CS 6190: Probabilistic Modelling Spring 2019

Homework 2

Abhinav Kumar (u1209853)

Analytical problems [60 points + 55 bonus]

1. [10 points] Given a Gaussian likelihood, $p(x|\mu,\sigma) = \mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)$,
   (a) [5 points] show that given σ fixed, the Jeffery's prior over μ, $\pi_J(\mu) \propto 1$;
   (b) [5 points] show that given μ fixed, the Jeffery's prior over σ, $\pi_J(\sigma) \propto \frac{1}{\sigma}$.

   We have

$$\pi_J(\boldsymbol{\theta}) \propto |I(\boldsymbol{\theta})|^{\frac{1}{2}} = \left|-\mathbb{E}\left[\frac{d^2 \ln p(\mathbf{x}|\boldsymbol{\theta})}{d\boldsymbol{\theta}^2}\right]\right|^{\frac{1}{2}}$$

$$\frac{d^2 \ln p(\mathbf{x}|\boldsymbol{\theta})}{d\boldsymbol{\theta}^2} = \frac{d^2\left[-\sigma - \frac{(x-\mu)^2}{2\sigma^2}\right]}{d\boldsymbol{\theta}^2} \tag{1}$$

   (a) When $\boldsymbol{\theta} = \mu$, (1) becomes

$$\frac{d^2 \ln p(x|\mu)}{d\mu^2} = \frac{d^2\left[-\sigma - \frac{(x-\mu)^2}{2\sigma^2}\right]}{d\mu^2} = -\frac{1}{\sigma^2} \tag{2}$$

   Hence, we have,

$$\pi_J(\mu) \propto \left|-\mathbb{E}\left[\frac{d^2 \ln p(x|\mu)}{d\mu^2}\right]\right|^{\frac{1}{2}} = \left|-\mathbb{E}\left(-\frac{1}{\sigma^2}\right)\right|^{\frac{1}{2}} = \frac{1}{\sigma}$$

   But, σ is kept constant and so, $\pi_J(\mu) \propto 1$.

   (b) When $\boldsymbol{\theta} = \sigma$, (1) becomes

$$\frac{d^2 \ln p(x|\sigma)}{d\sigma^2} = \frac{d^2\left[-\sigma - \frac{(x-\mu)^2}{2\sigma^2}\right]}{d\sigma^2} = -\frac{6(x-\mu)^2}{2\sigma^4} \tag{3}$$

   Hence, we have,

$$\begin{aligned}\pi_J(\sigma) &\propto \left|-\mathbb{E}\left[-\frac{6(x-\mu)^2}{2\sigma^4}\right]\right|^{\frac{1}{2}} \\ &= \left|\frac{3}{\sigma^4}\mathbb{E}\left[(x-\mu)^2\right]\right|^{\frac{1}{2}} \\ &= \left|\frac{3}{\sigma^2}\right|^{\frac{1}{2}} \\ &= \frac{3}{\sigma}\end{aligned}$$

   Hence, we have $\pi_J(\sigma) \propto \frac{1}{\sigma}$

2. [5 points] Derive the Jeffery's prior for λ in the Poisson likelihood, $p(x=n) = e^{-\lambda}\frac{\lambda^n}{n!}$.

   We have

$$\pi_J(\theta) \propto |I(\theta)|^{\frac{1}{2}} = \left|-\mathbb{E}\left[\frac{d^2 \ln p(x|\theta)}{d\theta^2}\right]\right|^{\frac{1}{2}}$$

$$\text{But,} \quad \frac{d^2 \ln p(x|\lambda)}{d\lambda^2} = \frac{d^2 \left[-\lambda + n \ln \lambda + c\right]}{d\lambda^2} = -\frac{n}{\lambda^2}$$

$$\text{So, } \pi_J(\theta) \propto \left| \frac{1}{\lambda^2} \mathbb{E}\left[n\right] \right|^{\frac{1}{2}} = \left| \frac{1}{\lambda^2} \lambda \right|^{\frac{1}{2}} = \frac{1}{\sqrt{\lambda}} \tag{4}$$

3. [5 points] Given an infinite sequence of Independently Identically Distributed (IID) random variables, show that they are exchangeable.

   We prove with induction over the sequence. Let's start with the base case

$$\begin{aligned} p(X_1 = x_1, X_2 = x_2) &= p(X_1 = x_1) p(X_2 = x_2) \text{ Using independence} \\ &= p(X_1 = x_2) p(X_2 = x_1) \text{ Using identical nature of random variables} \\ &= p(X_1 = x_2, X_2 = x_1) \end{aligned}$$

   Now, we assume that $X_1, X_2, ..., X_k$ are all IID and exchangeable. We have to show that this is also valid for $k+1$

$$\begin{aligned} &p(X_1 = x_1, X_2 = x_2, ..., X_k = x_k, X_{k+1} = x_{k+1}) \\ &= p(X_1 = x_1, X_2 = x_2, ..., X_k = x_k) p(X_{k+1} = x_{k+1}) \text{ Using independence} \\ &= p(X_1 = x_1) p(X_2 = x_2) ... p(X_k = x_k) p(X_{k+1} = x_{k+1}) \text{ Using IID for two variables} \\ &= p(X_1 = x_1) p(X_2 = x_2) ... p(X_k = x_{k+1}) p(X_{k+1} = x_k) \text{ Using exchangability for two variables} \\ &= p(X_1 = x_1, X_2 = x_2, ..., X_k = x_{k+1}, X_{k+1} = x_k) \end{aligned}$$

   Not that the above argument is valid for any variable x_k and therefore we can exchange any two variables. Hence, we prove the induction case as well.

4. [10 points] We discussed Polya's Urn problem as an example of exchangeability. If you do not recall, please look back at the slides we shared in the course website. Now, given two finite sequences $(0, 1, 0, 1)$ and $(1, 1, 0, 0)$, derive their probabilities and show they are the same.

$$\begin{aligned} p(0,1,0,1) &= \frac{W_0}{B_0 + W_0} \frac{B_0}{B_0 + W_0 + (a-1)} \frac{W_0 + (a-1)}{B_0 + W_0 + 2(a-1)} \frac{B_0 + (a-1)}{B_0 + W_0 + 3(a-1)} \\ p(1,1,0,0) &= \frac{B_0}{B_0 + W_0} \frac{B_0 + (a-1)}{B_0 + W_0 + (a-1)} \frac{W_0}{B_0 + W_0 + 2(a-1)} \frac{W_0 + (a-1)}{B_0 + W_0 + 3(a-1)} \end{aligned}$$

   The denominator in these two terms are exactly the same. The numerators are also the same. Hence the $p(0, 1, 0, 1)$ and $p(1, 1, 0, 0)$ are equal.

5. [10 points] For the logistic regression model, we assign a Gaussian prior over the feature weights, $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \lambda\mathbf{I})$. Please derive the Newton-Raphson updates.

   We want to minimize the posterior $-\ln p(\mathbf{w}|t) = -\ln p(t|\mathbf{w}) - \ln p(\mathbf{w})$ in each of the update step.

$$\mathbb{E}(-\ln p(t|\mathbf{w}) - \ln p(\mathbf{w})) = -\sum_{n=1}^{N} t_n \ln y_n + (1 - t_n) \ln(1 - y_n) + \ln(\lambda) + \frac{1}{2\lambda} \mathbf{w}^T \mathbf{w} \tag{5}$$

   where $y_n = \sigma(\mathbf{w}^T \boldsymbol{\phi}_n)$.

   Differentiating once wrt $\mathbf{w}$, we have

$$\begin{aligned} \nabla_{\mathbf{w}} \mathbb{E} = \nabla_{\mathbf{w}} \mathbb{E}(-\ln p(t|\mathbf{w}) - \ln p(\mathbf{w})) &= -\sum_{n=1}^{N} \frac{t_n}{y_n} y_n(1 - y_n) \boldsymbol{\phi}_n - \frac{(1 - t_n)}{(1 - y_n)} y_n(1 - y_n) \boldsymbol{\phi}_n + \frac{\mathbf{w}}{\lambda} \\ &= \sum_{n=1}^{N} (y_n - t_n) \boldsymbol{\phi}_n + \frac{\mathbf{w}}{\lambda} \\ &= \boldsymbol{\Phi}^T (\mathbf{y} - \mathbf{t}) + \frac{\mathbf{w}}{\lambda} \end{aligned} \tag{6}$$

   where $\boldsymbol{\Phi}$ is a $n \times d$ matrix and each $\boldsymbol{\phi}_n$ is a row vector.

   Differentiating once more wrt $\mathbf{w}$ to get the Hessian, we have

$$\nabla^2_{\mathbf{w}} \mathbb{E} = \nabla^2_{\mathbf{w}} \mathbb{E}(-\ln p(t|\mathbf{w}) - \ln p(\mathbf{w})) = \nabla_{\mathbf{w}} . \nabla^T_{\mathbf{w}} \mathbb{E}(-\ln p(t|\mathbf{w}) - \ln p(\mathbf{w}))$$

$$= \sum_{n=1}^{N} y_n(1-y_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^T + \frac{1}{\lambda}\mathbf{I}$$
$$= \boldsymbol{\Phi}^T\mathbf{R}\boldsymbol{\Phi} + \frac{1}{\lambda}\mathbf{I} \tag{7}$$

where $\mathbf{R} = n \times n$ diagonal matrix with $\mathbf{R}_{nn} = y_n(1-y_n)$.

Now, the Newton Raphson updates are

$$\mathbf{w}_{new} = \mathbf{w}_{old} - (\nabla^2_{\mathbf{w}}\mathbb{E})^{-1}\nabla_{\mathbf{w}}\mathbb{E}$$
$$= \mathbf{w}_{old} - \left(\boldsymbol{\Phi}^T\mathbf{R}\boldsymbol{\Phi} + \frac{1}{\lambda}\mathbf{I}\right)^{-1}\left[\boldsymbol{\Phi}^T(\mathbf{y}-\mathbf{t}) + \frac{\mathbf{w}_{old}}{\lambda}\right] \tag{8}$$

6. [**Bonus**][20 points] For the probit regression model, we assign a Gaussian prior over the feature weights, $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \lambda\mathbf{I})$. Please derive the Newton-Raphson updates.

We want to minimize the posterior $-\ln p(\mathbf{w}|t) = -\ln p(t|\mathbf{w}) - \ln p(\mathbf{w})$ in each of the update step.

$$\mathbb{E}(-\ln p(t|\mathbf{w}) - \ln p(\mathbf{w})) = -\sum_{n=1}^{N} t_n \ln y_n + (1-t_n)\ln(1-y_n) + \ln(\lambda) + \frac{1}{2\lambda}\mathbf{w}^T\mathbf{w} \tag{9}$$

where $y_n = \psi(\mathbf{w}^T\boldsymbol{\phi}_n) = \int_{-\infty}^{\mathbf{w}^T\boldsymbol{\phi}_n} \mathcal{N}(x|0,1)dx$. Hence,

$$\frac{dy_n}{d\mathbf{w}} = \mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n \tag{10}$$

Differentiating (9) once wrt $\mathbf{w}$, we have

$$\nabla_{\mathbf{w}}\mathbb{E} = \nabla_{\mathbf{w}}\mathbb{E}(-\ln p(t|\mathbf{w}) - \ln p(\mathbf{w})) = -\sum_{n=1}^{N} \frac{t_n}{y_n}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n - \frac{(1-t_n)}{(1-y_n)}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n + \frac{\mathbf{w}}{\lambda}$$
$$= \sum_{n=1}^{N} \frac{(y_n-t_n)}{y_n(1-y_n)}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n + \frac{\mathbf{w}}{\lambda} \tag{11}$$
$$= \boldsymbol{\Phi}^T\mathbf{M}(\mathbf{y}-\mathbf{t}) + \frac{\mathbf{w}}{\lambda} \tag{12}$$

where where $\boldsymbol{\Phi}$ is a $n \times d$ matrix and each $\boldsymbol{\phi}_n$ is a row vector. Also, $M = n \times n$ matrix with

$$\mathbf{M}_{nn} = \frac{\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)}{y_n(1-y_n)} \tag{13}$$

Differentiating (11) once more wrt $\mathbf{w}$ to get the Hessian, we have

$$\nabla^2_{\mathbf{w}}\mathbb{E} = \nabla^2_{\mathbf{w}}\mathbb{E}(-\ln p(t|\mathbf{w}) - \ln p(\mathbf{w}))$$
$$= \nabla_{\mathbf{w}}\cdot\sum_{n=1}^{N} \frac{(y_n-t_n)}{y_n(1-y_n)}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n^T + \nabla_{\mathbf{w}}\cdot\frac{\mathbf{w}^T}{\lambda}$$
$$= \sum_{n=1}^{N} \frac{\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n}{y_n(1-y_n)}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n^T - \sum_{n=1}^{N} \frac{(y_n-t_n)}{y_n^2(1-y_n)}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n^T$$
$$+ \sum_{n=1}^{N} \frac{(y_n-t_n)}{y_n(1-y_n)^2}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n^T + \sum_{n=1}^{N} \frac{(y_n-t_n)}{y_n(1-y_n)}\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)(-\mathbf{w}^T\boldsymbol{\phi}_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^T + \frac{1}{\lambda}\mathbf{I}$$
$$= \sum_{n=1}^{N} \left[\frac{1}{y_n(1-y_n)} - \frac{(y_n-t_n)}{y_n^2(1-y_n)} + \frac{(y_n-t_n)}{y_n(1-y_n)^2} - \frac{(y_n-t_n)\mathbf{w}^T\boldsymbol{\phi}_n}{y_n(1-y_n)\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)}\right]\mathcal{N}^2(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^T + \frac{1}{\lambda}\mathbf{I}$$
$$= \boldsymbol{\Phi}^T\mathbf{R}\boldsymbol{\Phi} + \frac{1}{\lambda}\mathbf{I} \tag{14}$$

where $\mathbf{R} = n \times n$ diagonal matrix with

$$\mathbf{R}_{nn} = \left[\frac{1}{y_n(1-y_n)} - \frac{(y_n - t_n)}{y_n^2(1-y_n)} + \frac{(y_n - t_n)}{y_n(1-y_n)^2} - \frac{(y_n - t_n)\mathbf{w}^T\boldsymbol{\phi}_n}{y_n(1-y_n)\mathcal{N}(\mathbf{w}^T\boldsymbol{\phi}_n|0,1)}\right]\mathcal{N}^2(\mathbf{w}^T\boldsymbol{\phi}_n|0,1) \tag{15}$$

Now, the Newton Raphson updates are

$$\begin{aligned}\mathbf{w}_{new} &= \mathbf{w}_{old} - (\nabla^2_{\mathbf{w}}\mathbb{E})^{-1}\nabla_{\mathbf{w}}\mathbb{E} \\ &= \mathbf{w}_{old} - \left(\boldsymbol{\Phi}^T\mathbf{R}\boldsymbol{\Phi} + \frac{1}{\lambda}\mathbf{I}\right)^{-1}\left[\boldsymbol{\Phi}^T\mathbf{M}(\mathbf{y}-\mathbf{t}) + \frac{\mathbf{w}_{old}}{\lambda}\right]\end{aligned} \tag{16}$$

where $\mathbf{M}$ and $\mathbf{R}$ are defined by equation (13) and (15).

7. [10 points] What are the link functions of the following models?

   (a) [5 points] Logistic regression

   (b) [5 points] Poisson regression: $p(x = n) = e^{-\lambda}\frac{\lambda^n}{n!}$ where $\lambda = e^{\mathbf{w}^\top\boldsymbol{\phi}}$.

   A standard linear model (e.g., a simple regression model) can be thought of as having two 'parts'. These are called the structural component and the random component. For example: $t = \mathbf{w}^\top\boldsymbol{\phi} + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2_\epsilon)$. The first terms $\mathbf{w}^\top\boldsymbol{\phi}$ constitute the structural component, and the ϵ (which indicates a normally distributed error term) is the random component.

   When the response variable y is not normally distributed (for example, if the response variable is binary), this approach may no longer be valid. The generalized linear model (GLM) was developed to address such cases. A GLM has three parts, a structural component, a link function, and a response distribution and is given by

$$g(\mathbb{E}_y(t)) = \mathbf{w}^\top\boldsymbol{\phi} \tag{17}$$

   Here, $\mathbf{w}^\top\boldsymbol{\phi}$ is again the structural component, $g()$ is the link function, and $g(\mathbb{E}_y(t))$ is the mean of a conditional response distribution at a given point in the covariate space. The link function does not link values of t and linear predictor, but the mean $\mathbb{E}_y(t)$. Moreover, this function converts the output in the custom range to the $(-\infty, \infty)$ which is the range of $\mathbf{w}^\top\boldsymbol{\phi}$ and hence is called the 'link' function.

   (a) For logistic regression, $y = f(\mathbf{w}^T\boldsymbol{\phi}) = \sigma(\mathbf{w}^T\boldsymbol{\phi})$. The likelihood of t is given by Bernoulli distribution (since the outcomes are two, we do not have any choice other than Bernoulli)

$$p(t) = y^t(1-y)^{1-t} \tag{18}$$

   and therefore $\mathbb{E}(t) = y$. So, using the definition (17), we have $g(\mathbb{E}(t)) = \mathbf{w}^T\boldsymbol{\phi}$ or $g(y) = \sigma^{-1}(y)$. The link function is inverse of logistic function and is given by

$$g(y) = \sigma^{-1}(y) = -\ln\left(\frac{1}{y} - 1\right) = \ln\left(\frac{y}{1-y}\right) \tag{19}$$

   (b) For Poisson regression, $y = e^{\mathbf{w}^\top\boldsymbol{\phi}}$. The likelihood of y is given by

$$p(t = n) = e^{-y}\frac{y^n}{n!} \tag{20}$$

   and therefore $\mathbb{E}_y(t) = y$. So, using the definition (17), we have $g(\mathbb{E}_y(t)) = \mathbf{w}^\top\boldsymbol{\phi}$ or $g(y) = \ln(y)$. The link function is therefore the logarithm function.

   References:

   - Stats Stackexchange - Difference between logit and probit models
   - Stats Stackexchange - Understand Link Function in GLM

8. [10 points] As we discussed in the class, the probit regression model is equivalent to given each feature vector $\boldsymbol{\phi}$, sampling a latent variable z from $\mathcal{N}(z|\mathbf{w}^\top\boldsymbol{\phi}, 1)$, and then sampling the binary label t from the step distribution, $p(t|z) = I(t = 0)I(z < 0) + I(t = 1)I(z \geq 0)$. Show if we marginalize out z, we recover the original likelihood of the probit regression.

$$
\begin{aligned}
p(t=1) &= \int_{-\infty}^{\infty} p(t,z)dz = \int_{-\infty}^{\infty} p(t=1|z)p(z)dz \\
&= \int_{-\infty}^{\infty} I(z>0)p(z)dz \\
&= \int_{0}^{\infty} p(z)dz \\
&= 1 - F(z<0) \quad F \text{ is the CDF of } z \\
&= 1 - F(x < -\mathbf{w}^\top \boldsymbol{\phi}) \quad \text{x is a standard normal variable} \\
&= 1 - F(x > \mathbf{w}^\top \boldsymbol{\phi}) \quad \text{Symmetric nature of standard normal} \\
&= 1 - (1 - F(x < \mathbf{w}^\top \boldsymbol{\phi})) \\
&= F(x < \mathbf{w}^\top \boldsymbol{\phi}) \\
&= \int_{-\infty}^{\mathbf{w}^\top \boldsymbol{\phi}} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx
\end{aligned} \tag{21}
$$

Reference: Wikipedia

9. [**Bonus**][5 points] For polynomial regression, show that given N training points, you can always choose the highest order M for the polynomial terms such that your model results in 0 training error (*e.g.,* mean squared error or mean absolute error). Please give the corresponding regression function as well.

   Let $f(x)$ be a M^{th} order polynomial such that $f(x_i) = y_i \forall i = 1, ..., N$.

   Let $g(x) = f(x) - y$ be a new M^{th} order polynomial. Then, finding f is equivalent to finding the roots of g. Clearly, $g(x)$ should be zero $\forall i = 1, ..., N$.

   Let $g(x) = \prod_{i=1}^{N} (x - x_i)$ by choosing power M exactly as N and x_i as the roots of $g(x)$. This will result in training error being zero for all training data points. Therefore, $f(x) = \prod_{i=1}^{N} (x - x_i) + y$.

10. [**Bonus**][30 points] Consider the decision problem for continuous variables. Given the decision variable t and input variable $\mathbf{x}$, we know $p(t, \mathbf{x})$, and we want to make an optimal decision $y(\mathbf{x})$ to predict t. To this end, we need to define a loss $L(t, y(\mathbf{x}))$ and then find $y(\mathbf{x})$ by minimizing

$$\mathbb{E}[L] = \int \int L(t, y(\mathbf{x}))p(\mathbf{x}, t)\mathrm{d}\mathbf{x}\mathrm{d}t.$$

   (a) [5 points] We mentioned in the class that if we choose the square loss, $L(t, y(\mathbf{x})) = (y(\mathbf{x}) - t)^2$, the optimal decision is given by $y(\mathbf{x}) = \mathbb{E}(t|\mathbf{x})$. We did not show it in the lecture. Show it now.

   (b) [10 points] Let us define a more general loss, $L(t, y(\mathbf{x})) = |y(\mathbf{x}) - t|^q$. Show that when $q = 1$ (mean absolute error), the optimal decision is the conditional median of t given $\mathbf{x}$. Given a random variable Y, its median is the value θ such that $p(Y \leq \theta) = p(Y \geq \theta)$. *Hint: please refer to PRML book page 703, Appendix D, for functional derivative.*

   (c) [15 points] Show that when $q = 0$, the optimal decision is the conditional mode (i.e., MAP estimation).

   We have,

$$\mathbb{E}[L] = \int \int L(t, y(\mathbf{x}))p(\mathbf{x}, t)\mathrm{d}\mathbf{x}\mathrm{d}t$$

   (a)

$$\mathbb{E}[L] = \int_{\mathbf{x}} \int_{t} (y(\mathbf{x}) - t)^2 p(\mathbf{x}, t)\mathrm{d}t\mathrm{d}\mathbf{x}$$

Using Euler Langrange Equations from calculus of variations on the function $P = \int_t (y(\mathbf{x}) - t)^2 p(\mathbf{x}, t) \mathrm{d}t$, we have

$$\begin{aligned}
&\frac{\partial P}{\partial y} = 0 \\
\implies & 2 \int_t (y(\mathbf{x}) - t) p(\mathbf{x}, t) \mathrm{d}t = 0 \\
\implies & y(\mathbf{x}) \int_t p(\mathbf{x}, t) dt = \int_t t p(\mathbf{x}, t) \mathrm{d}t \\
\implies & y(\mathbf{x}) p(\mathbf{x}) = \int_t t p(\mathbf{x}, t) \mathrm{d}t \\
\implies & y(\mathbf{x}) = \int_t t \frac{p(\mathbf{x}, t)}{p(\mathbf{x})} \mathrm{d}t \\
\implies & y(\mathbf{x}) = \int_t t p(t|\mathbf{x}) \mathrm{d}t \\
\implies & y(\mathbf{x}) = \mathbb{E}(t|\mathbf{x}) \qquad (22)
\end{aligned}$$

(b)

$$\mathbb{E}[L] = \int_{\mathbf{x}} \int_t |y(\mathbf{x}) - t|^q p(\mathbf{x}, t) \mathrm{d}t \mathrm{d}\mathbf{x}$$

Using Euler Langrange Equations from calculus of variations on the function

$$P = \int_t |y(\mathbf{x}) - t|^q p(\mathbf{x}, t) \mathrm{d}t \qquad (23)$$

, we have

$$\begin{aligned}
&\frac{\partial P}{\partial y} = 0 \\
\implies & \int_t q|y(\mathbf{x}) - t|^{q-1} sign(y(\mathbf{x}) - t) p(\mathbf{x}, t) \mathrm{d}t = 0 \quad \text{(Using subgradients)}
\end{aligned}$$

We consider two cases $y(\mathbf{x}) \geq t$ and $y(\mathbf{x}) < t$ separately. So, we have

$$\int_{-\infty}^{y(\mathbf{x})} q|y(\mathbf{x}) - t|^{q-1} p(\mathbf{x}, t) \mathrm{d}t - \int_{y(\mathbf{x})}^{\infty} q|y(\mathbf{x}) - t|^{q-1} p(\mathbf{x}, t) \mathrm{d}t = 0 \qquad (24)$$

Using (24), on substituting $q = 1$, we have

$$\begin{aligned}
&\int_{-\infty}^{y(\mathbf{x})} p(\mathbf{x}, t) \mathrm{d}t - \int_{y(\mathbf{x})}^{\infty} p(\mathbf{x}, t) \mathrm{d}t = 0 \\
\implies & \int_{-\infty}^{y(\mathbf{x})} \frac{p(\mathbf{x}, t)}{p(\mathbf{x})} \mathrm{d}t - \int_{y(\mathbf{x})}^{\infty} \frac{p(\mathbf{x}, t)}{p(\mathbf{x})} \mathrm{d}t = 0 \\
\implies & \int_{-\infty}^{y(\mathbf{x})} \frac{p(\mathbf{x}, t)}{p(\mathbf{x})} \mathrm{d}t = \int_{y(\mathbf{x})}^{\infty} \frac{p(\mathbf{x}, t)}{p(\mathbf{x})} \mathrm{d}t \\
\implies & \int_{-\infty}^{y(\mathbf{x})} p(t|\mathbf{x}) dt = \int_{y(\mathbf{x})}^{\infty} p(t|\mathbf{x}) dt \qquad (25)
\end{aligned}$$

which is the conditional median of random variable $t|\mathbf{x}$.

(c) From (24), we have

$$\int_{-\infty}^{y(\mathbf{x})} \frac{q}{|y(\mathbf{x})-t|^{1-q}} p(\mathbf{x},t)\mathrm{d}t - \int_{y(\mathbf{x})}^{\infty} \frac{q}{|y(\mathbf{x})-t|^{1-q}} p(\mathbf{x},t)\mathrm{d}t = 0 \tag{26}$$

When $q \to 0$ and $y(\mathbf{x}) \neq t$, the equation is trivially satisfied.
So, we go back to the original function P that we intended to minimize. From (23), when $q = 0$, the function P can be written as

$$P = \begin{cases} \int\limits_{t \neq y(\mathbf{x})} p(\mathbf{x},t)dt & \text{for } y(\mathbf{x}) \neq t \\ 0 \quad \text{for } y(\mathbf{x}) = t \end{cases}$$
$$\text{or, } P = p(\mathbf{x}) - p(\mathbf{x},t)$$

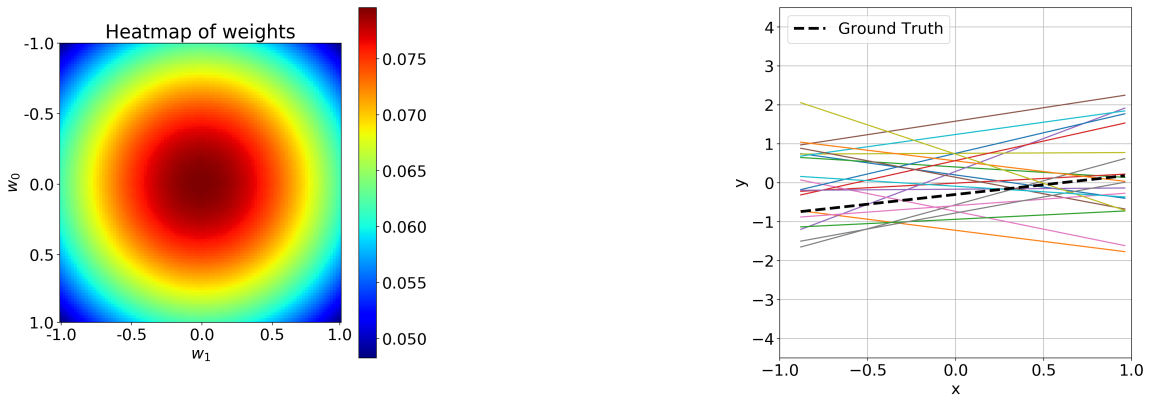
Hence, to minimize P, we have to choose $y(\mathbf{x}) = t$ such that $p(\mathbf{x},t)$ is maximum and therefore

$$\begin{aligned} y(\mathbf{x}) &= arg\max_t p(\mathbf{x},t) \\ &= arg\max_t \frac{p(\mathbf{x},t)}{p(\mathbf{x})} \\ &= arg\max_t p(t|\mathbf{x}) \end{aligned} \tag{27}$$

Practice [40 points + 45 Bonus]

1. [15 Points] Let us generate a simulation dataset for fun. We consider a linear regression model $y(\mathbf{x},\mathbf{w}) = w_0 + w_1 x$. We set the ground-truth $w_0 = -0.3$ and $w_1 = 0.5$. We generate 20 samples $[x_1, \ldots, x_{20}]$ from the uniform distribution in $[-1, 1]$. For each sample x_n, we obtain an sample y_n by first calculating $w_0 + w_1 x_n$ with the ground-truth values of w_0 and w_n, and then adding a Gaussian noise with zero mean, standard deviation 0.2. Now let us verify what we have discussed in the class. We use a Bayesian linear regression model. The prior of $\mathbf{w}$ is $\mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$, and the likelihood for each sample is $p(y_n|\mathbf{x}_n, \mathbf{w}) = \mathcal{N}(y_n|w_0 + w_1 x, \beta^{-1}\mathbf{I})$. Here we set $\alpha = 2$ and $\beta = 25$.

   (a) [3 points] Draw the heat-map of the prior $p(\mathbf{w})$ in the region $w_0 \in [-1, 1]$ and $w_1 \in [-1, 1]$, where you represent the values of $p(\mathbf{w})$ for different choices of $\mathbf{w}$ with different colors. The darker some given color (*e.g.,* red), the larger the value; the darker some the other given color (*e.g.,* blue), the smaller the value. Most colors should be in between. Then sample 20 instances of $\mathbf{w}$ from $p(\mathbf{w})$. For each w, draw a line $y = w_0 + w_1 x$ in the region $x, y \in [-1, 1]$. Ensure these 20 lines are in the same plot. What do you observe?



(a) Heatmap showing distribution of $\mathbf{w}$.

(b) Lines corresponding to 20 random samples of $\mathbf{w}$.

Figure 1: Distribution and lines generated from $\mathbf{w}$

We see the contours of the distribution to be circular which is because of the covariance matrix being scaled version of identity matrix. The lines in $x - y$ planes are now random.

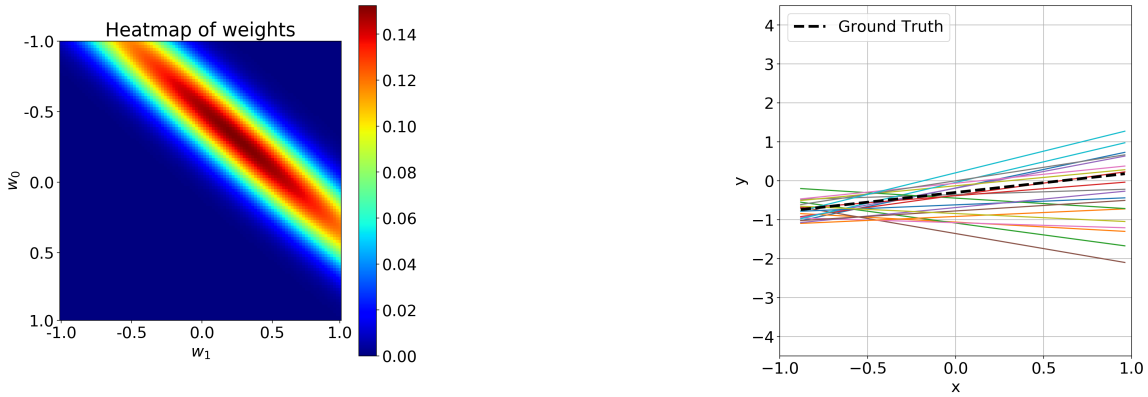
(b) [3 points] Calculate and report the posterior distribution of $\mathbf{w}$ given $(\mathbf{x}_1, y_1)$. Now draw the heat map of the distribution. Also draw the ground-truth of w_0 and w_1 in the heat map. Then from the posterior distribution, sample 20 instances of $\mathbf{w}$, for each of which draw a line $y = w_0 + w_1 x$ in the region $x, y \in [-1, 1]$. Ensure these 20 lines are in the same plot. Also draw (x_1, y_1) as a circle in that plot. What do you observe? Why?

From the class lectures, we know that when $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0)$ and $p(\mathbf{y}|\mathbf{w}, \mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{w}\mathbf{x}, \beta^{-1}\mathbf{I})$, the posterior is given by $p(\mathbf{w}|\mathbf{y}, \mathbf{x}) = N(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$ where $\mathbf{m}_N = \mathbf{S}_N(\mathbf{S}_0^{-1}\mathbf{m}_0 + \beta\mathbf{x}^T\mathbf{y})$ and $\mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \beta\mathbf{x}^T\mathbf{x}$.
Here, we have $\mathbf{S}_0 = \alpha^{-1}\mathbf{I}$ and $\mathbf{m}_0 = \mathbf{0}$. Substituting, the posterior distribution is

$$p(\mathbf{w}|\mathbf{t}, \mathbf{x}) = \mathcal{N}(w|\mathbf{m}_N, \mathbf{S}_N) \quad \text{where } \mathbf{S}_N = (\alpha\mathbf{I} + \beta\mathbf{x}^T\mathbf{x})^{-1} \text{ and } \mathbf{m}_N = \beta\mathbf{S}_N\mathbf{x}^T\mathbf{y} \tag{28}$$

We use this in the code and we get the $p(\mathbf{w}|\mathbf{t}, \mathbf{x}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$ where

$$\mathbf{m}_N = \begin{bmatrix} m_{w_1} \\ m_{w_0} \end{bmatrix} = \begin{bmatrix} 0.35 \\ -0.40 \end{bmatrix}, \mathbf{S}_N = \begin{bmatrix} 0.29 & 0.24 \\ 0.24 & 0.23 \end{bmatrix}$$



(a) Heatmap showing distribution of $\mathbf{w}$.

(b) Lines corresponding to 20 random samples of $\mathbf{w}$.

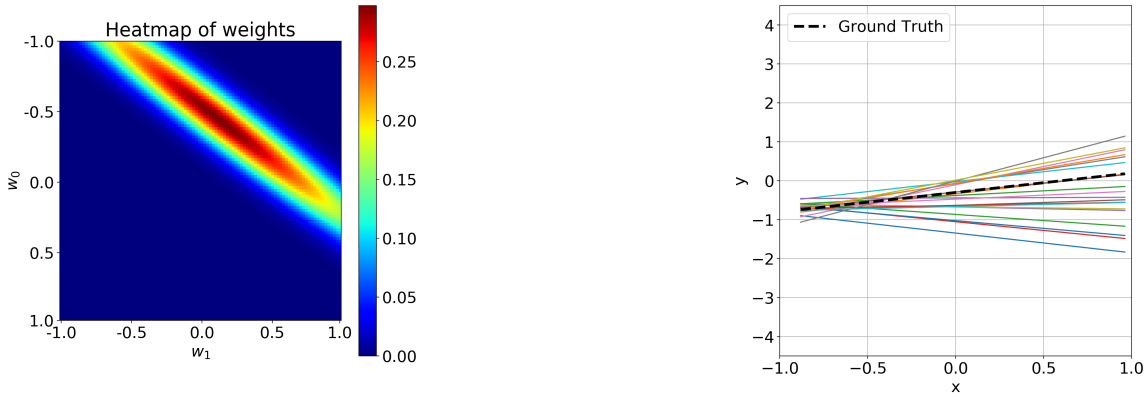
Figure 2: Distribution and lines generated from $\mathbf{w}$

We see the contours of the distribution no longer remains circular which is because of the likelihood playing the role and shifting it towards the true ground truth values. The lines in $x - y$ planes are also now not so random.

(c) [3 points] Calculate and report the posterior distribution of $\mathbf{w}$ given $(\mathbf{x}_1, y_1)$ and $(\mathbf{x}_2, y_2)$. Then draw the plots as the above. What do you observe now?

$p(\mathbf{w}|\mathbf{t}, \mathbf{x}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$ where

$$\mathbf{m}_N = \begin{bmatrix} m_{w_1} \\ m_{w_0} \end{bmatrix} = \begin{bmatrix} 0.35 \\ -0.40 \end{bmatrix}, \mathbf{S}_N = \begin{bmatrix} 0.26 & 0.19 \\ 0.19 & 0.16 \end{bmatrix}$$



(a) Heatmap showing distribution of $\mathbf{w}$.

(b) Lines corresponding to 20 random samples of $\mathbf{w}$.

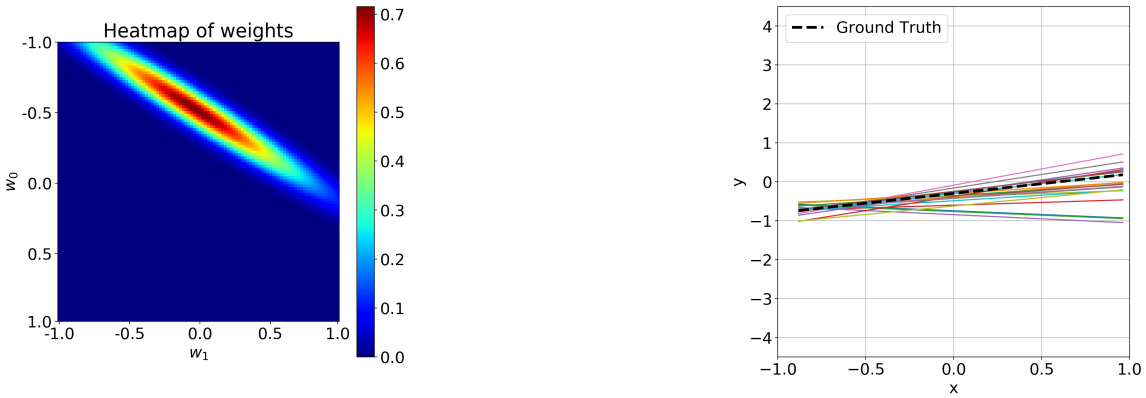
Figure 3: Distribution and lines generated from $\mathbf{w}$

We see the contours of the distribution shifts even more towards the true ground truth values. The lines in $x - y$ planes are also closer to the true values. With the addition of second point, the variation in covariance has reduced. The variation in heatmaps is lot more for w_1 and this is clearly visible in fitted lines in figure 3b have more variation in slope than intercept.

(d) [3 points] Calculate and report the posterior distribution of $\mathbf{w}$ given $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_5, y_5)\}$. Then draw the plots as the above. What do you observe now?

$p(\mathbf{w}|\mathbf{t}, \mathbf{x}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$ where

$$\mathbf{m}_N = \begin{bmatrix} m_{w_1} \\ m_{w_0} \end{bmatrix} = \begin{bmatrix} 0.39 \\ -0.37 \end{bmatrix}, \mathbf{S}_N = \begin{bmatrix} 0.18 & 0.11 \\ 0.11 & 0.08 \end{bmatrix}$$



(a) Heatmap showing distribution of $\mathbf{w}$.

(b) Lines corresponding to 20 random samples of $\mathbf{w}$.

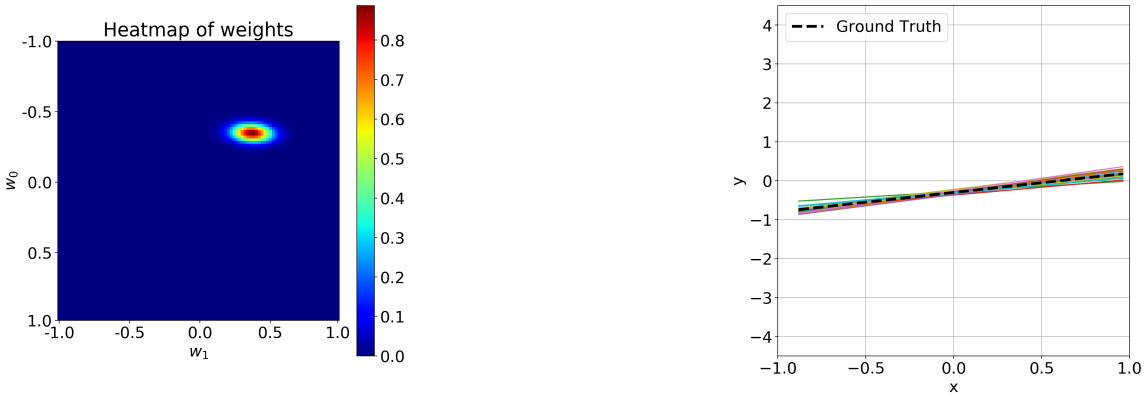
Figure 4: Distribution and lines generated from $\mathbf{w}$

We see the contours of the distribution shifts even more towards the true ground truth values. The lines in $x - y$ planes are more close to the true values.

(e) [3 points] Calculate and report the posterior distribution of $\mathbf{w}$ given all the 20 data points. Then draw the plots as the above. What do you observe now?

$p(\mathbf{w}|\mathbf{t}, \mathbf{x}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$ where

$$\mathbf{m}_N = \begin{bmatrix} m_{w_1} \\ m_{w_0} \end{bmatrix} = \begin{bmatrix} 0.49 \\ -0.30 \end{bmatrix}, \mathbf{S}_N = \begin{bmatrix} 0.01 & 0.00 \\ 0.00 & 0.00 \end{bmatrix}$$



(a) Heatmap showing distribution of $\mathbf{w}$.

(b) Lines corresponding to 20 random samples of $\mathbf{w}$.

Figure 5: Distribution and lines generated from $\mathbf{w}$

We see the contours of the distribution shifted almost at the true ground truth values. The mean values are exactly at the groundtruth values. The variance in the estimation has almost vanished and the fitted lines are in $x - y$ planes are super close to the true values.

2. [25 points] We will implement Logistic regression and Probit regression for a binary classification task — bank-note authentication. Please download the data "bank-note.zip" from Canvas. The features and labels are listed in the file "bank-note/data-desc.txt". The training data are stored in the file "bank-note/train.csv", consisting of 872 examples. The test data are stored in "bank-note/test.csv", and comprise of 500 examples. In both the training and testing datasets, feature values and labels are separated by commas. To ensure numerical stability and avoid overfitting, we assign the feature weights a standard normal prior $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

   (a) [15 points] Implement Newton-Raphson scheme to find the MAP estimation of the feature weights in the logistic regression model. Set the maximum number of iterations to 100 and the tolerance level to be $1e-5$, i.e., when the norm of difference between the weight vectors after one update is below the tolerance level, we consider it converges and stop updating the weights any more. Initially, you can set all the weights to be zero. Report the prediction accuracy on the test data. Now set the initial weights values be to be randomly generated, say, from the standard Gaussian, run and test your algorithm. What do you observe? Why?

   The equation for Newton Raphson update for Logistic Regression is (8). The test accuracy achieved is 99.00%. When we use random weights, the test accuracy achieved is again 99.00%. The performance remains the same suggesting the Hessian based method has less effect of initialization on the final solution.

   (b) [10 points] Implement MAP estimation algorithm for Probit regression model. You can calculate the gradient and feed it to any optimization algorithm, say, L-BFGS. Set the maximum number of iterations to 100 and the tolerance level to $1e-5$. Initially, you can set all the weights to zero. Report the prediction accuracy on the test data. Compared with logistic regression, which one is better? Now set the initial weights values be to be randomly generated, say, from the standard Gaussian, run and test your algorithm. What do you observe? Can you guess why?

   The test accuracy achieved when all weights is zero is 97.40%. Logistic regression is better since the optimization involved is different. LBFGS is a first order method while Newton Raphson is a second order method.

   When we use random weights this goes to 93.40%. First order gradient based methods might not converge better than second order Hessian based methods. Moreover, first order methods are more sensitive to intialization.

   (c) [**Bonus**][15 points]. Implement Newton-Raphson scheme to find the MAP estimation for Probit regression. Report the prediction accuracy.

   We also implemented Newton-Raphson update for Probit regression listed in (16). After using this update equation, the prediction accuracy of Probit regression on the test set with zero initialisation as well as random initialisation is 98.80%. The convergence steps needed for random initialization is much more than the one with zero initialization.

3. [**Bonus**][30 points] We will implement a multi-class logistic regression model for car evaluation task. The dataset is from UCI repository(`https://archive.ics.uci.edu/ml/datasets/car+evaluation`). Please download the processed dataset (car.zip) from Canvas. In this task, we have 6 car attributes, and the label is the evaluation of the car. The attribute and label values are listed in the file "data-desc.txt". All the attributes are categorical. Please convert each categorical attribute into binary features. For example, for "safety: low, med, high", we convert it into three binary features: "safety" is "low" or not, "safety" is "med" or not, and "safety" is "high" or not. The training data are stored in the file "train.csv", consisting of $1,000$ examples. The test data are stored in "test.csv", and comprise 728 examples. In both training and test datasets, attribute values are separated by commas; the file "data-desc.txt" lists the attribute names in each column. To ensure numerical stability and avoid overfitting, we assign the feature weights a standard normal prior $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

   (a) [15 points] Implement MAP estimation algorithm for multi-class logistic regression model. To do so, you can calculate the gradient and feed it to some optimization package, say, L-BFGS. Report the prediction accuracy on the test data.

   We randomly initialize the weights to zeros and also from $\mathcal{N}(0, 1)$. The accuracy on the test set in both the cases is 89.82% which is higher than random guess.

   (b) [15 points] Let us use an "ugly" trick to convert the multi-class classification problem into a binary classification problem. Let us train four logistic regression models, where each model predicts one particular label, i.e., "unacc" or not, "acc" or not, "good" or not, and "vgood" or not. Then for each test example, we run the models to get four logistic scores, i.e., the probability that each label is one. We choose the label with the highest score as the final prediction. Report the prediction accuracy on the test data. As compared with multi-class logistic regression ,which one is better?

   We randomly initialize the weights to zeros. The accuracy on the test set is 82.26%. Hence, multiclass classification is better than multiple binary models.