------------|--------------|--------------|--------------|--------------|
| MF gpc Laplace | 0.815 0.787 0.998 0.999 0.940 0.942 0.614 0.932 | | | | | | |
| MF gpc MCMC    | 0.809 0.780 0.997 0.999 0.922 0.946 0.624 0.927 | | | | | | |

Table 2: Comparison of ROC AUC in a single run for MCMC and Laplace inference on datasets from group 2 during verification tests.

2. Datasets from Penn Machine Learning Benchmarks repository [11]: we selected several representative benchmarks with different types of features, namely diabetes (dbts), german (grmn), waveform-40 (wvfr), satimage (stmg), splice (splc), spambase (spmb), hypothyroid (hpth), and mushroom (mshr). Since some datasets had multiple classes, we also selected one target representative class to test its classification against others: class 0 for waveform-40 and splice, class 1 for satimage and class 2 for diabetes. Low-fidelity labels were generated by flipping original labels with the specified probability (noise level).

3. Real datasets: we used music_genre (mscg) and sentiment_polarity (sntp) from [17], which had been annotated with crowd-sourcing. Each object in those datasets was labeled by multiple annotators, therefore we considered major voting statistic over object labels as high-fidelity and a single random annotation as low-fidelity. Since music_genre dataset had multiple classes, we tested each of them with one-vs-all scheme as separate datasets.

For datasets on crowd-sourcing annotation (group 3) we also compared our method with the state-of-the-art method gp-ma [17]. No comparison was made with [18], since we couldn’t find publicly available source code.

At the outset, we verified our implementation of Laplace inference by comparing its predictions with those of MCMC (with the same hyper-parameters) on real datasets from group 2 ensuring that true posteriors are non-Gaussian. Each training set contained 75 randomly sampled high fidelity observations and flip probability 0.2 in low-fidelity observations. The typical results of comparison are shown in Figure 1 and Table 2. The overall performance of two inference approaches is on par, whereas correlation behaviour resembles patterns observed in single-fidelity GPs classification [10], figure 6), which lends evidence supporting correctness of our method.

The main evaluation procedure was the following: for a single test we selected a small random subsample of high fidelity observations and 3 times larger subsample of low fidelity observations. We trained all methods on those subsamples and evaluated predictions on the high-fidelity test set. For each dataset we run 10 tests with different random subsamples, except sentiment_polarity, for which we run 50 tests.
In order to aggregate performance results across many tests and datasets, we report average ROC AUC across all tests and methods in tables 3, 4, 5 and 6. For those tests each training set contained 75 high fidelity observations. Methods that performed not worse than 1 percent compared to the best result on the dataset are highlighted with bold.

We also supplement results in tables with ROC AUC profiles, which show the share of tests where the corresponding methods had greater ROC AUC than the threshold pointed on the abscissa axis, in figures 2 and 3, the higher the curve, the better the corresponding method. Overall, MF gpc has a good performance, except sntp dataset. Notably, on this dataset all GPs-based methods have poor performance, which is not surprising, since we used a translation-invariant isotropic kernel, which is not suited well for highly clustered non-stationary data.

Finally, we studied how ratio of low- and high-fidelity samples sizes affects the classification quality. The total sample size contained 300 observations, whereas only part of them were of high-fidelity. Figure 4 shows the results on artificial datasets from group 1 and a real mscg dataset.

### Table 3: Average ROC AUC among multiple runs on artificial datasets from group 1.

| Noise level | Dimensionality | 2D | 5D | 10D | 20D | 2D | 5D | 10D | 20D |
|-------------|----------------|----|----|-----|-----|----|----|-----|-----|
| MF gpc      | 0.975          | 0.847 | 0.716 | 0.650 | 0.970 | 0.731 | 0.619 | 0.575 |
| gpc         | 0.971          | 0.732 | 0.605 | 0.582 | 0.970 | 0.725 | 0.613 | 0.578 |
| logit       | 0.743          | 0.599 | 0.561 | 0.563 | 0.743 | 0.600 | 0.557 | 0.565 |
| xgb         | 0.912          | 0.660 | 0.594 | 0.566 | 0.915 | 0.659 | 0.592 | 0.567 |
| C gpc       | 0.944          | 0.847 | 0.717 | 0.668 | 0.803 | 0.662 | 0.583 | 0.567 |
| C logit     | 0.719          | 0.614 | 0.576 | 0.593 | 0.665 | 0.577 | 0.552 | 0.550 |
| C xgb       | 0.922          | 0.727 | 0.650 | 0.626 | 0.806 | 0.622 | 0.583 | 0.567 |
| S gpc       | 0.954          | 0.800 | 0.677 | 0.618 | 0.943 | 0.716 | 0.611 | 0.588 |
| S logit     | 0.744          | 0.596 | 0.558 | 0.562 | 0.745 | 0.596 | 0.563 | 0.556 |
| S xgb       | 0.923          | 0.689 | 0.604 | 0.578 | 0.916 | 0.653 | 0.591 | 0.568 |

### Table 4: Average ROC AUC among multiple runs on datasets from group 2 with noise level 0.2.

|                | dbts | grmn | stmg | mshr | splc | spmb | hpth | wvfr |
|----------------|-----|------|------|------|------|------|------|------|
| MF gpc         | 0.827 | 0.752 | 0.997 | 0.997 | 0.937 | 0.934 | 0.680 | 0.918 |
| gpc            | 0.798 | 0.695 | 0.995 | 0.992 | 0.922 | 0.917 | 0.640 | 0.904 |
| logit          | 0.797 | 0.667 | 0.997 | 0.996 | 0.926 | 0.915 | 0.772 | 0.859 |
| xgb            | 0.773 | 0.690 | 0.976 | 0.991 | 0.976 | 0.936 | 0.857 | 0.889 |
| C gpc          | 0.824 | 0.740 | 0.996 | 0.995 | 0.935 | 0.925 | 0.578 | 0.910 |
| C logit        | 0.819 | 0.702 | 0.986 | 0.963 | 0.809 | 0.867 | 0.758 | 0.821 |
| C xgb          | 0.812 | 0.747 | 0.988 | 0.989 | 0.967 | 0.943 | 0.860 | 0.890 |
| S gpc          | 0.818 | 0.717 | 0.996 | 0.996 | 0.933 | 0.919 | 0.571 | 0.916 |
| S logit        | 0.810 | 0.658 | 0.996 | 0.994 | 0.927 | 0.920 | 0.747 | 0.847 |
| S xgb          | 0.783 | 0.714 | 0.971 | 0.987 | 0.974 | 0.932 | 0.800 | 0.900 |
Table 5: Average ROC AUC among multiple runs on datasets from group 2 with noise level 0.4.

| Method  | dbts | grmn | stmg | mshr | spc  | spmb | hpth | wvfr |
|---------|------|------|------|------|------|------|------|------|
| MF gpc  | 0.807| 0.695| 0.996| 0.997| 0.888| 0.923| 0.613| 0.912|
| gpc     | 0.802| 0.698| 0.995| 0.994| 0.920| 0.920| 0.626| 0.899|
| logit   | 0.798| 0.664| 0.998| 0.992| 0.934| 0.910| 0.801| 0.847|
| xgb     | 0.764| 0.663| 0.951| 0.992| 0.982| 0.926| 0.824| 0.873|
| C gpc   | 0.772| 0.643| 0.972| 0.969| 0.817| 0.869| 0.539| 0.866|
| C logit | 0.760| 0.635| 0.957| 0.830| 0.635| 0.754| 0.651| 0.727|
| C xgb   | 0.668| 0.627| 0.902| 0.943| 0.817| 0.857| 0.747| 0.765|
| S gpc   | 0.804| 0.700| 0.994| 0.995| 0.880| 0.909| 0.645| 0.904|
| S logit | 0.784| 0.663| 0.996| 0.993| 0.930| 0.907| 0.788| 0.844|
| S xgb   | 0.772| 0.666| 0.961| 0.990| 0.977| 0.930| 0.870| 0.898|

Table 6: Average ROC AUC among multiple runs on datasets from group 3 with natural noise.

| Method  | dbts | grmn | stmg | mshr | spc  | spmb | hpth | wvfr |
|---------|------|------|------|------|------|------|------|------|
| gpc     | 0.886| 0.781| 0.797| 0.775| 0.875| 0.850| 0.866| 0.795|
| logit   | 0.784| 0.663| 0.996| 0.993| 0.930| 0.907| 0.788| 0.844|
| xgb     | 0.772| 0.666| 0.961| 0.990| 0.977| 0.930| 0.870| 0.898|
| gp-ma   | 0.781| 0.797| 0.775| 0.875| 0.850| 0.866| 0.795| 0.796|

Figure 2: ROC AUC profiles for artificial datasets from group 1.

6 Conclusions

Multi-fidelity modeling of discrete response surfaces can be put to good use in a number of applied disciplines, yet such methods have got little attention so far. In this work we extended Laplace inference algorithm for classification based on GPs to make it work with multi-fidelity data. Despite the algorithm was proposed for two levels of fidelities, it can be trivially generalized to arbitrary number of levels. By modeling latent GPs dependency with a co-kriging schema, which has been used previously for multi-fidelity regression, our method can identify not only the overall relevance of low-fidelity data, but resolve local item-dependent discrepancies between fidelities due to inference on residual Gaussian process \( \delta \).

We evaluated our method on multiple artificial and real datasets with natural and various levels of simulated noise and compared its performance with a number of baseline approaches and a state-of-the-art method. We also demonstrated in our experiments (see Figure 4) that adding...
Figure 3: ROC AUC profiles for real datasets from groups 2 and 3. Colors represent the same legend as in figure 2.

Figure 4: MF gpc and gpc performance w.r.t. share of high-fidelity observations in the sample; gpc was trained only on high-fidelity part of sample. In figure (4a) grey shades represent different levels of noise in low-fidelity; in figure (4b) shaded area shows 20 and 80 percentiles over multiple runs, natural indicates that MF gpc was trained on real data.

noisy low-fidelity to the training set increases quality of high-fidelity data classification, even when the high-fidelity training set is small. Depending on the dataset nature, MF gpc can alternate its performance with respect to other methods, however, it is more resistant to different noise levels in low-fidelity labels. That is, when the classifiers based on GPs can learn datasets well, MF gpc has a top performance, whereas in other cases our method is on par with the considered methods.
References

[1] "Neural networks for pattern recognition", author=Bishop, Christopher M, year=1995, publisher=Oxford university press.

[2] O. Dekel and O. Shamir. Good learners for evil teachers. In Proceedings of the 26th annual international conference on machine learning, pages 233–240. ACM, 2009.

[3] O. Dekel and O. Shamir. Vox populi: Collecting high-quality labels from a crowd. 2009.

[4] C. Dribusch. Multi-fidelity construction of explicit boundaries: Application to aerelasticity. The University of Arizona, 2013.

[5] A. I. Forrester, A. Sóbester, and A. J. Keane. Multi-fidelity optimization via surrogate modelling. In Proceedings of the royal society of london a: mathematical, physical and engineering sciences, volume 463, pages 3251–3269. The Royal Society, 2007.

[6] P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In Proceedings of the ACM SIGKDD workshop on human computation, pages 64–67. ACM, 2010.

[7] J.-B. Mouret and K. Chatzilygeroudis. 20 years of reality gap: a few thoughts about simulators in evolutionary robotics. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, pages 1121–1124. ACM, 2017.

[8] B. Mozafari, P. Sarkar, M. J. Franklin, M. I. Jordan, and S. Madden. Active learning for crowd-sourced databases. CoRR, abs/1209.3686, 2012.

[9] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari. Learning with noisy labels. In Advances in neural information processing systems, pages 1196–1204, 2013.

[10] H. Nickisch and C. E. Rasmussen. Approximations for binary Gaussian process classification. Journal of Machine Learning Research, 9(Oct):2035–2078, 2008.

[11] R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore. PMLB: a large benchmark suite for machine learning evaluation and comparison. BioData Mining, 10(1):36, Dec 2017.

[12] W. Pawlus, K. G. Robbersmyr, and H. R. Karimi. Mathematical modeling and parameters estimation of a car crash using data-based regressive model approach. Applied Mathematical Modelling, 35(10):5091 – 5107, 2011.

[13] B. Peherstorfer, K. Willcox, and M. Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. Preprint, pages 1–57, 2016.

[14] N. D. Penna and M. D. Reid. Crowd & prejudice: An impossibility theorem for crowd labelling without a gold standard, 2012.

[15] C. E. Rasmussen and C. K. Williams. Gaussian processes for machine learning. MIT Press, 2006.

[16] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. Journal of Machine Learning Research, 11(Apr):1297–1322, 2010.
[17] F. Rodrigues, F. Pereira, and B. Ribeiro. Gaussian process classification and active learning with multiple annotators. In *International Conference on Machine Learning*, pages 433–441, 2014.

[18] P. Ruiz, E. Besler, R. Molina, and A. K. Katsaggelos. Variational Gaussian process for missing label crowdsourcing classification problems. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE, 2016.

[19] K. Schäcke. On the Kronecker Product. 2013.

[20] V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008.

[21] R. Snow, B. O’Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics, 2008.

[22] D. J. Toal. Some considerations regarding the use of multi-fidelity kriging in the construction of surrogate models. *Struct. Multidiscip. Optim.*, 51(6):1223–1245, June 2015.

[23] P. Welinder, S. Branson, P. Perona, and S. J. Belongie. The multidimensional wisdom of crowds. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2424–2432. Curran Associates, Inc., 2010.

[24] J. Whitehill, T.-f. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043, 2009.

[25] A. Zaytsev and E. Burnaev. Large scale variable fidelity surrogate modeling. *Annals of Mathematics and Artificial Intelligence*, Apr 2017.

[26] A. Zaytsev and E. Burnaev. Minimax approach to variable fidelity data interpolation. In *AISTATS*, pages 652–661, 2017.

[27] J. Zhang, X. Wu, and V. S. Sheng. Learning from crowdsourced labeled data: a survey. *Artificial Intelligence Review*, 46(4):543–576, 2016.
## A Appendix

This appendix contains detailed information about some key identities and experimental setup.

| Notations | Descriptions | Specification |
|-----------|--------------|---------------|
| $\Omega$  | the measurable domain of data | $\Omega \subset \mathbb{R}^d$ |
| $c(\cdot)$ | a binary function defined on $\Omega$ | - |
| $D_H$     | high-fidelity sample | $D_H = \{(x^H_i, y^H_i)\}_{i=1}^n$ |
| $D_L$     | low-fidelity sample | $D_L = \{(x^L_i, y^L_i)\}_{i=1}^n$ |
| $X_H$     | points of high-fidelity sample | $X_H = \{x^H_i\}_{i=1}^n$ |
| $X_L$     | points of low-fidelity sample | $X_L = \{x^L_i\}_{i=1}^n$ |
| $f_H(\cdot)$ | latent Gaussian Process for high-fidelity | $f_H(x) = \rho f_L(x) + \delta(x)$ |
| $f_L(\cdot)$ | latent Gaussian Process for low-fidelity | - |
| $\delta(\cdot)$ | latent residual Gaussian Process | $\delta(\cdot)$ |
| $\rho$    | linear coefficient for co-kriging dependency | $\rho \in \mathbb{R}$ |
| $k_l(\cdot, \cdot)$ | prior kernel for $f_L(\cdot)$ | - |
| $k_d(\cdot, \cdot)$ | prior kernel for $\delta(\cdot)$ | - |
| $\theta_l$ | parameters of kernel $k_l$ | a multi-dimensional real vector |
| $\theta_d$ | parameters of kernel $k_d$ | a multi-dimensional real vector |
| $\sigma(\cdot)$ | logistic function | $\sigma(z) = \frac{1}{1 + \exp(-z)}$ |
| $\omega(\cdot)$ | first derivative of $\sigma(\cdot)$ | $\omega(z) = \sigma(z)(1 - \sigma(z))$ |
| $\zeta(\cdot)$ | second derivative of $\sigma(\cdot)$ | $\zeta(x) = \sigma(x)(1 - \sigma(x))(1 - 2\sigma(x))$ |
| $\lambda$ | log-likelihood | $\log p(y^L, y^H | x, X_H, \rho, \theta_l, \theta_d)$ |
| $L$       | approximate marginal log-likelihood | - |

Table 7: Some of notations used in the paper.

### A.1 Inference for equation (16)

\[
\frac{\partial \log |B|}{\partial \rho} = \text{tr} \left( B^{-1} \frac{\partial B}{\partial \rho} \right) = \text{tr} \left( B^{-1} \left( \frac{\partial W^{\frac{1}{2}}}{\partial \rho} K W^{\frac{1}{2}} + W^{\frac{1}{2}} K \frac{\partial W^{\frac{1}{2}}}{\partial \rho} \right) \right) = 
\]

\[
= \text{tr} \left( B^{-1} \left( W^{-\frac{1}{2}} W^{\frac{1}{2}} \right) \frac{\partial W^{\frac{1}{2}}}{\partial \rho} K W^{\frac{1}{2}} \right) + \text{tr} \left( B^{-1} W^{\frac{1}{2}} K \frac{\partial W^{\frac{1}{2}}}{\partial \rho} W^{\frac{1}{2}} W^{-\frac{1}{2}} \right) = 
\]

\[
= \text{tr} \left( W^{\frac{1}{2}} B^{-1} W^{-\frac{1}{2}} \left( W^{\frac{1}{2}} \frac{\partial W^{\frac{1}{2}}}{\partial \rho} \right) \right) + \text{tr} \left( W^{-\frac{1}{2}} B^{-1} W^{\frac{1}{2}} K \left( \frac{\partial W^{\frac{1}{2}}}{\partial \rho} W^{\frac{1}{2}} \right) \right) = 
\]

\[
= \text{tr} \left( (K^{-1} + W)^{-1} \frac{\partial W}{\partial \rho} \right) = \sum_{\text{all elements}} (K^{-1} + W)^{-1} \frac{\partial W}{\partial \rho} \right) 
\]

The last line of (18) is obtained because of the following identities:

\[
\frac{\partial W^{\frac{1}{2}}}{\partial \rho} W^{\frac{1}{2}} + W^{\frac{1}{2}} \frac{\partial W^{\frac{1}{2}}}{\partial \rho} = \frac{\partial W^{\frac{1}{2}} W^{\frac{1}{2}}}{\partial \rho} = \frac{\partial W}{\partial \rho} \quad (19)
\]

\[
KW^{\frac{1}{2}} B^{-1} W^{-\frac{1}{2}} = K \left( W^{\frac{1}{2}} B^{-1} W^{\frac{1}{2}} \right) W^{-1} = K (K + W^{-1})^{-1} W^{-1} = (K^{-1} + W)^{-1} \quad (20)
\]
\[ W^{-\frac{1}{2}} W^{\frac{1}{2}} B^{-1} K = W^{-1} \left( W^{\frac{1}{2}} B^{-1} W^{\frac{1}{2}} \right) K = W^{-1} (K + W^{-1})^{-1} K = (K^{-1} + W)^{-1} \tag{21} \]

### A.2 Components of (12)

Let us denote \( M \triangleq (K^{-1} + W)^{-1} \).

For indices \( i \) corresponding to low-fidelity data on \( X_L \) (\( i = 1 \ldots n_l \)):

\[ M_{i,i} \frac{\partial^3}{\partial \xi_i^3} \lambda \equiv M_{i,i} \zeta(f_L(x_L^i)) \tag{22} \]

For indices \( i \) corresponding to low-fidelity data on \( X_H \) (\( i = n_l + 1 \ldots n_l + n_h \)):

\[ \left( M_{i,i} \frac{\partial^3}{\partial \xi_i^3} + 2M_{i,i+n_h} \frac{\partial^3}{\partial \xi_{i+n_h} \partial \xi_i^3} + M_{i+n_h,i+n_h} \frac{\partial^3}{\partial \xi_{i+n_h}^3} \right) \lambda \equiv \left( M_{i,i} \rho^3 + 2M_{i,i+n_h} \rho^2 + M_{i+n_h,i+n_h} \rho \right) \zeta(\rho f^L(x_H^i-x_L^{i-n_l}) + \delta_{i-n_l}) \tag{23} \]

For indices \( i \) corresponding to delta on \( X_H \) (\( i = n_l + n_h + 1 \ldots n_l + 2n_h \)):

\[ \left( M_{i,i} \frac{\partial^3}{\partial \xi_i^3} + 2M_{i,i+n_h} \frac{\partial^3}{\partial \xi_{i-n_h} \partial \xi_i^3} + M_{i-n_h,i-n_h} \frac{\partial^3}{\partial \xi_{i-n_h}^3} \right) \lambda \equiv \left( M_{i,i} + 2M_{i,i-n_h} \rho + M_{i-n_h,i-n_h} \rho^2 \right) \zeta(\rho f^L(x_H^i-x_L^{i-n_l-n_h}) + \delta_{i-n_l-n_h}) \tag{24} \]

### A.3 Specifications of implementation

For experiments we used Python 3.6.

- Implementations of Gaussian Process Classifiers and Logistic Regressions were used from scikit-learn package\(^2\). Out method was implemented on top of GaussianProcessClassifier module from this package;
- Classifiers based on GPs used isotropic RBF kernels;
- Classifiers based on GPs and Logistic Regression were used in the pipeline with the Standard Scaler features preprocessor;
- Implementation of Gradient Boosting Classifier was used from XGBoost module\(^3\) with the following parameters: \( n_{\text{estimators}}=100, \max_{\text{depth}}=3, \text{learning\_rate}=0.05, \text{subsample}=0.85 \);
- For each run of the evaluation procedure we generated a random training subsample that has at least one label of each class (both for low- and high-fidelity subsamples), that is, positive and negative;
- The set of random seeds for different runs was shared across series of method-dataset evaluations.

---

\(^2\)http://scikit-learn.org/
\(^3\)https://xgboost.readthedocs.io