Sheet 4

Q1) For the linear regression problem, derive the Hessian matrix of the loss function. Comment on the convexity of such a function.

$$\mathbf{H} = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1\,\partial x_n} \\[2.2ex] \dfrac{\partial^2 f}{\partial x_2\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2\,\partial x_n} \\[2.2ex] \vdots & \vdots & \ddots & \vdots \\[2.2ex] \dfrac{\partial^2 f}{\partial x_n\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

$$L = (1/2N) * \|y - y(x, w)\|^2$$
$$L = (1/2N) * \|y - Xw\|^2$$
$$L = (1/2N) * (y - Xw)^T (y - Xw)$$
$$L = (1/2N) * (y^T - (Xw)^T)\,(y - Xw)$$
$$L = (1/2N) * (y^T - w^T X^T)\,(y - Xw)$$
$$L = 1/N(0.5y^T y \; - w^T X^T y + 0.5w^T X^T Xw)$$
$$\nabla L = 1/N(-X^T y + X^T Xw)$$
$$\nabla^2 L = 1/N(X^T X)$$

The Hessian is positive for all its values and this make it convex and sometimes strictly convex if all of the x elements are larger than 0

Q2) Derive the classification boundary equation for the logistic regression problem.

Classification Boundary equation WX=0

$Logistic(x) = 1/(1 + e^{-Wx})$

If we substitute this value with the sigmoid we get the output to be 0.5

$(at\ x = 0)\ \ {-}{-}{>}\ Logistic(0) = 0.5$

Q3) Carry out five Gradient Descent iterations for the following linear regression problem:-

X	0	1	2	3	4	5	6	7	8
Y	0	0.81	0.95	0.31	-0.59	-1	-0.59	0.31	0.95

Use a learning rate of 0.01.

y_hat = Xw

Initial start $w = [0\ 0]$

N=9, alpha=0.01

$$L = (1/2N)(y - Xw)^T\ (y - Xw)$$

$$\nabla L = (1/N)(X^T Xw - X^T y)$$

Update rule: $w_{i+1} = w_i - \alpha \nabla L$

Iteration	Weights	Loss	delta
0	[0 0]	0.2416	[-0.27888889 -0.12777778]
1	[0.00278889 0.00127778]	0.2408	[-0.21056296 -0.11534444]
2	[0.00489452 0.00243122]	0.2402	[-0.15822158 -0.10576848]
3	[0.00647673 0.00348891]	0.2399	[-0.11812728 -0.09838193]
4	[0.00765801 0.00447273]	0.2397	[-0.08741649 -0.09267302]

Q4) For the regression problem in (3), find the Hessian matrix after 5 iterations. Is this learning process convex? Hint (Find the Eigenvalues. Also, you may visualize the loss function vs the weighting coefficients.)

$$Av = \lambda v$$

$$Av - \lambda v = 0$$

$$|A - \lambda I| v = 0$$

$$|A - \lambda I| = 0$$

$A = H,\ H = X^T X$

X = [[0, 1], [1, 1], [2, 1], [3,1], [4,1], [5,1], [6,1], [7,1], [8,1]]

Hessian matrix =X^TX= [[204, 36],
[36, 9]]

$|H - \lambda I| = 0$

$$\left| \begin{bmatrix} 204 & 36 \\ 36 & 9 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right| = 0$$

$$\left| \begin{bmatrix} 204 - \lambda & 36 \\ 36 & 9 - \lambda \end{bmatrix} \right| = 0$$

$$(204 - \lambda)(9 - \lambda) - 36^2 = 0$$
$$204 * 9 - 204\lambda - 9\lambda + \lambda^2 = 36^2$$
$$1836 - 213\lambda + \lambda^2 = 1296$$
$$\therefore \lambda = 210.43, \lambda = 2.57$$

Eigenvalues are positives, so it is a convex learning process

---

Q5) Use the closed form solution technique to find the linear model parameters of problem (3).

$$[\mathbf{D}]_{(N)X(d+1)}[\mathbf{W}]_{(d+1)X1} = [\mathbf{Y}]_{(N)X1}$$

$$W = (D^T D)^{-1} D^T Y$$

D: N x (d + 1) , N: number of samples, d: features number

D = [[0, 1]
[1, 1]
[2, 1]
[3, 1]

[4, 1]
[5, 1]
[6, 1]
[7, 1]
[8, 1]]

D^T= [[0 1 2 3 4 5 6 7 8] ,
[1 1 1 1 1 1 1 1 1]]

Y=[0
0.81
0.95
0.31
-0.59
-1
-0.59
0.31
0.95]

Substituted in the weights equation with D,D^T,Y To get W

$W = [-0.03483333 \;\; 0.26711111]$

Q6) Fit the following model, $y = \sum^{M}{}_{i=0}(a_i * x^i)$ on the data given in problem (3) for M =10. Comment on the results. Use the gradient descent optimization with a suitable learning rate.

$y(x, a) = a_0x^0 + a_1x^1 + a_2x^2 + a_3x^3 + ... + a_9x^9 + a_{10}x^{10} = \sum^{M}{}_{i=0}(a_i * x^i)$

$L(a) = (1/2N)\sum_{n=0}{}^{N}(y_n - \sum^{M}{}_{i=0}(a_i * (x_n)^i))^2$

$L = (1/N)(0.5y^Ty - w^TX^Ty + 0.5w^TX^TXw)$

$\nabla L = (1/N)(X^TXw - X^Ty)$

$H = \nabla^2 L = (1/N)(X^TX)$

$\alpha = H^{-1}$

Update Rule: $W_{i+1} = W_i - \alpha \nabla L$

iteration	Weights	Loss	∇L	$\|\nabla L\|$
0	[1.68 0.24 0. 0.02 -0.08 0.03 -0. -0. 0. -0.]	0.29	[0.56, 0.98, 3.33, 12.46, 44.08, 101.8, -422.74, -9871.28, -114600.17, -1119495.6]	1125389. 365
1	[0.05 1. -0.2 0.05 -0.08 0.03 -0. -0. 0. -0.]	0.0	,2.28 ,0.37 ,0.07 ,0.02] ,790.19 ,108.53 ,15.37 ,44347.1 ,5875.33 [338480.59	341424.8 55
2	[-0.05 1.05 -0.22 0.05 -0.08 0.03 -0. -0. 0. -0.]	0.0	[-0.02, -0.04, -0.2, -1.26, -8.18, -56.02, -397.59, -2893.08, -21439.6, -161088.21]	162534.9 12
3	[0.02 1.02 -0.21 0.05 -0.08 0.03 -0. -0. 0. -0.]	0.0	[0.01, 0.03, 0.16, 1.02, 7.13, 51.62, 383.0, 2888.66, 22041.99, 169654.71]	171105.4 13

The iterations will continue to around 18 steps (with epsilon 0.01) …
Code for the problem will be found in folder sheet4_codes

Q7) Show that the weighting coefficients can be estimated in a closed form solution linear regression problem as follows:
$W = (D^T D + \lambda I)^{-1} D^T Y$ adopting a regularization term in the loss function.

$$\tilde{J}(\mathbf{W}) = \frac{1}{2N}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - Y_n\}^2 + \frac{\lambda}{2}\|\mathbf{W}\|^2$$

$y(x_n, w) = Dw$

$J = 1/2\ [(Dw - y)^T (Dw - y)] + \lambda/2\ \|W\|^2$

$J = 1/2[((Dw)^T - y^T)(Dw - y)] + \lambda/2\ w^T w$

$J = 1/2[(w^T D^T - y^T)(Dw - y)] + \lambda/2\ w^T w$
$J = 1/2[w^T D^T Dw - w^T D^T y - y^T Dw + y^T y] + \lambda/2\ w^T w$
Note: $w^T D^T y\ =\ y^T Dw$ because the result of the dot product in the two cases will give us a scalar value.

$J = 1/2 * w^T D^T Dw - w^T D^T y + 1/2 * y^T y + \lambda/2\ w^T w$
$\nabla J = D^T Dw - D^T y + \lambda w$
At $\nabla J = 0$ the desired weights are reached

$D^T Dw - D^T y + \lambda w = 0$
$D^T Dw + \lambda w = D^T y$
$(D^T D + \lambda I)w = D^T y$
$w = (D^T D + \lambda I)^{-1} D^T y$

Q8) The following data set of 2D points, {(-1, -1), (+1, -1), (-1, +1), (+1, +1)} and their corresponding labels {+1, +1, +1, -1} is trained with a logistic regression model. Assume a suitable learning rate, find and visualize the classification boundary.

1. Initialize weights with [0 0 0], alpha = 1 & epsilon = 0.2
2. Compute the gradient, delta += $-y_i X_i$ / $(1+\exp(y_i W.X_i))$
3. Check if delta is less than epsilon, if not continue else exit
4. Update weights $W_{i+1}\ =\ W_i\ -\ \alpha\, L_w/N$
5. Go To Step 2

Results are provided in sheet4_codes/Q8_logisitic_regression.py

---

Q9) Given a data set of RGB colors, {(0, 0, 0), (255, 0, 0), (0, 255, 0), (0, 0, 255), (255, 255, 0), (0, 255, 255), (255, 0, 255), (255, 255, 255)} and their corresponding labels {+1, +1, +1, -1, +1, -1, -1, +1}, train a logistic regression model using gradient descent with a suitable learning rate. Visualize the classification boundary. You may need to normalize the input feature vectors.

Normalizing feature vector: divide features by 255

So the dataset now is {(0, 0, 0), (1, 0, 0), (0, 1, 0), (0, 0, 1), (1, 1, 0), (0, 1, 1), (1, 0, 1), (1, 1, 1)}

And we can apply the Batch Gradient Descent Algorithm

1. Initialize weights with [0 0 0], alpha = 1 & epsilon = 0.2
2. Compute the gradient, delta += $-y_iX_i$ / $(1+\exp(y_iW.X_i))$
3. Check if delta is less than epsilon, if not continue else exit
4. Update weights $W_{i+1} = W_i - \alpha\, L_w/N$
5. Go To Step 2

Results are provided in sheet4_codes/Q9_logisitic_normilization.py

---

Q10) Consider using an identity activation function with the logistic regression problem. Recommend a loss function and derive the learning equation adopting a regularization term.

Loss function:

$$L = \log(1 + \exp(-y \cdot \hat{y}))$$

Y_hat = $X_{Nx(d+1)} \cdot w_{(d+1)x1}$

$$L = 1/N \sum log_e(1 + e^{-yXw})$$

$$L_{Regularized} = L + (\lambda/2) * \|w\|^2$$

$$L_{Regularized} = L + (\lambda/2) * w^T w$$

$$\nabla L = (1/N) \frac{-y\,.\,x\ exp(-y\,.\,Xw)}{1 + exp(-y\,.\,Xw)} + \lambda w$$

Update Rule: $W_{i+1} = W_i - \alpha\, \nabla L$