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

Student Name: Jinang Shah
Roll Number: 170649
Date: October 30, 2020

QUESTION 1

The optimisation problem for the absolute loss regression problem with l_1 regularisation is,

$$w_{opt} = argmin_w \sum_{n=1}^{N} |y_n - w^T x_n| + \lambda \|w\|_1; \|w\|_1 = \sum_{d=1}^{D} |w_d|$$

Here, D is the number of dimensions for input x_n; N is the total number of available training samples; $\lambda > 0$ is the regularization parameter.

Here, the objective function is, $L_r = \sum_{n=1}^{N} |y_n - w^T x_n| + \lambda \|w\|_1$

For the first part, **yes**, the given objective function is **convex**. Rather than going into full detailed proof, let us understand this with the function that is simple but has similar characteristics as L_r, $f(x) = |x|$; here, x is scalar.

Since, $f(x)$ is non differentiable at $x = 0$, we can't confirm its convexity through second derivative, as it is 0 everywhere else. Now, we know that for any two points x_1, x_2 in R,
$|\alpha x_1 + (1-\alpha)x_2| \leqslant |\alpha x_1| + |(1-\alpha)x_2| \implies f(\alpha x_1 + (1-\alpha)x_2) \leqslant f(\alpha x_1) + f((1-\alpha)x_2); \alpha > 0$
Above relation i.e. **jensen's inequality** is also an informal definition for convexity, which conveys that $f(x) = |x|$ is a convex function.

Breaking L_r into two sub functions, absolute loss function, and l_1 regularisation function. Individually, both are just multi-dimensional extension of $f(x)$, which does not affect a function's characteristics. Thus, since both the sub functions of L_r share the same nature of convexity, it is safe to imply that the function L_r is convex in nature too.

For the second part, $x_n = [x_{n1}, ..., x_{nd}, ..., x_{nD}]^T; w = [w_1, ..., w_d, ..., w_D]^T$

Sub gradient of L_r w.r.t. w_d is, $\frac{\partial L_r}{\partial w_d} = \sum_{n=1}^{N} -\alpha_n x_{nd} + \lambda \beta_d$

where,
$\alpha_n = sign(y_n - w^T x_n)$ if $y_n - w^T x_n \neq 0$, else c_1 where $c_1 \in [-1, 1]$
$\beta_d = sign(w_d)$ if $w_d \neq 0$, else c_2 where $c_2 \in [-1, 1]$

Thus, sub gradient of L_r w.r.t. w is,

$\frac{\partial L_r}{\partial w} = [\frac{\partial L_r}{\partial w_1}, ..., \frac{\partial L_r}{\partial w_d}, ..., \frac{\partial L_r}{\partial w_D}]^T$

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

Student Name: Jinang Shah
Roll Number: 170649
Date: October 30, 2020

QUESTION 2

Here,
for input $x_n = [x_{n1}, ..., x_{nd}, ..., x_{nD}]^T$, weight vector $w = [w_1, ..., w_d, ..., w_D]^T$,
and mask vector $m_n = [m_{n1}, ..., m_{nd}, ..., m_{nD}]^T$;
where m_{nd} follows distribution of **Bernoulli(p)** i.e. $P(m = 1) = p, P(m = 0) = 1-p; 0 \leq p \leq 1$

$E[m_{nd}] = 1p + 0(1-p) = p; E[m_{nd}^2] = 1p + 0(1-p) = p \implies E[m] = E[m^2] = p$
$Var(m_{nd}) = E[m_{nd}^2] - E[m_{nd}]^2 = p - p^2 = p(1-p)$

Given, loss function $L = \sum_{n=1}^{N}(y_n - w^T\tilde{x}_n)^2$; Since, y_n has the same shape as $w^T\tilde{x_n}$, which is 1X1, we can assume them to be a scalar quantity. Now, we can write our loss function L in the following way: $L = \sum_{n=1}^{N} y_n^2 - 2(y_n)(w^T\tilde{x}_n) + (w^T\tilde{x}_n)^2$

$E[L] = E[\sum_{n=1}^{N} y_n^2 - 2(y_n)(w^T\tilde{x}_n) + (w^T\tilde{x}_n)^2] = \sum_{n=1}^{N} E[y_n^2] - E[2(y_n)(w^T\tilde{x}_n)] + E[(w^T\tilde{x}_n)^2]$

$E[L] = \sum_{n=1}^{N} y_n^2 - 2(y_n)E[(w^T\tilde{x}_n)] + E[(w^T\tilde{x}_n)^2]$; Since, y_n is a known and constant scalar

$E[\tilde{x}_n] = E[x_n.m_n] = x_n E[m_n] = px_n; E[\tilde{x}_n^2] = E[x_n^2.m_n^2] = x_n^2 E[m_n^2] = px_n^2$
$Var(\tilde{x}_{nd}) = Var(x_{nd}m_{nd}) = x_{nd}^2 Var(m_{nd}) = x_{nd}^2 p(1-p)$

$E[w^T\tilde{x}_n] = E[\sum_{d=1}^{D} w_d\tilde{x}_{nd}] = \sum_{d=1}^{D} E[w_d\tilde{x}_{nd}] = \sum_{d=1}^{D} w_d(px_{nd}) = pw^T x_n$

$E[(w^T\tilde{x}_n)^2] = Var(w^T\tilde{x}_n) + E[w^T\tilde{x}_n]^2 = Var(\sum_{d=1}^{D} w_d\tilde{x}_{nd}) + (pw^T x_n)^2$
$E[(w^T\tilde{x}_n)^2] = \sum_{d=1}^{D} w_d^2 Var(\tilde{x}_{nd}) + (pw^T x_n)^2 = \sum_{d=1}^{D} w_d^2 x_{nd}^2 Var(m_{nd}) + (pw^T x_n)^2$

$E[(w^T\tilde{x}_n)^2] = p(1-p)w^{2^T} x_n^2 + (pw^T x_n)^2$; Here, $w^{2^T} x_n^2 = \sum_{d=1}^{D} w_d^2 x_{nd}^2$

Putting all the above derivation into the derived equation for $E[L]$,
$E[L] = \sum_{n=1}^{N} y_n^2 - 2(y_n)E[(w^T\tilde{x}_n)] + E[(w^T\tilde{x}_n)^2]$
$E[L] = \sum_{n=1}^{N} y_n^2 - 2(y_n)(pw^T x_n) + p(1-p)w^{2^T} x^2 + (pw^T x_n)^2$
$E[L] = \sum_{n=1}^{N}(y_n - pw^T x_n)^2 + p(1-p)w^{2^T} x^2$

$E[L] = \sum_{n=1}^{N}(y_n - pw^T x_n)^2 + p(1-p)\sum_{n=1}^{N}\sum_{d=1}^{D} w_d^2 x_{nd}^2$

Due to the term other than the normal mean squared error loss in the above expression of the expected loss function, it quite resembles with the regularised loss function. And minimizing, the given loss function, will be in the end, equivalent to minimizing the regularised loss function.

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

Student Name: Jinang Shah
Roll Number: 170649
Date: October 30, 2020

QUESTION 3

Here, we are solving multi-output regression problem, with squared loss function, also defined as $Tr[(Y - XW)^T(Y - XW)]$; Here, Y:NxM, X:NxD, W:DxM, $Tr[x] = TRACE[x]$. Furthermore, we divide $W = BS$; with B:DxK, S:KxM, $K < min\{D, M\}$.

$L = Tr[(Y - XW)^T(Y - XW)]$
$L = Tr[Y^TY] - Tr[Y^TXW] - Tr[W^TX^TY] + Tr[W^TX^TXW]$

Standard operations, $\nabla_W Tr[V^TV] = 0; \nabla_W Tr[V^TWH] = VH^T; \nabla_W Tr[H^TW^TV] = VH^T$
$\nabla_W Tr[H^TW^TWH] = \nabla_W Tr[WHH^TW^T] = W((HH^T)^T + HH^T) = 2WHH^T$
$\nabla_W Tr[H^TW^TV^TVWH] = 2V^TVWHH^T$ (using chain rule on the above expression)

Putting $W = BS$, $L = Tr[Y^TY] - Tr[Y^TXBS] - Tr[S^TB^TX^TY] + Tr[S^TB^TX^TXBS]$

Since, we can't solve for both B, S simultaneously, we'll solve them separately. The standard operations defined above should be sufficient for the following calculations.

First for $B, \nabla_B L = 0$, given S,
$\nabla_B L = 0 - X^TYS^T - X^TYS^T + 2X^TXBSS^T = 0 \implies X^TXBSS^T = X^TYS^T$

$B = (X^TX)^{-1}X^TYS^T(SS^T)^{-1}$

Now, for S, $\nabla_S L = 0$, given B,
$\nabla_S L = 0 - B^TX^TY - B^TX^TY + 2B^TX^TXBS = 0 \implies B^TX^TXBS = B^TX^TY$

$S = (B^TX^TXB)^{-1}B^TX^TY$

If noticed, $W = BS = (X^TX)^{-1}X^TY$, analytical output of L if W were not to be broken further into B, S. W has one definitive answer for this problem, but the results for B, S suggests that there exist infinitely many possibilities for B, S such that $W = BS$.

First, initialising S, then calculating B with its derived expression, and then other way around with the derived expression for S. This will further continue until convergence. This is an alternating optimization algorithm to learn B, S. We can also use gradient descent here instead of a direct analytical calculation.

In both of the expressions for B and S, there is a need for calculating inverse of a matrix, which can be quite expensive computationally. For B, we will need to calculate inverse for two times, whereas for S, it will be for one time. So, apart from the said differences, sub problems for both B and S, seems to be more or less equally computationally expensive, but since we don't have to apply gradient approach, both seems to be equally easy in the sense that we can calculate them directly analytically.

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

Student Name: Jinang Shah
Roll Number: 170649
Date: October 30, 2020

QUESTION 4

Optimisation using Newton's Method, for the given loss function $L(w)$ and w^t,
$J(w) = L(w^t) + \nabla L(w^t)^T(w - w^t) + \frac{1}{2}(w - w^t)^T\nabla^2 L(w^t)(w - w^t)$

$w^{t+1} = argmin_w J(w)$

At minima of $J(w)$, $\nabla J(w) = 0 \implies \nabla L(w^t) + \frac{1}{2}2\nabla^2 L(w^t)(w - w^t) = 0$
$\implies \nabla^2 L(w^t)(w - w^t) = -\nabla L(w^t) \implies w = w^t - (\nabla^2 L(w^t))^{-1}\nabla L(w^t)$

$w^{t+1} = w^t - (\nabla^2 L(w^t))^{-1}\nabla L(w^t) = w^t - (H^t)^{-1}g^t$, for $H^t = \nabla^2 L(w^t), g^t = \nabla L(w^t)$

Given loss function $L(w^t) = \frac{1}{2}(y - Xw^t)^T(y - Xw^t) + \frac{\lambda}{2}{w^t}^T w^t$; here w:Dx1; X:NxD; y:Nx1

$\nabla L(w^t) = g^t = -X^T(y - Xw^t) + \lambda w^t = (X^TX + \lambda I)w^t - X^Ty$
$\nabla^2 L(w^t) = H^t = X^TX + \lambda I$

Furthermore, from the derived expressions of g^t and H^t, we can also write g^t as,
$g^t = H^t w^t - X^Ty \implies (H^t)^{-1}g^t = (H^t)^{-1}(H^t w^t - X^Ty) = w^t - (H^t)^{-1}X^Ty$

Using the above derived relation for w^{t+1},
$w^{t+1} = w^t - (H^t)^{-1}g^t = w^t - (w^t - (H^t)^{-1}X^Ty) = (H^t)^{-1}X^Ty$

$w^{t+1} = (H^t)^{-1}X^Ty$, here $H^t = \nabla^2 L(w^t)$

Since, the derived update equation of newton's method for the ridge regression is a closed form solution, the model will take **only 1 iteration** to converge for a particular w^t.

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

Student Name: Jinang Shah
Roll Number: 170649
Date: October 30, 2020

QUESTION 5

We have a six faced dice, which has been rolled N times, out of which the numbers 1,2,3,4,5,6 were observed $N_1, N_2, N_3, N_4, N_5, N_6$ times respectively. Assuming that probability for dice showing to be k^{th} face to be equal to $\pi_k \in (0, 1)$. Here, $k = \{1, 2, 3, 4, 5, 6\}$.

We have our probability vector $\pi = [\pi_1, \pi_2, ..., \pi_6]$. Also, $\sum_{k=1}^{6} \pi_k = 1; \sum_{k=1}^{6} N_k = N$.

We assume our likelihood to follow **multinomial distribution**, with parameters N_k, whereas our class prior to follow **dirichlet distribution**, with parameters α_k for $k = \{1, 2, 3, 4, 5, 6\}$.

Here, assuming $\sum_{k=1}^{6} \alpha_k = \alpha$, and filtering out constant values from the following optimisation problem for MAP estimate, and adding lagrange constraint, we get,

$\pi_{MAP} = argmin_\pi - \sum_{k=1}^{6} N_k log(\pi_k) - \sum_{k=1}^{6} (\alpha_k - 1) log(\pi_k) + \lambda((\sum_{k=1}^{6} \pi_k) - 1)$

$\pi_{MAP} = argmin_\pi - \sum_{k=1}^{6} (N_k + \alpha_k - 1) log(\pi_k) + \lambda((\sum_{k=1}^{6} \pi_k) - 1)$

Differentiating the function w.r.t. π_k and equating to 0,
$-\frac{N_k + \alpha_k - 1}{\pi_k} + \lambda = 0 \implies \lambda \pi_k = N_k + \alpha_k - 1$; assuming $\pi_k \in (0, 1)$

Taking sum over i on both sides i.e. multiplying with the operator $\sum_{k=1}^{6}$,
$\sum_{k=1}^{6} \lambda \pi_k = \sum_{k=1}^{6} N_k + \alpha_k - 1 \implies \lambda = N + \alpha - 6$

Using the value of λ in the above equation, we get,
$\pi_{k_{MAP}} = \frac{N_k + \alpha_k - 1}{N + \alpha - 6}$; which also satisfies the condition, $\sum_{k=1}^{6} \pi_k = 1$

So, $\pi_{MAP} = [\pi_{1_{MAP}}, \pi_{2_{MAP}}, ..., \pi_{6_{MAP}}]$

MAP vs MLE: In general, when the prior follows uniform distribution, the MAP and MLE estimates will be identical, but in any other scenario this won't hold. MLE does not take into account the prior information, if available. Furthermore, we can say that aside from the case of uniform distribution, if we have a prior information then the MAP will be better estimate than MLE. So here if $\alpha_k = 0$ for all possible k, then we will have same MLE and MAP estimates, but in any other case, both will be different. If we have a confident prior belief, which is not uniform distribution, then MAP estimate will be better than MLE.

Full posterior over π: $p(\pi/y) = \frac{p(\pi)p(y/\pi)}{p(y)}$; Here, as used in MAP, $p(y/\pi)$ follows multinomial, whereas $p(\pi)$, prior follows dirichlet distribution.

Assuming $p(y)$ to be a constant for finding the distribution of the posterior,

$p(\pi/y) \propto p(\pi)p(y/\pi) \implies p(\pi/y) \propto (C_1 \prod_{k=1}^{6} \pi_k^{N_k})(C_2 \prod_{k=1}^{6} \pi_k^{\alpha_k - 1}) \implies p(\pi/y) \propto \prod_{k=1}^{6} \pi_k^{N_k + \alpha_k - 1}$

The above derived equation suggests that the posterior over probability vector π follows the **dirichlet distribution** with parameters $N_k + \alpha_k$, for $k = \{1, 2, 3, 4, 5, 6\}$.

The π for which the $p(\pi/y)$ achieves its maximum, will be its MAP estimation. Now, if we put value of each α_k to be 0 in our obtained MAP estimate, then we will have a uniform distribution as prior, for which we know that the MAP and MLE estimates will be identical, so that's our MLE estimate. So, yes, we can calculate our MLE and MAP estimates with the obtained posterior, without needing to calculate them separately.