Intro to ML 5521 Majid Farhadloo

$$\frac{\partial \text{empiricalLoss}}{\partial w_0} = \frac{1}{N}\sum_{t=1}^{N}\left(r^t - (w_1x^t + w_0)\right)^2$$

(1) $\frac{1}{N}\sum_{t=1}^{N} (r^t)^2 - 2r^t(w_1x^t + w_0) + (w_1x^t + w_0)^2$

$$\downarrow$$

$$\left((w_1x^t)^2 + 2w_1x^t w_0 + w_0^2\right)$$

$\frac{\partial(1)}{\partial w_0}$

$\Rightarrow \frac{1}{N}\sum_{t=1}^{N} -2r^t + 2(w_1x^t + w_0) \Rightarrow$

(2) $\frac{1}{N}\sum_{t=1}^{N} 2(-r^t + (w_1x^t + w_0)) = 0$

multiply (2) by $\frac{N}{2} \Rightarrow$

(3) $\sum_{t=1}^{N} w_0 + \sum_{t=1}^{N} w_1x^t = \sum_{t=1}^{N} r^t$

$\Rightarrow \left(Nw_0 + \sum_{t=1}^{N} w_1x^t = \sum_{t=1}^{N} r^t\right)$

with simillar approach take derivative of (1) with respect to w_1

$\Rightarrow$ (4)

$\Rightarrow \frac{1}{N}\sum_{t=1}^{N} 2(r^t - w_1x^t - w_0)(-x^t) = 0$

(4) $\times \frac{N}{2}$

$$\Downarrow \;\; (5) \quad w_0 \sum_{t=1}^{N} x^t + w_1 \sum_{t=1}^{N} (x^t)^2 = \sum_{t=1}^{N} x^t r^t$$

(6) $\rightarrow$ divid 3 by N

$$\Downarrow \;\; \underline{\underline{w_0 + w_1 x^t = r^t}}$$

we have two equation of two derivative $\frac{\partial(1)}{w_0}$, $\frac{\partial(1)}{w_1}$

$\hookrightarrow$ use them to solve equation.

$$\xrightarrow{\times \sum_{t=1}^{N} x^t} N w_0 + w_1 \sum_{t=1}^{N} x^t = \sum_{t=1}^{N} r^t \quad (3)$$

$$\xrightarrow{\times N} w_0 \sum_{t=1}^{N} x^t + w_1 \sum_{t=1}^{N} (x^t)^2 = \sum_{t=1}^{N} x^t r^t \quad (5)$$

$$(7) \left(N w_0 \sum_{t=1}^{N} x^t + w_1 \left(\sum_{t=1}^{N} x^t\right)^2 = \sum_{t=1}^{N} x^t \sum_{t=1}^{N} r^t \right)$$

$$(8) \quad N w_0 \sum_{t=1}^{N} x^t + N w_1 \left(\sum_{t=1}^{N} x^t\right)^2 = N \sum_{t=1}^{N} x^t r^t$$

$(8) \xRightarrow{-(7)}$ and then divide by L.H.S

$$w_1 = \frac{N \sum_{t=1}^{N} x^t r^t - \sum_{t=1}^{N} x^t \sum_{t=1}^{N} r^t}{(N-1)\left(\sum_{t=1}^{N} x^t\right)^2}$$

In order to solve for w_0, we need to plugin w_1 and solve equation (6).

The solution for w_1 and w_0 are optimal solution.

(ii) ((iii))

(1) $$E(w_2, w_1, w_0 | \text{train}) = \frac{1}{N} \sum_{t=1}^{N} \left(r^t - \left(w_2 (x^t)^{2020} + w_1 x^t + w_0\right)\right)^2$$

$$\frac{\partial(1)}{\partial w_0}$$

$$\Rightarrow \frac{1}{N} \sum_{t=1}^{N} (r^t)^2 - 2(r^t)\left(v_2 (x^t)^{2020} + v_1 x^t + v_0\right) + \left(v_2 (x^t)^{2020} + v_1 x^t + v_0\right)^2$$

(2) $$\left((v_2 x^t)^{2020}\right)^2 + (v_1 x^t)^2 + (v_0)^2 + 2 v_2 (x^t)^{2020} v_1 x^t + 2 v_2 (x^t)^{2020} v_0 + 2 v_1 x^t v_0$$

$$\frac{\partial(2)}{\partial v_0} = \frac{1}{N} \sum_{t=1}^{N} 2\left(-r^t + v_2 (x^t)^{2020} + v_1 x^t + v_0\right) = 0$$

$$\Rightarrow \times \frac{N}{2} \Rightarrow N v_0 + v_2 \sum_{t=1}^{N} (x^t)^{2020} + v_1 \sum_{t=1}^{N} x^t = \sum_{t=1}^{N} r^t$$

with simillar approach we can get the derivative w.r.t v_1, v_2.

$$v_1 \Rightarrow v_0 \sum_{t=1}^{N} x^t + v_1 \sum_{t=1}^{N} (x^t)^2 + v_2 \sum_{t=1}^{N} (x^t)^{2021} = \sum_{t=1}^{N} x^t r^t$$

$$v_2 \rightarrow v_0 \sum_{t=1}^{N} (x^t)^2 + v_1 \sum_{t=1}^{N} (x^t)^{2021} + v_2 \sum_{t=1}^{N} (x^t)^{2\times 2020} = \sum_{t=1}^{N} (x^t)^{2020} r^t$$

$$\begin{bmatrix} N & \sum_{t=1}^{N} x^t & \sum_{t=1}^{N} (x^t)^{2020} \\ \sum_{t=1}^{N} x^t & \sum_{t=1}^{N} (x^t)^2 & \sum_{t=1}^{N} (x^t)^{2021} \\ \sum_{t=1}^{N} (x^t)^2 & \sum_{t=1}^{N} (x^t)^{2021} & \sum_{t=1}^{N} (x^t)^{2\times 2020} \end{bmatrix} \begin{bmatrix} v_0 \\ v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \sum_{t=1}^{N} r^t \\ \sum_{t=1}^{N} x^t r^t \\ \sum_{t=1}^{N} (x^t)^{2020} r^t \end{bmatrix}$$

in order to solve optimal solutions for v_0, v_1, v_2, it is a good idea that lets matlab or other computer programming language solve it.

(iii))

iii) Yes, his claim is correct, adding more parameter in data model gives us a better fit, and this results in a better model and less error. This could also remind us about overfiting a model.

2.
$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \end{bmatrix}$$

$tr(A) = \underline{76}$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 4 & 9 & 16 \\ 1 & 8 & 27 & 64 \end{bmatrix} = A^T \to tr(A^T) = \underline{76}$$

$tr(A^T A) = tr(A A^T) = \boxed{15278}$

$$\begin{bmatrix} 4 & 10 & 30 & 100 \\ 10 & 30 & 100 & 354 \\ 30 & 100 & 354 & 1300 \\ 100 & 354 & 1300 & 4890 \end{bmatrix}$$

ii) as matrix is indicated as 4x4 M, calculating the |A| determines a volume of 4 dimensional "parallelogram" formed by rows of A.

$|A| = \underline{12.000}$

iii) there are multiple ways to derive to solution, one way is to calculate the matrix, and see if the matrix is full rank or not which in this it is as there is no way to derive a column by scale one column + another column and this case rank of matrix is 4 and shows ~~rows~~ columns are linearly independent.

$$\text{rank}(A) = \min(4 \times 4) = \boxed{4}$$

Also, as $|A| \neq 0$ shows that A is not clearly linearly dependent.

Introduction to ML – HW1

Majid Farhadloo

February 11, 2020

1 Question 3 part I

============== method: LinearSVC, dataset: Boston50
[0.6862745098039216, 0.8431372549019608, 0.8823529411764706, 0.8235294117647058, 0.6862745098039216, 0.8235294117647058, 0.7, 0.8, 0.62, 0.7]

method: LinearSVC

dataset: Boston50

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.3137	0.1569	0.1176	0.1765	0.3137	0.1765	0.3000	0.2000	0.3800	0.3000	0.2435	0.0831

============== method: LinearSVC, dataset: Boston75
[0.9803921568627451, 0.49019607843137253, 0.9215686274509803, 0.8235294117647058, 0.6274509803921569, 0.9607843137254902, 0.86, 0.8, 0.78, 0.84]

method: LinearSVC

dataset: Boston75

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0196	0.5098	0.0784	0.1765	0.3725	0.0392	0.1400	0.2000	0.2200	0.1600	0.1916	0.1430

============== method: LinearSVC, dataset: Digits
[0.9722222222222222, 0.95, 0.9555555555555556, 0.95, 0.9555555555555556, 0.9611111111111111, 0.9611111111111111, 0.9720670391061452, 0.9329608938547486, 0.9608938547486033]

method: LinearSVC

dataset: Digits

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0278	0.0500	0.0444	0.0500	0.0444	0.0389	0.0389	0.0279	0.0670	0.0391	0.0429	0.0109

============== method: SVC, dataset: Boston50 [0.6862745098039216, 0.8627450980392157, 0.7843137254901961, 0.7647058823529411, 0.7254901960784313, 0.6862745098039216, 0.82, 0.8, 0.76, 0.72]

method: SVC

dataset: Boston50

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.3137	0.1373	0.2157	0.2353	0.2745	0.3137	0.1800	0.2000	0.2400	0.2800	0.2390	0.0548

============== method: SVC, dataset: Boston75 [0.8431372549019608, 0.6666666666666666, 0.7843137254901961, 0.7058823529411765, 0.7843137254901961, 0.8627450980392157, 0.8, 0.7, 0.82, 0.72]

method: SVC

dataset: Boston75

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.1569	0.3333	0.2157	0.2941	0.2157	0.1373	0.2000	0.3000	0.1800	0.2800	0.2313	0.0632

============== method: SVC, dataset: Digits [0.9722222222222222, 0.9944444444444445, 0.9944444444444445, 1.0, 0.9888888888888889, 0.9888888888888889, 0.9888888888888889, 0.9888268156424581, 0.994413407821229, 0.994413407821229]

method: SVC

dataset: Digits

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0278	0.0056	0.0056	0.0000	0.0111	0.0111	0.0111	0.0112	0.0056	0.0056	0.0095	0.0070

=============== method: LogisticRegression, dataset: Boston50 [0.8823529411764706, 0.8823529411764706, 0.9019607843137255, 0.8627450980392157, 0.8627450980392157, 0.8431372549019608, 0.88, 0.8, 0.84, 0.88]

method: LogisticRegression

dataset: Boston50

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.1176	0.1176	0.0980	0.1373	0.1373	0.1569	0.1200	0.2000	0.1600	0.1200	0.1365	0.0278

=============== method: LogisticRegression, dataset: Boston75 [0.8823529411764706, 0.9803921568627451, 0.8823529411764706, 0.8627450980392157, 0.9607843137254902, 0.8235294117647058, 0.88, 0.88, 0.96, 0.92]

method: LogisticRegression

dataset: Boston75

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.1176	0.0196	0.1176	0.1373	0.0392	0.1765	0.1200	0.1200	0.0400	0.0800	0.0968	0.0477

=============== method: LogisticRegression, dataset: Digits [0.9555555555555556, 0.9722222222222222, 0.9777777777777777, 0.9611111111111111, 0.9722222222222222, 0.9888888888888889, 0.9722222222222222, 0.9385474860335196, 0.9776536312849162, 0.9608938547486033]

method: LogisticRegression

dataset: Digits

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0444	0.0278	0.0222	0.0389	0.0278	0.0111	0.0278	0.0615	0.0223	0.0391	0.0323	0.0134

2 Question 3 part II

============== method: LinearSVC, dataset: Boston50

method: LinearSVC

dataset: Boston50

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.2598	0.1969	0.3780	0.3780	0.2126	0.3071	0.3780	0.3465	0.2677	0.2756	0.3000	0.0648

============== method: LinearSVC, dataset: Boston75
method: LinearSVC

dataset: Boston75

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.2362	0.1024	0.0866	0.2362	0.5197	0.2126	0.1811	0.2677	0.2283	0.3071	0.2378	0.1141

============== method: LinearSVC, dataset: Digits
method: LinearSVC

dataset: Digits

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0422	0.0578	0.0600	0.0622	0.0489	0.0511	0.0622	0.0600	0.0622	0.0578	0.0564	0.0065

============== method: SVC, dataset: Boston50

method: SVC

dataset: Boston50

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.2283	0.1732	0.2283	0.2677	0.2913	0.2520	0.2047	0.1969	0.2441	0.2598	0.2346	0.0339

============== method: SVC, dataset: Boston75

method: SVC

dataset: Boston75

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.1890	0.2362	0.2126	0.2362	0.1811	0.2362	0.3071	0.2520	0.2756	0.2441	0.2370	0.0356

============== method: SVC, dataset: Digits

method: SVC

dataset: Digits

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0089	0.0044	0.0067	0.0067	0.0000	0.0133	0.0044	0.0178	0.0244	0.0133	0.0100	0.0069

============== method: LogisticRegression, dataset: Boston50

method: LogisticRegression

dataset: Boston50

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.1732	0.1496	0.1260	0.1181	0.1417	0.1575	0.1811	0.1339	0.1102	0.1339	0.1425	0.0218

============== method: LogisticRegression, dataset: Boston75 method: LogisticRegression

dataset: Boston75

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.1496	0.0787	0.0709	0.0630	0.1102	0.1260	0.1260	0.0945	0.1102	0.1102	0.1039	0.0258

============== method: LogisticRegression, dataset: Digits

method: LogisticRegression

dataset: Digits

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0511	0.0333	0.0444	0.0244	0.0444	0.0311	0.0378	0.0289	0.0400	0.0489	0.0384	0.0084

3 Question 4

============== method: LinearSVC, dataset: X1
method: LinearSVC

dataset: X1

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0778	0.0778	0.0944	0.1000	0.1500	0.0556	0.0667	0.0670	0.1229	0.0838	0.0896	0.0272

============== method: LinearSVC, dataset: X2
method: LinearSVC

dataset: X2

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0111	0.0222	0.0056	0.0222	0.0111	0.0111	0.0056	0.0112	0.0168	0.0000	0.0117	0.0068

=============== method: SVC, dataset: X1

method: SVC

dataset: X1

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0111	0.0333	0.0167	0.0167	0.0333	0.0111	0.0111	0.0391	0.0223	0.0279	0.0223	0.0100

=============== method: SVC, dataset: X2

method: SVC

dataset: X2

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0111	0.0111	0.0000	0.0278	0.0000	0.0000	0.0056	0.0223	0.0056	0.0112	0.0095	0.0090

=============== method: LogisticRegression, dataset: X1
method: LogisticRegression

dataset: X1

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0444	0.0778	0.0722	0.0611	0.0500	0.1000	0.0667	0.0726	0.0670	0.0894	0.0701	0.0158

=============== method: LogisticRegression, dataset: X2

method: LogisticRegression

dataset: X2

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean	std dev
0.0167	0.0111	0.0167	0.0111	0.0056	0.0222	0.0278	0.0000	0.0112	0.0112	0.0133	0.0075