Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Deeksha Arora
Roll Number: 20111017
Date: October 30, 2020

QUESTION 1

The given absolute loss regression problem with l_1 regularization is

$$\boldsymbol{w}_{opt} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \sum_{n=1}^{N} \left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right| + \lambda \|\boldsymbol{w}\|_1, where\ \|\boldsymbol{w}\|_1 = \sum_{d=1}^{D} |w_d|$$

Let $L_1(\boldsymbol{w}) = \sum_{n=1}^{N} \left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right|$ and $L_2(\boldsymbol{w}) = \lambda \|\boldsymbol{w}\|_1$

The function $L_1(\boldsymbol{w})$ is summation of N absolute value functions. We know that absolute value function is always convex and second derivative of convex functions is positive. So, the second derivative of sum of convex functions is also positive. Since, second derivative of sum of convex functions is positive, therefore, sum of two convex functions is also a convex function. Hence, function $L_1(\boldsymbol{w})$ is convex in nature.

The function $L_2(\boldsymbol{w})$ is absolute value function and hence convex.

Let $L(\boldsymbol{w})$ represents the regularized loss function, therefore

$$L(\boldsymbol{w}) = \sum_{n=1}^{N} \left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right| + \lambda \|\boldsymbol{w}\|_1$$

$$\Rightarrow L(\boldsymbol{w}) = L_1(\boldsymbol{w}) + L_2(\boldsymbol{w})$$

Since $L_1(\boldsymbol{w})$ and $L_2(\boldsymbol{w})$ are convex functions and we know that sum of two convex functions is also convex, so $L(\boldsymbol{w})$ is a ***convex function.***

Even though absolute value function is convex, it is non-differentiable. So we define subgradient for $L(\boldsymbol{w})$ as:

$$\partial_w L(\mathbf{w}) = \partial_w \left(\sum_{n=1}^{N} \left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right| + \lambda \|\boldsymbol{w}\|_1 \right)$$

From Sum Rule of Differentiation:

$$\partial_w L(\mathbf{w}) = \partial_w L_1(\boldsymbol{w}) + \partial_w L_2(\boldsymbol{w})$$

Sub Gradient of $L_1(\boldsymbol{w}) = \sum_{n=1}^{N} \left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right|$:

Let $l_n(\boldsymbol{w}) = \left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right|$. This function is non-differentiable at $\left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right| = 0$.

Let $\boldsymbol{u'}_n$ be the sub-gradient vector for $l_n(\boldsymbol{w})$, defined as:

$$\boldsymbol{u'}_n = \partial_w \left(l_n(\boldsymbol{w}) \right)$$

$$\boldsymbol{u'}_n = \begin{cases} \boldsymbol{x}_n & \text{if } y_n - \boldsymbol{w}^\top \boldsymbol{x}_n < 0, \\ -\boldsymbol{c}\boldsymbol{x}_n, where\ c \in [-1, +1] & \text{if } y_n - \boldsymbol{w}^\top \boldsymbol{x}_n = 0 \\ -\boldsymbol{x}_n & \text{if } y_n - \boldsymbol{w}^\top \boldsymbol{x}_n > 0, \end{cases} \tag{1}$$

Let $\boldsymbol{u}$ is the sub-gradient vector for $L_1(\boldsymbol{w}) = \sum_{n=1}^{N} \left| y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \right|$, defined as:

$$\begin{aligned} \boldsymbol{u} &= \partial_w L_1(\boldsymbol{w}) \\ &= \sum_{n=1}^{N} \partial_w \left(l_n(\boldsymbol{w}) \right) \\ &= \sum_{n=1}^{N} \boldsymbol{u'}_n \end{aligned} \tag{2}$$

where values of $\boldsymbol{u'}_n$ can be substituted from equation (1).

Sub-Gradient of $L_2(\boldsymbol{w}) = \lambda \|\boldsymbol{w}\|_1$, where $\|\boldsymbol{w}\|_1 = \sum_{d=1}^{D} |w_d|$:

It is absolute value function and is non-differentiable at 0 and hence we define subgradient for it. Let $\boldsymbol{v}$ be the sub-gradient vector for $L_2(\boldsymbol{w})$, then the i^{th} element of this sub gradient vector is given by:

$$v_i = \partial_{w_i} \left(\text{Ł}_2(\boldsymbol{w}) \right) = \begin{cases} -\lambda, & \text{if } w_i < 0, \\ [-\lambda, +\lambda], & \text{if } w_i = 0 \\ \lambda, & \text{if } w_i > 0, \end{cases} \tag{3}$$

Sub-Gradient of $L(\boldsymbol{w}) = L_1(\boldsymbol{w}) + L_2(\boldsymbol{w})$:

Let $\boldsymbol{g}$ denotes the sub-gradient vector of $L(\boldsymbol{w})$, then the i^{th} element of this vector is given by:

$$g_i = u_i + v_i \tag{4}$$

The values of u_i and v_i are substituted from equations (2) and (3) respectively.

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Deeksha Arora
Roll Number: 20111017
Date: October 30, 2020

QUESTION 2

The new squared loss function with masked inputs is defined as :

$$\begin{aligned} L(\mathbf{w}) &= \sum_{n=1}^{N} \left(y_n - \mathbf{w}^T \tilde{\mathbf{x}}_n\right)^2, \ where\ \tilde{\mathbf{x}}_\mathbf{n} = \mathbf{x_n} \circ \mathbf{m_n} \\ &= \sum_{n=1}^{N} \left(y_n^2 - y_n \mathbf{w}^T \tilde{\mathbf{x}}_\mathbf{n} - y_n \mathbf{w} \tilde{\mathbf{x}}_\mathbf{n}^T + \mathbf{w} \tilde{\mathbf{x}}_n^T \tilde{\mathbf{x}}_n \mathbf{w}^T\right) \end{aligned} \tag{5}$$

Taking the expectation w.r.t. $\mathbf{m_n}$ we get:

$$\begin{aligned} \mathrm{E}\left[L(\mathbf{w})\right] &= \sum_{n=1}^{N} \mathrm{E}\left[y_n^2 - y_n \mathbf{w}^T \tilde{\mathbf{x}}_n - y_n \mathbf{w} \tilde{\mathbf{x}}_n^T + \mathbf{w} \tilde{\mathbf{x}}_n^T \tilde{\mathbf{x}}_n \mathbf{w}^T\right] \\ &= \sum_{n=1}^{N} \mathrm{E}\left[y_n^2\right] - \mathrm{E}\left[y_n \mathbf{w}^T \tilde{\mathbf{x}}_n\right] - \mathrm{E}\left[y_n \mathbf{w} \tilde{\mathbf{x}}_n^T\right] + \mathrm{E}\left[\mathbf{w} \tilde{\mathbf{x}}_n^T \tilde{\mathbf{x}}_n \mathbf{w}^T\right] \\ &= \sum_{n=1}^{N} y_n^2 - y_n \mathbf{w}^T \mathrm{E}\left[\tilde{\mathbf{x}}_n\right] - y_n \mathbf{w} \mathrm{E}\left[\tilde{\mathbf{x}}_n^T\right] + \mathbf{w} \mathrm{E}\left[\tilde{\mathbf{x}}_n^T \tilde{\mathbf{x}}_n\right] \mathbf{w}^T \end{aligned} \tag{6}$$

The expected value of a matrix is the matrix of cell-wise expected values, so

$$\mathrm{E}\left[\tilde{\mathbf{x}}\right]_i = \mathrm{E}\left[(\mathbf{x} \circ \mathbf{m})_i\right] = \mathbf{x}_i \mathrm{E}\left[\mathbf{m}_i\right] = p\mathbf{x}_i$$

where p is the probability of retaining a feature.
Also,

$$\mathrm{Cov}(\tilde{\mathbf{x}}_n^T, \tilde{\mathbf{x}}_n) = \mathrm{E}\left[\tilde{\mathbf{x}}_n^T \tilde{\mathbf{x}}_n\right] - \mathrm{E}\left[\tilde{\mathbf{x}}_n^T\right] \mathrm{E}\left[\tilde{\mathbf{x}}_n\right] \tag{7}$$

$$\Rightarrow \mathrm{E}\left[\tilde{\mathbf{x}}_n^T \tilde{\mathbf{x}}_n\right] = \mathrm{Cov}(\tilde{\mathbf{x}}_n) + \mathbf{p}^2 \mathbf{x}_n^T \mathbf{x}_n \tag{8}$$

Substituting values of $\mathrm{E}\left[\tilde{\mathbf{x}_n}\right]$, $\mathrm{E}\left[\tilde{\mathbf{x}_n^T}\right]$ and $\mathrm{E}\left[\tilde{\mathbf{x}}_n^T \tilde{\mathbf{x}}_n\right]$ in equation (5) gives:

$$\mathrm{E}\left[L(\mathbf{w})\right] = \sum_{n=1}^{N} y_n^2 - p y_n \mathbf{w}^T \mathbf{x}_n - p y_n \mathbf{w} \mathbf{x}_n^T + \mathbf{w}^T \left(\mathrm{Cov}(\tilde{\mathbf{x}}_n) + \mathbf{p}^2 \mathbf{x}_n^T \mathbf{x}_n\right) \mathbf{w} \tag{9}$$

$$\Rightarrow \mathrm{E}\left[L(\mathbf{w})\right] = \sum_{n=1}^{N} \left(y_n - p\mathbf{w}^T \mathbf{x}_n\right)^2 + \mathbf{w}^T \mathrm{Cov}(\tilde{\mathbf{x}}_n) \mathbf{w} \tag{10}$$

where $\mathrm{Cov}\left(\tilde{\mathbf{x}}_n\right)$ is a diagonal matrix of size D x D with $p(p-1)\mathbf{x}_{nm}^2$ as diagonal elements, where m ranges from 1 to D.
Let,

$$\sum_{n=1}^{N} \mathrm{Cov}\left(\tilde{\mathbf{x}}_n\right) = \sum$$

Therefore,

$$\mathrm{E}\left[L\left(\mathbf{w}\right)\right] = \sum_{n=1}^{N} \left(y_n - p\mathbf{w}^T\mathbf{x}_n\right)^2 + \mathbf{w}^T \sum \mathbf{w}$$

This equation is similar to the expression of a regularised loss function with $\mathbf{w}^T \sum \mathbf{w}$ as the regularization term. Therefore, minimizing the expected value of $\sum_{n=1}^{N} \left(y_n - \mathbf{w}^T\tilde{\mathbf{x}}_n\right)^2$ is equivalent to minimizing a regularized loss function.

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Deeksha Arora
Roll Number: 20111017
Date: October 30, 2020

QUESTION 3

For the problem, $\{\hat{\mathbf{B}}, \hat{\mathbf{S}}\} = \arg\min_{\mathbf{B},\mathbf{S}} \text{TRACE}\left[(\mathbf{Y} - \mathbf{XBS})^\top (\mathbf{Y} - \mathbf{XBS})\right]$

The expression for loss function can be written as:

$$L(\mathbf{B},\mathbf{S}) = \text{TRACE}\left[(\mathbf{Y} - \mathbf{XBS})^T (\mathbf{Y} - \mathbf{XBS})\right]$$

$$L(\mathbf{B},\mathbf{S}) = \text{TRACE}\left[\left(\mathbf{Y}^T - \mathbf{S}^T\mathbf{B}^T\mathbf{X}^T\right)(\mathbf{Y} - \mathbf{XBS})\right]$$

$$L(\mathbf{B},\mathbf{S}) = \text{TRACE}\left[\left(\mathbf{Y}^T\mathbf{Y} - \mathbf{Y}^T\mathbf{XBS} - \mathbf{S}^T\mathbf{B}^T\mathbf{X}^T\mathbf{Y} + \mathbf{S}^T\mathbf{B}^T\mathbf{X}^T\mathbf{XBS}\right)\right]$$

$$L(\mathbf{B},\mathbf{S}) = \text{TRACE}\left[\mathbf{Y}^T\mathbf{Y}\right] - \text{TRACE}\left[\mathbf{Y}^T\mathbf{XBS}\right] - \text{TRACE}\left[\mathbf{S}^T\mathbf{B}^T\mathbf{X}^T\mathbf{Y}\right] + \text{TRACE}\left[\mathbf{S}^T\mathbf{B}^T\mathbf{X}^T\mathbf{XBS}\right]$$

Keeping $\boldsymbol{S}$ constant and differentiating L(B,S) with respect to B, we get:

$$\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{B}} = 0 - \left(\mathbf{Y}^T\mathbf{X}\right)^T\mathbf{S}^T - \left(\mathbf{SY}^T\mathbf{X}\right)^T + \left(\left(\mathbf{SS}^T\mathbf{B}^T\mathbf{X}^T\mathbf{X}\right)^T + \left(\mathbf{S}^T\mathbf{B}^T\mathbf{X}^T\mathbf{X}\right)^T\mathbf{S}^T\right)$$

$$\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{B}} = 0 - \mathbf{X}^T\mathbf{YS}^T - \mathbf{X}^T\mathbf{YS}^T + \left(\mathbf{X}^T\mathbf{XBSS}^T + \mathbf{X}^T\mathbf{XBSS}^T\right)$$

$$\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{B}} = 2\left(\mathbf{X}^T\mathbf{XBSS}^T\right) - 2\left(\mathbf{X}^T\mathbf{YS}^T\right)$$

Put $\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{B}} = 0$. Therefore,

$$\left(\mathbf{X}^T\mathbf{XBSS}^T\right) = \left(\mathbf{X}^T\mathbf{YS}^T\right)$$

$$\Rightarrow \mathbf{B} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\left(\mathbf{X}^T\mathbf{YS}^T\right)\left(\mathbf{SS}^T\right)^{-1}$$

Keeping $\boldsymbol{B}$ constant and differentiating L($\boldsymbol{B}$,$\boldsymbol{S}$) with respect to $\boldsymbol{S}$, we get:

$$\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{S}} = 0 - \mathbf{B}^T\mathbf{X}^T\mathbf{Y} - \left(\mathbf{Y}^T\mathbf{XB}\right)^T + \left(\left(\mathbf{S}^T\mathbf{B}^T\mathbf{X}^T\mathbf{XB}\right)^T + \left(\mathbf{B}^T\mathbf{X}^T\mathbf{XBS}\right)\right)$$

$$\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{S}} = 0 - \mathbf{B}^T\mathbf{X}^T\mathbf{Y} - \mathbf{B}^T\mathbf{X}^T\mathbf{Y} + \left(\mathbf{B}^T\mathbf{X}^T\mathbf{XBS} + \mathbf{B}^T\mathbf{X}^T\mathbf{XBS}\right)$$

$$\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{S}} = 2\left(\mathbf{B}^T\mathbf{X}^T\mathbf{XBS}\right) - 2\left(\mathbf{B}^T\mathbf{X}^T\mathbf{Y}\right)$$

Put $\frac{\partial L(\mathbf{B},\mathbf{S})}{\partial \mathbf{S}} = 0$. Therefore,

$$\left(\mathbf{B}^T\mathbf{X}^T\mathbf{XBS}\right) = \left(\mathbf{B}^T\mathbf{X}^T\mathbf{Y}\right)$$

$$\mathbf{S} = \left(\mathbf{B}^T\mathbf{X}^T\mathbf{XB}\right)^{-1}\left(\mathbf{B}^T\mathbf{X}^T\mathbf{Y}\right)$$

$$\Rightarrow \mathbf{S} = \left(\left(\mathbf{XB}\right)^T\left(\mathbf{XB}\right)\right)^{-1}\left(\mathbf{XB}\right)^T\mathbf{Y}$$

ALT-OPT algorithm to learn $\boldsymbol{B}$ and $\boldsymbol{S}$ for the optimization problem $\{\hat{\mathbf{B}}, \hat{\mathbf{S}}\} = \arg\min_{\mathbf{B},\mathbf{S}} \text{TRACE}\left[(\mathbf{Y} - \mathbf{XBS})^\top (\mathbf{Y} - \mathbf{XBS})\right]$ is as follows:

1. Initialize one the variables. Ex: $\boldsymbol{S} = \boldsymbol{S}^{(0)}$, t=0
2. $\mathbf{B}^{(t+1)} = \arg\min_{\mathbf{B}} \mathcal{L}\left(\mathbf{B}, \mathbf{S}^{(t)}\right)$
$$= \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\left(\mathbf{X}^T\mathbf{Y}\mathbf{S}^{(\mathbf{t})^T}\right)\left(\mathbf{S}^{(\mathbf{t})}\mathbf{S}^{(\mathbf{t})^T}\right)^{-1}$$
3. $\mathbf{S}^{(t+1)} = \arg\min_{\mathbf{S}} \mathcal{L}\left(\mathbf{B}^{(t+1)}, \mathbf{S}\right)$
$$= \left(\left(\mathbf{XB}^{(\mathbf{t+1})}\right)^T\left(\mathbf{XB}^{(\mathbf{t+1})}\right)\right)^{-1}\left(\mathbf{XB}^{(\mathbf{t+1})}\right)^T\mathbf{Y}$$
4. $t = t + 1$. Go to step 2, if not converged yet.

To compute matrix $\boldsymbol{B}$, we need to compute inverse of two matrices, one is $\left(\mathbf{X}^T\mathbf{X}\right)$ which is a $\boldsymbol{D}$ x $\boldsymbol{D}$ matrix and the other is $\left(\mathbf{SS}^T\right)$ which is $\boldsymbol{K}$ x $\boldsymbol{K}$ matrix. On the other hand, to compute $\boldsymbol{S}$, we need to find inverse of $\left((\mathbf{XB})^T(\mathbf{XB})\right)$ which is a $\boldsymbol{K}$ x $\boldsymbol{K}$ matrix. The computation of inversion of a $\boldsymbol{K}$ x $\boldsymbol{K}$ matrix is common in both the subproblems, but subproblem for $\boldsymbol{B}$ requires an additional inversion of a $\boldsymbol{D}$ x $\boldsymbol{D}$ matrix which makes computation of matrix $\boldsymbol{B}$ harder than that of matrix $\boldsymbol{S}$. However, the matrix $\left(\mathbf{X}^T\mathbf{X}\right)$ is same for all iterations of ALT-OPT algorithm. If we assume that inversion of $\left(\mathbf{X}^T\mathbf{X}\right)$ can be precomputed and cached then we don't need to solve it in every iteration. Hence, if the inversion of $\left(\mathbf{X}^T\mathbf{X}\right)$ **is cached**, solving the subproblem for $\boldsymbol{B}$ becomes **equally hard** as solving the subproblem for $\boldsymbol{S}$. However, if the inversion of $\left(\mathbf{X}^T\mathbf{X}\right)$ **is not cached**, then solving the subproblem for $\boldsymbol{B}$ is **harder** than solving the subproblem for $\boldsymbol{S}$.

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Deeksha Arora
Roll Number: 20111017
Date: October 30, 2020

QUESTION 4

In Newton's method at each point the quadratic (second-order) approximation of L($\boldsymbol{w}$) is minimized. So,

$$\begin{aligned}\boldsymbol{w}^{(t+1)} &= \arg\min_{\boldsymbol{w}} \left[L\left(\boldsymbol{w}^{(t)}\right) + \nabla L\left(\boldsymbol{w}^{(t)}\right)^{\top}\left(\boldsymbol{w}-\boldsymbol{w}^{(t)}\right) + \frac{1}{2}\left(\boldsymbol{w}-\boldsymbol{w}^{(t)}\right)^{\top}\nabla^2 L\left(\boldsymbol{w}^{(t)}\right)\left(\boldsymbol{w}-\boldsymbol{w}^{(t)}\right)\right] \\ &= \boldsymbol{w}^{(t)} - \left(\nabla^2 L\left(\boldsymbol{w}^{(t)}\right)\right)^{-1}\nabla L\left(\boldsymbol{w}^{(t)}\right) \\ &= \boldsymbol{w}^{(t)} - \left(\boldsymbol{H^{(t)}}\right)^{-1}\boldsymbol{g}^{(t)} \end{aligned} \tag{11}$$

where $\boldsymbol{g}^{(t)} = \nabla L\left(\boldsymbol{w}^{(t)}\right)$ represents the gradient vector for the t$^{\text{th}}$ iteration and $\boldsymbol{H}^{(t)} = \nabla^2 L\left(\boldsymbol{w}^{(t)}\right)$ represents the Hessian matrix for the t$^{\text{th}}$ iteration.

Ridge regression problem is defined as:

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \frac{1}{2}(\boldsymbol{y}-\mathbf{X}\boldsymbol{w})^{\top}(\boldsymbol{y}-\mathbf{X}\boldsymbol{w}) + \frac{\lambda}{2}\boldsymbol{w}^{\top}\boldsymbol{w}$$

where $L(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{y}-\mathbf{X}\boldsymbol{w})^{\top}(\boldsymbol{y}-\mathbf{X}\boldsymbol{w}) + \frac{\lambda}{2}\boldsymbol{w}^{\top}\boldsymbol{w}$

Differentiating $L(\boldsymbol{w})$ with respect to $\boldsymbol{w}$ we get:

$$\begin{aligned}\nabla L(\boldsymbol{w}) &= \frac{1}{2}\nabla\left((\boldsymbol{y}-\mathbf{X}\boldsymbol{w})^{\top}(\boldsymbol{y}-\mathbf{X}\boldsymbol{w}) + \lambda\boldsymbol{w}^{\top}\boldsymbol{w}\right) \\ &= \frac{1}{2}\left(2\mathbf{X}^T\mathbf{X}\boldsymbol{w} - 2\mathbf{X}^T\boldsymbol{y} + 2\lambda\boldsymbol{w}\right) \\ &= \mathbf{X}^T\mathbf{X}\boldsymbol{w} - \mathbf{X}^T\boldsymbol{y} + \lambda\boldsymbol{w}\end{aligned} \tag{12}$$

Differentiating $\nabla L(\boldsymbol{w})$ with respect to $\boldsymbol{w}$ we get:

$$\begin{aligned}\nabla^2 L(\boldsymbol{w}) &= \nabla\left(\mathbf{X}^T\mathbf{X}\boldsymbol{w} - \mathbf{X}^T\boldsymbol{y} + \lambda\boldsymbol{w}\right) \\ &= \mathbf{X}^{\top}\mathbf{X} + \lambda\mathbf{I}\end{aligned} \tag{13}$$

Therefore, the gradient vector and the Hessian matrix for the t$^{\text{th}}$ iteration are given by:

$$\boldsymbol{g}^{(t)} = -\boldsymbol{X}^{\top}(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}^{(t)}) + \lambda\boldsymbol{w}^{(t)} \tag{14}$$

$$\boldsymbol{H}^{(t)} = \boldsymbol{X}^{\top}\boldsymbol{X} + \lambda\boldsymbol{I} \tag{15}$$

Initializing $\boldsymbol{w}$ as $\boldsymbol{w^{(0)}}$ and substituting values of $\boldsymbol{w^{(t)}}$, gradient vector and Hessian matrix in equation (11) gives:

$$\begin{aligned}\Rightarrow \boldsymbol{w}^{(1)} &= \boldsymbol{w}^{(0)} - \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1}\left[-\boldsymbol{X}^T\mathbf{y} + \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}^{(t)} + \lambda\boldsymbol{w}^{(0)}\right] \\ &= \boldsymbol{w}^{(0)} - \left(\boldsymbol{X}^T X + \lambda\boldsymbol{I}\right)^{-1}\left[-\boldsymbol{X}^T\mathbf{y} + \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)\boldsymbol{w}^{(0)}\right] \\ &= \boldsymbol{w}^{(0)} + \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X}^T\mathbf{y} - \boldsymbol{w}^{(0)} \\ &= \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X}^T\mathbf{y}\end{aligned} \tag{16}$$

The expression obtained for $\boldsymbol{w}^{(1)}$ is independent of the iteration number t. Therefore, the expression for $\boldsymbol{w}^{(2)}$ will be same as that of $\boldsymbol{w}^{(1)}$ i.e.

$$\boldsymbol{w}^{(2)} = \boldsymbol{w}^{(1)} = \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X}^T\mathbf{y} \tag{17}$$

Hence, Newton's method took 2 iterations to converge, where the algorithm is run for the second iteration to identify convergence.

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Deeksha Arora
Roll Number: 20111017
Date: October 30, 2020

QUESTION 5

Likelihood:
The given problem is of a categorical distribution. For N dice rolls, $\boldsymbol{X} = \{x_1, \ldots, x_N\}$, where $x_i \in \{1, \ldots, 6\}$. So, the Multinoulli distribution is an appropriate choice for the likelihood.

$$p\left(\boldsymbol{X} \mid \boldsymbol{\pi}\right) = \text{ multinoulli } \left(\boldsymbol{X} \mid \boldsymbol{\pi}\right) \tag{18}$$

If we assume that the data is independent and identically distributed, then likelihood has the form:

$$p\left(\boldsymbol{X} \mid \boldsymbol{\pi}\right) = \prod_{k=1}^{6} \pi_k^{N_k} \tag{19}$$

where $\boldsymbol{\pi}$ is the probability vector such that $\pi_k = p\left(\boldsymbol{X} = k\right)$ and $N_k = \sum_{i=1}^{N} \mathbb{I}\left(y_i = k\right)$ is the number of times event k occurred.
Log-Likelihood:

$$\log(p\left(\boldsymbol{X}|\boldsymbol{\pi}\right)) = \sum_{k=1}^{6} N_k \log(\pi_k)$$

Prior distribution :
The probability vector $\boldsymbol{\pi} = [\pi_1, \pi_2, ...\pi_6]$ is a K-dimensional vector. So, the Dirichlet distribution is the appropriate prior for this problem, given by:

$$p\left(\boldsymbol{\pi} \mid \boldsymbol{\alpha}\right) = \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{6} \pi_i^{\alpha_i - 1} \tag{20}$$

where the normalizing constant $B(\boldsymbol{\alpha})$ is a multivariate beta function, given by:

$$\mathrm{B}(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{N} \Gamma\left(\alpha_i\right)}{\Gamma\left(\sum_{i=1}^{N} \alpha_i\right)}, \quad \boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_6] \tag{21}$$

Log-Prior (without the constant term):

$$\log(p\left(\boldsymbol{\pi} \mid \boldsymbol{\alpha}\right)) = \sum_{k=1}^{6} (\alpha_k - 1) \log(\pi_k)$$

MAP Estimate:
For derivation of MAP estimate, we need to use a Lagrange multiplier to enforce the equality constraint that $\sum_{k=1}^{6} \pi_k = 1$. The constrained objective function or the Lagrangian is given by the log likelihood plus log prior plus the constraint:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\pi}, \lambda) &= \sum_{k=1}^{6} N_k \log \pi_k + \sum_{k=1}^{6} (\alpha_k - 1) \log \pi_k + \lambda \left(1 - \sum_{k=1}^{6} \pi_k\right) \\
&= \sum_{k=1}^{6} (N_k + \alpha_k - 1) \log \pi_k + \lambda \left(1 - \sum_{k=1}^{6} \pi_k\right)
\end{aligned} \tag{22}$$

Taking derivative of $\mathcal{L}(\boldsymbol{\pi}, \lambda)$ with respect to λ gives:

$$\frac{\partial \mathcal{L}}{\partial \lambda} = \left(1 - \sum_{k=1}^{6} \pi_k\right) = 0 \tag{23}$$

Taking derivative of $\mathcal{L}(\boldsymbol{\pi}, \lambda)$ with respect to π_k gives:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \pi_k} &= \frac{N_k + \alpha_k - 1}{\pi_k} - \lambda = 0 \\ \Rightarrow N_k + \alpha_k - 1 &= \lambda \, \pi_k \end{aligned} \tag{24}$$

Using the sum-to-one constraint to solve for λ:

$$\begin{aligned} \sum_k N_k + \alpha_k - 1 &= \lambda \sum_k \pi_k \\ \Rightarrow \lambda &= N + \alpha_0 - 6 \end{aligned} \tag{25}$$

where $\alpha_0 \triangleq \sum_{k=1}^{6} \alpha_k$ is the equivalent sample size of the prior. Substituting value of λ in equation (24) gives:

$$\hat{\pi}_k = \frac{N_k + \alpha_k - 1}{N + \alpha_0 - 6} \tag{26}$$

which is the required MAP estimate.
MAP solution is expected to be better than MLE solution in the following cases:

1. When number of training examples is less. In this case, the MLE solution tends to overfit and is often unreliable. On the other hand, MAP solution will give more accurate results beacuse it uses the regularizer term which prevents overfitting.

2. When the training data contains examples from a specific class. For example, considering the dice roll problem where in every roll, the dice shows face 6. In this case, the MLE estimate will give $\pi_6 = 1$, and the probability for all other faces is 0. MAP solution would give more reliable estimation in this case.

Full posterior distribution:
The posterior distribution is given by:

$$\begin{aligned} p(\boldsymbol{\pi} \mid \boldsymbol{X}, \boldsymbol{\alpha}) &= \frac{p(\boldsymbol{X} \mid \boldsymbol{\pi}) * p(\boldsymbol{\pi} \mid \boldsymbol{\alpha})}{p(\boldsymbol{X})} \\ &= \frac{\prod_{k=1}^{6} \pi_k^{N_k} * \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{6} \pi_i^{\alpha_i - 1}}{\int (\prod_{k=1}^{6} \pi_k^{N_k} * \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{6} \pi_i^{\alpha_i - 1}) d\boldsymbol{\pi}} \end{aligned} \tag{27}$$

Since the denominator is constant, the posterior can be represented as:

$$\begin{aligned} p(\boldsymbol{\pi} \mid \boldsymbol{X}, \boldsymbol{\alpha}) &\propto p(\boldsymbol{X} \mid \boldsymbol{\pi}) p(\boldsymbol{\pi} \mid \boldsymbol{\alpha}) \\ &\propto \prod_{k=1}^{6} \pi_k^{N_k} \pi_k^{\alpha_k - 1} \\ &\propto \prod_{k=1}^{6} \pi_k^{\alpha_k + N_k - 1} \\ &= \mathrm{Dirichlet}\left(\boldsymbol{\pi} \mid \alpha_1 + N_1, \ldots, \alpha_6 + N_6\right) \end{aligned} \tag{28}$$

Yes, given the above posterior we can get the MLE and MAP estimate without solving the MLE and MAP optimization problems.

MAP estimate from the posterior:

The MAP estimate will be the mode of the above Dirichlet distribution and is given by:

$$\hat{\pi}_k(MAP) = \frac{N_k + \alpha_k - 1}{N + \alpha_0 - 6} \tag{29}$$

MLE estimate from the posterior:

When the prior is uniform, MAP estimate is equivalent to MLE estimate. Therefore, to get the MLE solution, we need to make the prior uniform. This can be done by equating each component of $\boldsymbol{\alpha}$ to 1 and equation (29) will give the MLE solution as:

$$\hat{\pi}_k(MLE) = \frac{N_k}{N} \tag{30}$$