Probabilistic Machine Learning (CS772A)
Homework 1 (Due date: Sept 2, 2017, 11:59pm)

Instructions

- We will only accept electronic submissions and the main writeup must be as a PDF file. If you are handwriting your solutions, please scan the hard-copy and convert it into PDF. Your name and roll number should be clearly written at the top. In case you are submitting multiple files, all files must to be zipped and **submitted as a single file** (named: your-roll-number.zip). Please do not email us your submissions. Your submissions have to be uploaded at the following link: `https://tinyurl.com/y9ebkxbe`).
- Each late submission will receive a 10% penalty per day for up to 3 days. No submissions will be accepted after the 3rd late day.

Problem 1 (20 marks)

Consider N count-valued observations $\{x_1, x_2, \ldots, x_N\}$ drawn i.i.d. from a Poisson distribution $p(x|\lambda) = \frac{\lambda^x e^{-\lambda}}{x!}$ where λ is the rate parameter of the Poisson. Assume a gamma prior on λ, i.e., $p(\lambda) = \text{Gamma}(\lambda; \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda}$, where $\alpha > 0$ is the *shape parameter* and $\beta > 0$ is the *rate parameter*, respectively, of the gamma.[1]. Note that, for this parameterization of gamma distribution, the prior's *mode* is $\frac{\alpha-1}{\beta}$ and mean is $\frac{\alpha}{\beta}$.

- Derive the MLE and MAP estimates for λ.
- Derive the expression for the full posterior distribution for λ.
- Show that the MAP estimate (i.e., mode of the posterior) can be written as weighted combination of the MLE estimate and the prior's mode. Likewise, show that the posterior's *mean* can be written as a weighted combination of the MLE estimate and the prior's *mean*.

Problem 2 (10 marks)

(As you observe more and more..) Recall that, for a Bayesian linear regression model with $p(y|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}^\top \boldsymbol{x}, \beta^{-1})$ and $p(\boldsymbol{w}) = \mathcal{N}(0, \lambda^{-1}\mathbf{I})$, the *posterior predictive distribution* is $p(y_*|\boldsymbol{x}_*) = \mathcal{N}(\mu_N^\top \boldsymbol{x}_*, \beta^{-1} + \boldsymbol{x}_*^\top \Sigma_N \boldsymbol{x}_*) = \mathcal{N}(\mu_N^\top \boldsymbol{x}_*, \sigma_N^2(\boldsymbol{x}_*))$. Here μ_N and Σ_N are the mean and covariance matrix of the Gaussian posterior on $\boldsymbol{w}$, s.t., $\mu_N = \Sigma(\beta \sum_{n=1}^N y_n \boldsymbol{x}_n)$ and $\Sigma_N = (\beta \sum_{n=1}^N \boldsymbol{x}_n \boldsymbol{x}_n^\top + \lambda \mathbf{I})^{-1}$. We have used the subscripts N to denote that the model is learned using N training examples.

What will happen to the variance of the posterior predictive distribution as the training set size N increases. In particular, does $\sigma_{N+1}^2(\boldsymbol{x}_*)$ become smaller, or larger, or remain the same as $\sigma_N^2(\boldsymbol{x}_*)$? You need to formally justify your answer. To do this, you may need to make use of the following matrix identity:

$$(\mathbf{M} + \boldsymbol{v}\boldsymbol{v}^\top)^{-1} = \mathbf{M}^{-1} - \frac{(\mathbf{M}^{-1}\boldsymbol{v})(\boldsymbol{v}^\top \mathbf{M}^{-1})}{1 + \boldsymbol{v}^\top \mathbf{M}^{-1}\boldsymbol{v}}$$

Where $\mathbf{M}$ denotes a square matrix and $\boldsymbol{v}$ denotes a column vector.

[1] There is an alternate parameterization of gamma in terms of shape α and scale θ, for which $p(\lambda) \propto \lambda^{\alpha-1}e^{-\frac{\lambda}{\theta}}$

Problem 3 (40 marks)

(Different Flavours of Generative Classification) Suppose we have N labeled training examples $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$ and our goal is to design a generative classification model using this data. Assume feature vector $\boldsymbol{x}_n$ to be D dimensional and label $y_n \in \{1, \ldots, K\}$, i.e., multi-class classification with K classes. Consider three cases:

- Each feature x_{nd} in $\boldsymbol{x}_n$ is real-valued, i.e., $\boldsymbol{x}_n \in \mathbb{R}^D$.
- Each feature x_{nd} in $\boldsymbol{x}_n$ is binary, i.e., $\boldsymbol{x}_n \in \{0, 1\}^D$.
- Each feature x_{nd} in $\boldsymbol{x}_n$ is discrete with V possible values, i.e., $\boldsymbol{x}_n \in \{1, \ldots, V\}^D$.

To design the model, we need to define the class-prior distribution $p(y)$ and K class-conditional distributions $p(\boldsymbol{x}|y)$, and estimate the parameters of these distributions. To reduce the number of parameter to be estimated for the class-conditionals, we will assume them, for each of the three cases, to be of the form $p(\boldsymbol{x}_n|y_n = k) = \prod_{d=1}^D p(x_{nd}|y_n = k)$, where the type of distribution $p(x_{nd}|y_n = k)$ will depend on the type of the feature x_{nd} (real/binary/discrete; note however that, in each of the three cases, *all* features in $\boldsymbol{x}_n$ are of same type).

- Which class-conditional distributions would you choose for $p(\boldsymbol{x}_n|y_n = k)$ (or its constituents $p(x_{nd}|y_n = k)$) for each of the above three cases? Name these distributions. What parameters define these distributions?
- For each of the three cases, assuming the class prior distribution $p(y)$ to be multinoulli with parameter $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_K\}$, i.e, $p(y_n|\boldsymbol{\pi}) = \prod_{k=1}^K \pi_k^{\mathbb{I}[y_n=k]}$, derive the MLE solution for *all the parameters* of this generative model (basically, $\boldsymbol{\pi}$ and the parameters defining the class-conditionals $p(x_{nd}|y_n = k)$).

  Note: With some thought and using some intuition, you should be able to "guess" the MLE estimates of these parameters, without doing a full derivation of MLE. If you are confident in your guess and intuition, we would accept that answer and give you full credit even if you don't provide a full derivation (but the guess better be correct since there won't be any partial credits if the guessed answer is incorrect :-)).

Problem 4 (30 marks): Programming Assignment

(Bayesian Linear Regression) Consider a toy data set consisting of 10 training examples $\{x_n, y_n\}_{n=1}^{10}$ with each input x_n as well as the output y_n being scalars. The data is given below.

$$\begin{aligned} \boldsymbol{x} &= [-2.23, -1.30, -0.42, 0.30, 0.33, 0.52, 0.87, 1.80, 2.74, 3.62]; \\ \boldsymbol{y} &= [1.01, 0.69, -0.66, -1.34, -1.75, -0.98, 0.25, 1.57, 1.65, 1.51] \end{aligned}$$

We would like to learn a Bayesian linear regression model using this data, assuming a Gaussian likelihood model for the outputs with fixed noise precision $\beta = 4$. However, instead of working with the original scalar-valued inputs, we will map each input x using a degree-k polynomial as $\phi_k(x) = [1, x, x^2, \ldots, x^k]^\top$. Note that, when using the mapping ϕ_k, each original input becomes $k + 1$ dimensional. Denote the entire set of mapped inputs as $\phi_k(\boldsymbol{x})$, a $10 \times (k + 1)$ matrix. We will consider three cases: $k = 1, 2, 3$, and learn a Bayesian linear regression model for each case. Assume the following prior on the regression weights: $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \mathbf{0}, \mathbf{I})$.

- Compute and plot the **mean** of the posterior predictive distribution $p(y_*|\phi_k(x_*), \phi_k(\boldsymbol{x}), \boldsymbol{y}, \beta)$ on the interval $x_* \in [-4, 4]$. On the same plot, also show the predictive posterior mean plus-or-minus two times the predictive posterior standard deviation (see slide 18 from lecture-3 to get a sense of what such a plot looks like).
- Compute the marginal likelihood $p(\boldsymbol{y} \mid \phi_k(\boldsymbol{x}), \beta)$ of the training data for each of the 3 mappings $k = 1, 2, 3$. Which of these 3 "models" seems to explain the data the best?

- Suppose you could include an additional training input x' (along with its output y') to "improve" your learned model using this additional example. Where in the region $x \in [-4, 4]$ would you like to choose x' to be? Explain your answer briefly,

You may use any programming language (e.g., MATLAB or Python) but you should implement the code yourself (and not use an existing implementation of Bayesian linear regression). Submit the plots as well as the code.

Problem 5 (Not for Credit, but Recommended)

(Reading Assignment) For this problem, your task will be to study **Edward**[2], which is a software framework for quickly prototyping probabilistic/Bayesian models. I highly encourage you to explore Edward, install and play with it. We might find it useful for your class project, or if you are doing research on probabilistic/Bayesian modeling.

Another useful framework designed with similar goals is PyMC3. A documentation is available at `https://peerj.com/articles/cs-55.pdf`. The github link with instructions is here: `https://github.com/pymc-devs/pymc3`.

[2] `https://arxiv.org/pdf/1610.09787v2.pdf` and `https://arxiv.org/pdf/1701.03757v1.pdf`