Probabilistic Machine Learning
(CS772A, Fall 2022)
Homework 2
Due Date: October 17, 2022 (11:59pm)

Instructions:

- Only electronic submissions will be accepted. Your main PDF writeup must be typeset in LaTeX (please also refer to the "Additional Instructions" below).
- Your submission will have two parts: The main PDF writeup (to be submitted via Gradescope `https://www.gradescope.com/`) and the code for the programming part (to be submitted via this Dropbox link: `https://tinyurl.com/5brzd8f7`). Both parts must be submitted by the deadline to receive full credit (**delay in submitting either part would incur late penalty for both parts**). We will be accepting late submissions upto 72 hours after the deadline (with every 24 hours delay incurring a 10% late penalty, applied on per-hour delay basis). We won't be able to accept submissions after that.
- We have created your Gradescope account (you should have received the notification). Please use your IITK CC ID (not any other email ID) to login. Use the "Forgot Password" option to set your password.

Additional Instructions

- We have provided a LaTeX template file `hw2sol.tex` to help typeset your PDF writeup. There is also a style file `pmi.sty` that contain shortcuts to many of the useful LaTeX commends for doing things such as boldfaced/calligraphic fonts for letters, various mathematical/greek symbols, etc., and others. Use of these shortcuts is recommended (but not necessary).
- Your answer to every question should begin on a new page. The provided template is designed to do this automatically. However, if it fails to do so, use the `\clearpage` option in LaTeX before starting the answer to a new question, to *enforce* this.
- While submitting your assignment on the Gradescope website, you will have to specify on which page(s) is question 1 answered, on which page(s) is question 2 answered etc. To do this properly, first ensure that the answer to each question starts on a different page.
- Be careful to flush all your floats (figures, tables) corresponding to question n before starting the answer to question $n+1$ otherwise, while grading, we might miss your important parts of your answers.
- Your solutions must appear in proper order in the PDF file i.e. solution to question n must be complete in the PDF file (including all plots, tables, proofs etc) before you present a solution to question $n+1$.
- For the programming part, all the code and README should be zipped together and submitted as a single file named `yourrollnumber.zip`. Please DO NOT submit the data provided.

Problem 1: Gaussian Processes (30 marks)

Part 1: GP Posterior (10 marks)

When discussing about GP regression, we saw that we can bypass the computation of GP posterior and can directly compute the posterior predictive $p(y_*|\boldsymbol{y})$ for a new input $\boldsymbol{x}_*$. Suppose we do care about the GP posterior and would like to derive its expression, given training data $(\mathbf{X}, \boldsymbol{y}) = \{\boldsymbol{x}_n, y_n\}_{n=1}^N$.

Assume a zero mean GP prior $p(f) = \mathcal{GP}(0, \kappa)$ which, from the GP definition, is equivalent to $p(\boldsymbol{f}) = \mathcal{N}(\mathbf{0}, \mathbf{K})$ where $\boldsymbol{f} = [f(\boldsymbol{x}_1,), \ldots, f(\boldsymbol{x}_N)]^\top$ is an $N \times 1$ vector and $\mathbf{K}$ is the $N \times N$ kernel matrix with $K_{nm} = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$. Assume a likelihood model $p(y_n|\boldsymbol{x}_n, f) = \mathcal{N}(y_n|f(\boldsymbol{x}_n), \sigma^2)$, where $f \sim \mathcal{GP}(0, \kappa)$. Derive the expression for the GP posterior, i.e., $p(\boldsymbol{f}|\boldsymbol{y})$. You are free to use standard results for Gaussians.

Part 2: Visualizing GP Priors and Posteriors for Regression (20 marks)

Assume a GP prior $\mathcal{GP}(0, \kappa)$ where κ is the squared exponential (SE) kernel $\kappa(x, x') = \rho^2 \exp\left(-\frac{(x-x')^2}{\ell^2}\right)$. Note that this is the scalar input version of the SE kernel, and it can be generalized to the vector input case by replacing $(x - x')^2$ by $(\boldsymbol{x} - \boldsymbol{x}')^\top(\boldsymbol{x} - \boldsymbol{x}')$. Assume $\rho^2 = 1$.

Our data will consist of scalar inputs and will be generated using the model $y = \sin(x) + \epsilon_n$ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$ where $\sigma^2 = 0.05$. We will generate $N = 100$ uniformly spaced inputs $x_1, \ldots, x_N$ in the interval $[0, 4\pi]$ and generate the corresponding outputs $y_1, \ldots, y_N$ from the above model.

For each of the following 5 values of ℓ from $[0.2, 0.5, 1, 2, 10]$, your task will be the following

- Draw a random sample from the GP prior $p(\boldsymbol{f}) = \mathcal{N}(\mathbf{0}, \mathbf{K})$ and plot it.
- Plot the *mean* of the GP posterior (from Part 1), again on the same figure but with a different color.
- On the same plot, also show the true function ($\sin(x)$) evaluated at the generated inputs (use a different color for this too). Note that this curve will be the same in all the 5 cases.

Your code should be submitted as a Python notebook.

You may use any library function to draw from a multivariate normal distribution. Also note that, when computing the kernel matrix $\mathbf{K}$, you may need to add a small positive number to the diagonal entries to make it invertible.

What difference do you see between the plots generated using $\ell = [0.2, 0.5, 1, 2, 10]$, in particular w.r.t. shapes of prior/posterior of GP vs the true function?

Problem 2: Speeding Up Gaussian Processes (20 marks)

Consider Gaussian Process (GP) regression where $y_n = f(\boldsymbol{x}_n) + \epsilon_n$ with f modeled by $\mathcal{GP}(0, \kappa)$ where GP mean function is 0 and kernel/covariance function is κ, and noise $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$. For simplicity, we will assume the noiseless setting, so $y_n = f(\boldsymbol{x}_n) = f_n$. Given N training inputs $(\mathbf{X}, \boldsymbol{f}) = \{\boldsymbol{x}_n, f_n\}_{n=1}^N$, we have seen that the posterior predictive distribution for a new input $\boldsymbol{x}_*$ is

$$p(f_*|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{f}) = \mathcal{N}(f_*|\boldsymbol{k}_*^\top \mathbf{K}^{-1} \boldsymbol{f}, \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^\top \mathbf{K}^{-1} \boldsymbol{k}_*)$$

In the above, $\mathbf{K}$ is the $N \times N$ kernel matrix of training inputs and $\boldsymbol{k}_*$ is $N \times 1$ vector of kernel based similarities of $\boldsymbol{x}_*$ with each of the training inputs. As we know, the above has $O(N^3)$ cost due to $N \times N$ matrix inversion.

Let's consider a way to reduce this cost to make GPs more scalable. To do this, suppose there are another set of *pseudo* training inputs $\mathbf{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M\}$ with $M \ll N$, along with their respective noiseless *pseudo* outputs $\boldsymbol{t} = \{t_1, \ldots, t_M\}$ modeled by the same GP, i.e., $t_m = f(\boldsymbol{z}_m)$. Note again that $(\mathbf{Z}, \boldsymbol{t})$ are NOT known.

Now assume the likelihood for each training output f_n to be modeled by a posterior predictive having the same form as the GP regression's posterior predictive (given above) but with $(\mathbf{Z}, \boldsymbol{t})$ acting as "pseudo" training data.

$$p(f_n|\boldsymbol{x}_n, \mathbf{Z}, \boldsymbol{t}) = \mathcal{N}(f_n|\tilde{\boldsymbol{k}}_*^\top \tilde{\mathbf{K}}^{-1}\boldsymbol{t}, \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) - \tilde{\boldsymbol{k}}_n^\top \tilde{\mathbf{K}}^{-1}\tilde{\boldsymbol{k}}_n)$$

In the above, $\tilde{\mathbf{K}}$ is the $M \times M$ kernel matrix of the pseudo inputs $\mathbf{Z}$ and $\tilde{\boldsymbol{k}}_n$ is the $M \times 1$ vector of kernel based similarities of $\boldsymbol{x}_n$ with each of the pseudo inputs $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M$.

For this problem setup, your goals are the following

1. Derive and write down the expression of the posterior predictive distribution for the output y_* of a new input $\boldsymbol{x}_*$, i.e., $p(y_*|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{f}, \mathbf{Z})$. Note that here we are assuming that we have estimated the unknown pseudo inputs $\mathbf{Z}$ which this posterior predictive will be conditioned on, in addition to the actual training inputs and outputs $(\mathbf{X}, \boldsymbol{f})$. Note however that the pseudo outputs $\boldsymbol{t}$ will still need to be marginalized out to obtain this posterior predictive, so your derivation must also do that. How does this posterior predictive for y_* compare with the usual GP's posterior predictive for y_* in terms of computational cost?
2. Part (1) requires the pseudo inputs $\mathbf{Z}$. Since we don't know them, we must estimate them from the actual training data $(\mathbf{X}, \boldsymbol{f})$. Let's use MLE-II to estimate $\mathbf{Z}$. In particular, derive the expression for the marginal likelihood $p(\boldsymbol{f}|\mathbf{X}, \mathbf{Z})$. You do not have to show how to solve the optimization problem for MLE-II. Just write down the MLE-II objective.

Note: Feel free to use properties of Gaussians (e.g., marginals from conditionals) to avoid deriving everything from scratch.

Problem 3 (30 marks)

Consider linear regression with likelihood defined by Student t distribution $p(y_n|\boldsymbol{x}_n, \boldsymbol{w}, \sigma^2, \nu) = \mathcal{T}(y_n|\boldsymbol{w}^\top \boldsymbol{x}_n, \sigma^2, \nu)$ and a Gaussian prior on the weights $\boldsymbol{w}$, i.e., $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|0, \rho^2\mathbf{I}_D)$. A Student t likelihood is often better than a Gaussian likelihood since it models outliers better (since it is a heavy-tailed distribution). Assume we are given N training examples, $(\mathbf{X}, \boldsymbol{y}) = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ to infer $\boldsymbol{w}$.

Unfortunately, the Student t likelihood is not conjugate to the Gaussian prior! However, thankfully, the Student t distribution can be expressed in the following "infinite mixture" form

$$\mathcal{T}(y|\mu, \sigma^2, \nu) = \int \mathcal{N}(y|\mu, \sigma^2/z)\text{Gamma}(z|\frac{\nu}{2}, \frac{\nu}{2})dz$$

The above is called a "Gaussian scale mixture) (note that variance is also called the scale). Essentially, we obtain Student t by taking infinite many Gaussians, each with a different variance σ^2/z, where z is another latent variable that we have introduced, and then integrating out z.

Although estimating $\boldsymbol{w}$ would have been otherwise hard due to lack of conjugacy in this case, if we explicitly also keep the variables $z_1, \ldots, z_N$ in the model, this will give us an "augmented" model that has conjugacy with a simple inference procedure!

Essentially, in this augmented model, we can consider the joint distribution of the output y_n and the augmented variable z_n. So, instead of $\mathcal{T}(y|\mu, \sigma^2, \nu)$, we will consider $p(y, z|\mu, \sigma^2, \nu) = \mathcal{N}(y|\mu, \sigma^2/z)\text{Gamma}(z|\frac{\nu}{2}, \frac{\nu}{2})$.

In our linear regression problem, since the z_n's that we will introduce for each $\mathcal{T}(y_n|\boldsymbol{w}^\top \boldsymbol{x}_n, \sigma^2, \nu)$ aren't known, these need to be inferred as well, along with our main variable of interest $\boldsymbol{w}$.

(1) Construct a Gibbs sampler for $p(\boldsymbol{w}, z_1, \ldots, z_N|\mathbf{X}, \boldsymbol{y})$. Derive the conditional posteriors of all the unknowns and clearly write down their expressions of their parameters. Assume all other unknowns (σ^2, ν, ρ^2) to be known.

Avoid very detailed steps in the derivations. If some updates are easy to obtain using standard formulae (e.g., Gaussian posterior updates), please feel free to use those.

(2) Derive an EM algorithm for this model such that the E step infers the augmented variables $z_1, \ldots, z_N$ and the M step estimates the weight vector $\boldsymbol{w}$. Assume all other unknowns (σ^2, ν, ρ^2) to be known.

Problem 4: EM for Sparse Modeling (20 marks)

Consider a linear regression model $\boldsymbol{y} = \mathbf{X}\boldsymbol{w} + \boldsymbol{\epsilon}$ with $\boldsymbol{y} = [y_1, \ldots, y_N]^\top$ is the $N \times 1$ response vector, $\mathbf{X}$ is the $N \times D$ feature matrix, and $\boldsymbol{\epsilon} = [\epsilon_1, \ldots, \epsilon_N]^\top$ is the $N \times 1$ vector of i.i.d. Gaussian noise $\mathcal{N}(0, \sigma^2)$. Let us assume the following prior on each entry of the weight vector $\boldsymbol{w} \in \mathbb{R}^D$

$$p(w_d|\sigma, \gamma_d) = \begin{cases} \mathcal{N}(0, \sigma^2 v_0), & \text{if } \gamma_d = 0 \\ \mathcal{N}(0, \sigma^2 v_1), & \text{if } \gamma_d = 1 \end{cases}$$

where $v_1 > v_0 > 0$. Further assume the priors $p(\gamma_d) = \text{Bernoulli}(\theta)$, $d = 1, \ldots, D$, $p(\theta) = \text{Beta}(a_0, b_0)$, and $p(\sigma^2) = \text{IG}(\nu/2, \nu\lambda/2)$, where IG denotes the inverse-gamma prior in its shape-scale paramerization. Note that the prior on w_d can also be written as $p(w_d|\sigma, \gamma_d) = \mathcal{N}(0, \sigma^2 \kappa_{\gamma_d})$ with $\kappa_{\gamma_d} = \gamma_d v_1 + (1 - \gamma_d) v_0$.

- What is the effect of assuming the above prior on $\boldsymbol{w}$ (answer not using more than 50 words)?
- Derive an EM algorithm for doing inference for this model. Your algorithm should give the (conditional) posterior over the weight vector $\boldsymbol{w}$ and point estimates (MAP) for the remaining unknowns $(\boldsymbol{\gamma}, \sigma^2, \theta)$.