MACHINE LEARNING
HOMEWORK - 2
SHRAINIK JAIN - 1323338

1) $H(x) = sgn \left\{ \sum_{t=1}^{T} \alpha_t h_t(x) \right\} = sgn\{f(x)\}$

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

$\epsilon_{Training}$ = Probability of wrong prediction over the distribution of weighted data points

$= P_{i \sim D_t} [H_t(x_i) \neq y_i]$

$= \dfrac{\text{Number of training point where we predicted incorrectly}}{\text{Total number of training points}}$

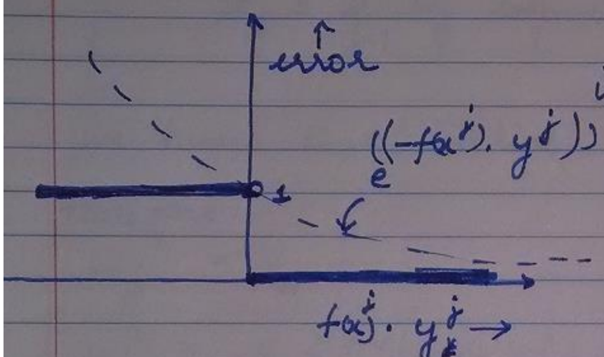
$$\boxed{\epsilon_{Train} = \frac{\sum_{j=1}^{N} \mathbb{1}\{H_o(x^j) \neq y^j\}}{N}} \rightarrow (1)$$

← i.e. 1, when $H(x^j) \neq y^j$
0, otherwise.

we make an error when
$y^j = 1$ and $f(x^j) \leq 0$ { sgn is -1 }
or
$y^j = -1$ and $f(x^j) \geq 0$ { sgn is +1 }

This implies that we make an error when $\boxed{f(x^j) \cdot y^j \leq 0}$

i.e. error for a single point $= \begin{cases} 1 & , f(x^j).y^j < 0 \\ 0, & \text{otherwise} \end{cases}$



in other words

$$\epsilon_{Train} = \frac{1}{N} \sum_{j=1}^{N} (\text{error for point } j)$$

$$\boxed{= \frac{1}{N} \sum_{j=1}^{N} \mathbb{1}(f(x^j.y_j) < 0)}$$

↳ ②

The above step function for the error of classifying j^{th} point is bounded by $\exp(-f(x^j).y^j)$ because:

$$\exp(-f(x_j).y^j) > 1 \quad \text{for} -f(x^j).y^j < 0$$

$$1 \geq \exp(-f(x_j).y^j) > 0 \quad \text{for} -f(x^j).y^j \geq 0$$

i.e. $\boxed{\exp(-f(x_j).y^j) \geq f(x^j).y^j \Big\} \text{ for all } (x^j, y^j)}$

↳ ③

combining equations ①, ② & ③ we get:

$$\boxed{\epsilon_{Training} = \frac{1}{N} \sum_{j=1}^{N} 1\{H(x^j) \neq y^j\} \leq \frac{1}{N} \sum_{j=1}^{N} \exp(-f(x^j).y^j)}$$

Q1. 2) $$w_j^{(t+1)} = \frac{w_j^{(t)} \exp(-\alpha_t y^j h_t(x^j))}{Z_t}$$

$$Z_t = \sum_{j=1}^{N} w_j^{(t)} \exp(-\alpha_t y^j h_t(x^j))$$

we know that initial $w^{1}_{j=1 \text{ to } N} = 1/N$ } equal weights

therefore:

$$w_j^{1} = 1/N \quad \dots$$

$$w_j^{2} = 1/N \left(\frac{\exp(-\alpha_{\cancel{t}1} y^j h_1(x^j))}{Z_{\cancel{t}1}} \right)$$

$$\vdots$$

$$w_j^{(t+1)} = 1/N \left(\frac{\exp(\alpha_1 y^j h_1(x^j))}{Z_1} \right) \cdot \left(\frac{\exp(-\alpha_2 y^j h_2(x^j)}{Z_2} \right) \cdot$$

$$\dots\dots\dots \left(\frac{\exp(-\alpha_t y^j h_t(x^j)}{Z_t} \right)$$

$$\boxed{w_j^{(t+1)} = 1/N \; \frac{\prod_{\cancel{t}=1}^{t} \exp(-\alpha_{\cancel{t}i} y^j h_i(x^j))}{\prod_{i=1}^{t} Z_{\cancel{t}i}}}$$

now Sum of all weights at each iteration is 1.

So, $\sum_{j=1}^{N} \omega_j^{t+1} = 1$

$$\therefore \quad \frac{1}{N}\sum_{j=1}^{N} \frac{\left(\prod_{t=1}^{T} \exp(-\alpha_t y^j h_t^j(x^j))\right)}{\prod_{t=1}^{T} Z_t} = 1$$

$$\frac{1}{N}\sum_{j=1}^{N} \exp\left(\sum_{t=1}^{T}(-\alpha_t y^j h_t(x^j))\right) = \prod_{t=1}^{T} Z_t$$

$$\frac{1}{N}\sum_{j=1}^{N} \exp\left((-y^j)\left(\sum_{t=1}^{T} \alpha_t h_t(x^j)\right)\right) = \prod_{t=1}^{T} Z_t$$

. also $\sum_{t=1}^{T} \alpha_t h_t(x^j) = f(x_j)$

$$\therefore \quad \boxed{\frac{1}{N}\sum_{j=1}^{N} \exp(-y^j \cdot f(x^j)) = \prod_{t=1}^{T} Z_t .}$$

Q.E.D.

Q1) 3) (a) $\epsilon_t = \sum_{j=1}^{m} w_j^t \mathbb{1}\{h_t(x^j) \neq y^j\}$

$$z_{\dot{t}} = (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$$

$$\alpha_t \text{ for } z_t \text{ max} \Rightarrow \frac{\partial z_t}{\partial \alpha_t} = 0$$

$$\Rightarrow -(1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t) = 0$$

$$\Rightarrow (1-\epsilon_t)\exp(-\alpha_t) = \epsilon_t \exp(\alpha_t)$$

$$\Rightarrow \ln(1-\epsilon_t) - \alpha_t = \ln(\epsilon_t) + \alpha_t$$

$$\Rightarrow \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) = 2\alpha_t$$

$$\boxed{\hat{\alpha}_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)}$$

$$z_t^{opt} = (1-\epsilon_t)\exp\left(-\frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)\right) + \epsilon_t \exp\left(\frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)\right)$$

$$\Rightarrow (1-\epsilon_t)\cdot\left(\frac{1-\epsilon_t}{\epsilon_t}\right)^{-1/2} + \epsilon_t\left(\frac{1-\epsilon_t}{\epsilon_t}\right)^{1/2}$$

$$\Rightarrow (1-\epsilon_t)^{1/2}\epsilon_t^{1/2} + \epsilon_t^{1/2}(1-\epsilon_t)^{1/2}$$

$$\Rightarrow \boxed{Z_t^{opt} = 2\sqrt{\epsilon_t(1-\epsilon_t)}}$$

Q.E.D.

Q1 3) (b) $\epsilon_t = \frac{1}{2} - \gamma_t$

$$\Rightarrow Z_t^{opt} = 2\sqrt{(\tfrac{1}{2} - \gamma_t)(1 - (\tfrac{1}{2} - \gamma_t))}$$

$$\Rightarrow \cdot \frac{\cancel{2}}{\cancel{2}} \sqrt{(1-2\gamma_t)(2\gamma_t - 1)}$$

$$Z_t^{opt} \Rightarrow \sqrt{1 - 4\gamma_t^2}$$

also, ~~$Z_t^{opt} = (1-\epsilon_t)\exp(-\frac{1}{2}$~~

~~$Z_t \leq$~~ $Z_t^{opt} = e^{\frac{1}{2}\ln(1-4\gamma_t^2)}$

also, since $\log(1-x) \leq -x$ for $0 \leq x \leq 1$

$$\ln(1-4\gamma_t^2) \leq -4\gamma_t^2$$

$\Uparrow$

because

$0 \leq \gamma_t \leq \frac{1}{2}$

$\gamma_t^2 \leq \frac{1}{4}$

$0 \leq 4\gamma_t^2 < 1$

therefore $\Rightarrow$

$$Z_t^{opt} \leq e^{\frac{1}{2}(-4\gamma_t^2)} = e^{-2\gamma_t^2}$$

$$\therefore Z_t\ opt \leq \exp(-2\gamma_t^2)$$

$$\therefore \boxed{Z_t \leq \exp(-2\gamma_t^2)}$$ Q.E.D.

$$\therefore \quad \epsilon_{Training} \le \prod_{t=1}^{T} Z_t \le \exp\left(-2 \sum_{t=1}^{T} \gamma_t^2\right)$$

Q 1. 3) c) if each classifier is better than random,

then $\epsilon_T < 1/2$

$$\epsilon_T = 1/2 - \gamma_t$$

~~for all γ_t~~ for some $\boxed{0 \le \gamma_t < 1/2}$

this implies $\exists \boxed{\gamma = \min(\gamma_1, \gamma_2 \ldots \gamma_t)}$

$\therefore$

$$\epsilon_{Training} \le \exp\left(-2 \sum_{t=1}^{T} \gamma_t^2\right)$$

since $\forall \gamma_t$ $\boxed{\gamma \le \gamma_t}$

$$\exp(-2\gamma_t^2) \le \exp(-2\gamma^2)$$

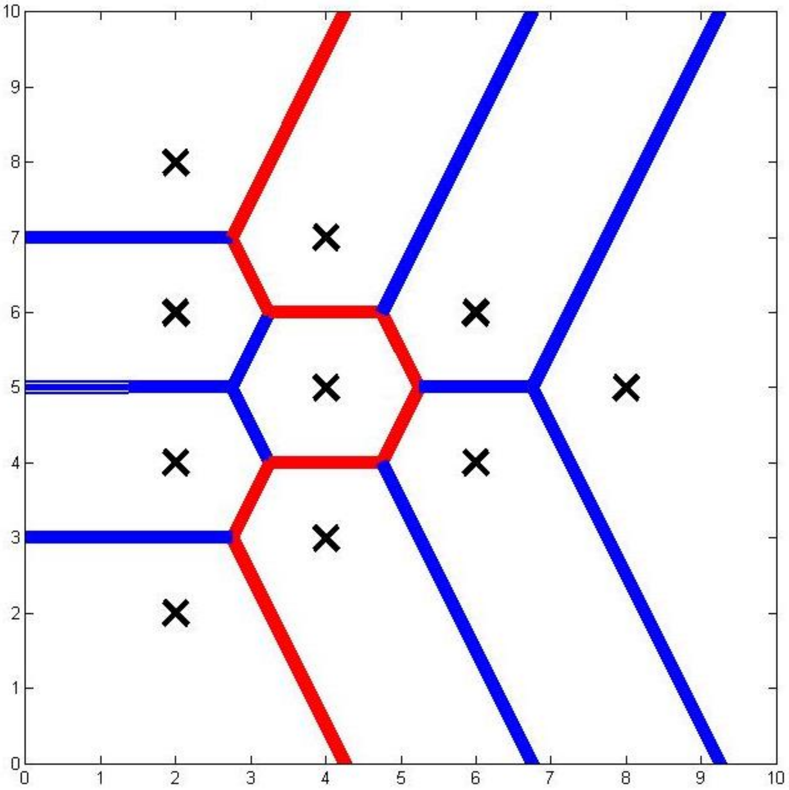
$$\therefore \epsilon_{Train} \le \underbrace{\exp(-2\gamma^2) . \exp(-2\gamma^2) \ldots \ldots \exp(-2\gamma^2)}_{T \text{ times}}$$

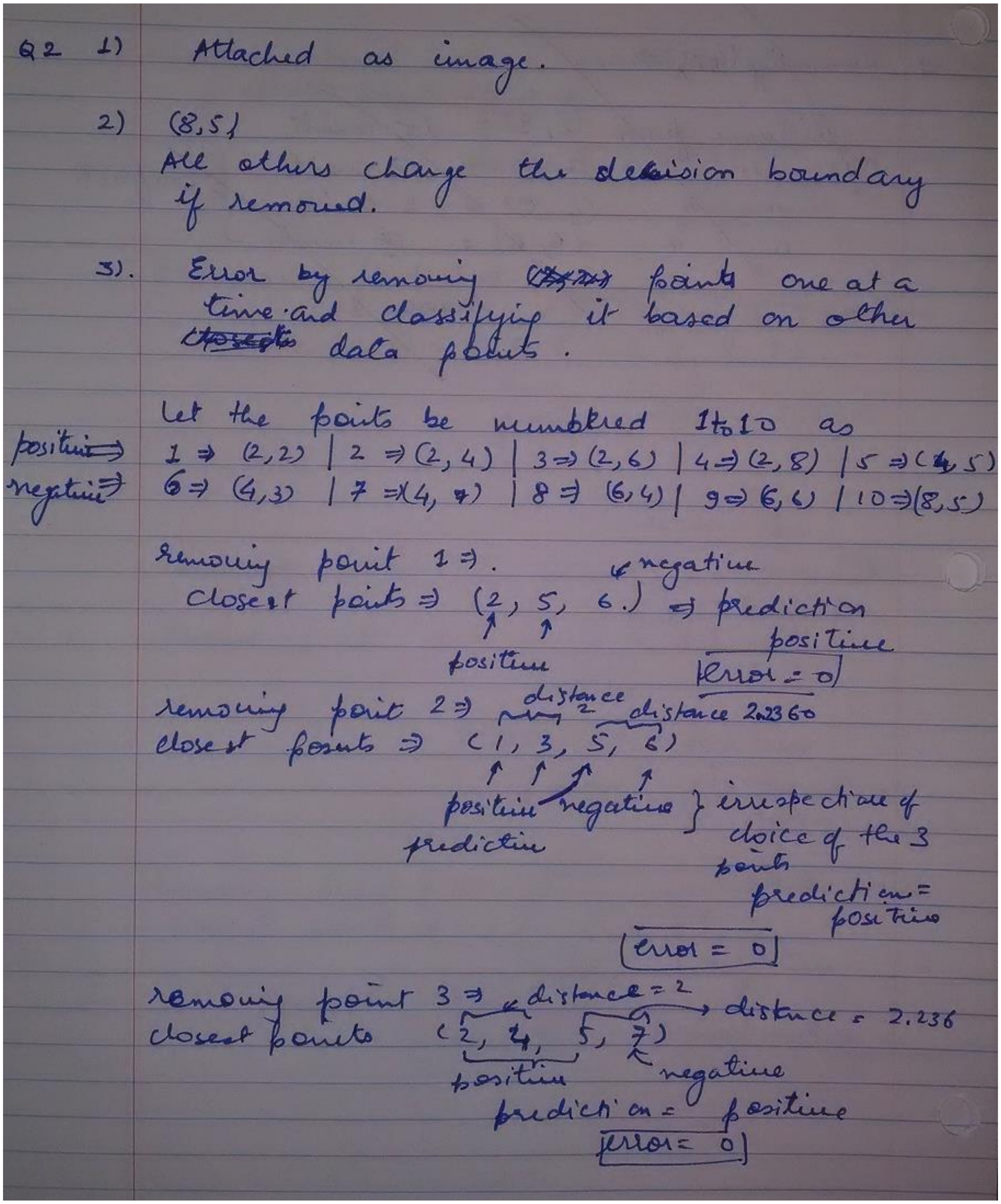
$$\boxed{\epsilon_{Train} \le \exp(-2T\gamma^2)}$$

Q. E. D.

Ques 2. 1

The following figure is the Voronoi diagram for the dataset. The Redline is the decision boundary, i.e. all points left of it are positive.





Q2 1) Attached as image.

2) (8,5)

All others change the decision boundary if removed.

3). Error by removing ~~[illegible]~~ points one at a time and classifying it based on other ~~closests~~ data points.

Let the points be numbered 1to10 as

positive ⇒ 1 ⇒ (2,2) | 2 ⇒ (2,4) | 3 ⇒ (2,6) | 4 ⇒ (2,8) | 5 ⇒ (4,5)

negative ⇒ 6 ⇒ (4,3) | 7 ⇒ (4,7) | 8 ⇒ (6,4) | 9 ⇒ (6,6) | 10 ⇒ (8,5)

removing point 1 ⇒.

closest points ⇒ (2, 5, 6.) ⇒ prediction positive

2, 5 ↑ positive; 6 ← negative

$\boxed{\text{Error} = 0}$

removing point 2 ⇒

closest points ⇒ (1, 3, 5, 6)

distance 2 (1, 3); distance 2.2360 (5, 6)

1, 3 ↑ positive; 5, 6 ↑ negative } irrespective of choice of the 3 points

prediction positive

prediction = positive

$\boxed{\text{error} = 0}$

removing point 3 ⇒

closest points (2, 4, 5, 7)

distance = 2 (2, 4); distance = 2.236 (5, 7)

2, 4 positive; 7 ← negative

prediction = positive

$\boxed{\text{error} = 0}$

removing point 4 ⇒

negative

closest points (3, 5, 7)

positive

prediction = positive

$\boxed{\text{error} = 0}$

~~p~~ removing point 5 ⇒

negative

positive

closest points = (6, 7, . 2, 3, 8, 9)

distance = 2 distance 2.236

$\boxed{\text{error} = 1}$

removing point 6 ⇒ distance = 2 distance = 2.236

negative

closest points = (5, 1, 2, 8 }

positive

prediction = positive

$\boxed{\text{error} = 1}$ } as actual value is negative

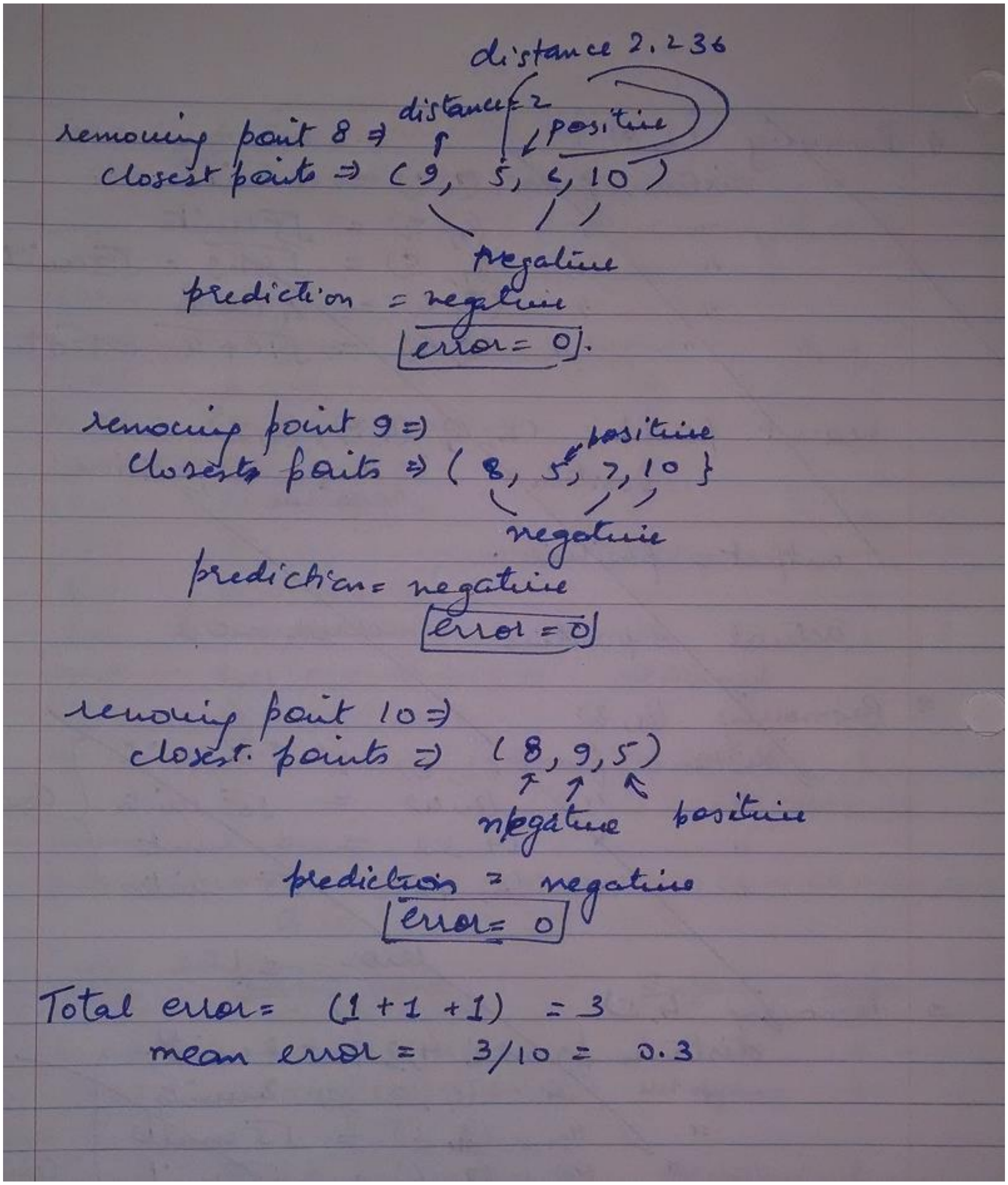
removing point 7 ⇒ distance 2 distance 2.236

closest points ⇒ (5, 3, 4, 9 }

positive negative

prediction ⇒ positive

$\boxed{\text{error} = 1}$ Actual value = negative



distance 2.236

removing point 8 ⇒ distance = 2 positive

closest points ⇒ (9, 5, 7, 10)

negative

prediction = negative

$\boxed{\text{error} = 0}$.

removing point 9 ⇒ positive

Closests points ⇒ (8, 5, 7, 10}

negative

prediction = negative

$\boxed{\text{error} = 0}$

removing point 10 ⇒

closest. points ⇒ (8, 9, 5)

negative positive

prediction = negative

$\boxed{\text{error} = 0}$

Total error = (1 + 1 + 1) = 3

mean error = 3/10 = 0.3

Q2. 4) Removing feature 1:-

Data points ⇒ (2, +) (4, +) (6,+) (8, +) (5, +)
(3, -) (7, -) (4, -) (6, -) (5, -)

LOOCV iterations

Point to remove	Closest Points	Output	Error.
(2, +)	(3, -);(4, -);(4, +)	-	1
(4, +)	(4, -);(3, -);(5, -);(5, +)	-	1
(6, +)	(6, -);(5, +);(5, -)	-	1
(8, +)	(7, -);(6, -);(6, +)	-	1
(5, +)	(5, -);(4, -);(6, -);(4, +);(6, +)	Not Defined	1
(3, -)	(2, +); (4, +); (4, -)	+	1
(7, -)	(6, +); (6, -); (8, +)	+	1
(4, -)	(4, +); (3, -); (5, -); (5, +)	Not Defined	1
(6, -)	(6, +); (5, +); (5, -)	+	1
(5, -)	(5, +); (4, -); (4, +); (6, -); (6, +)	Not Defined	1
		Total error:	10

Removing feature 2:

Data points ⇒ (2, +) (2, +) (2, +) (2, +) (4, +) (4, -) (4, -), (6, -), (6, -)
(8, -)

LOOCV Iterations.

Point to remove	Closest points	Output	Error
(2, +)	(2, +); (2, +); (2, +)	+	0
Same for other points at (2, +)		+	0
(4, +)	$(4,-);(4,-);\underbrace{[(2,+)]\times 4}_{\text{4 points with }(2,+)};(6,-);(6,-)$	Not defined	1
(4, -)	(4, +), (4, -) [(2, +)]×4; (6, -); (6, -)	Not defined	1
(4, -)	⟷ Same ⟷	Not defined	1
(6, -)	(6, -); (4, -); (4, -); (4, +); (8, -);	-	0
(6, -)	← same →	-	0
(8, -)	(6, -); (6, -); (4, -); (4, -); (4, +)	-	0
		Total ⇒	3

THEREFORE, we can safely eliminate feature 2.

Q3. 1) Let Y = has college degree and N = Does not have college degree

Root ⇒ $\boxed{\begin{matrix} Y = 6 \\ N = 4 \end{matrix}}$ $H(Y) = -\frac{6}{10} \log 6/10 - \frac{4}{10} \log 4/10$

$\Rightarrow 0.970951.$

Step 1:

we have the following intuitive splits

① salary ≤ 27000

② salary ≥ 65000

③ age ~~8~~ ≥ 43

$$\text{IG for choice 1} \Rightarrow 0.9709 - \overset{H[Y|X]}{\left[-0.8\left(\frac{2}{8}\log 2/8\right) + \frac{6}{8}\log(6/8) - 0.2 \times 0\right]}$$

$\Rightarrow 0.322$

$$\text{IG for choice 2} \Rightarrow 0.9709 - \overset{H[Y|X]}{\left[-0.3 \times 0 - 0.7\left(3/7 \log(3/7) + 4/7 \log(4/7)\right)\right]}$$

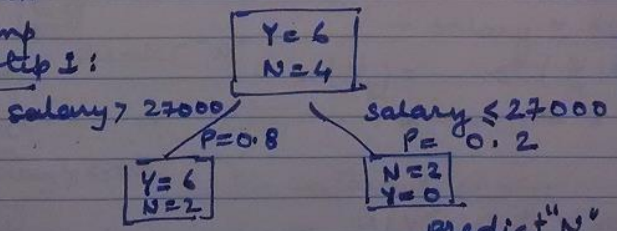
$\Rightarrow 0.282$

$$\text{IG for choice 3} \Rightarrow 0.9709 - \left[-0.5\left(2/5 \log 2/5 + 3/8 \log 3/5 + 1/5 \log 1/5 + 4/8 \log 4/5\right)\right]$$

$\Rightarrow 0.12\cancel{5}560.$

Max information gain for choice 1:

Decision stump after step 1:



$\boxed{IG = 0.322}$

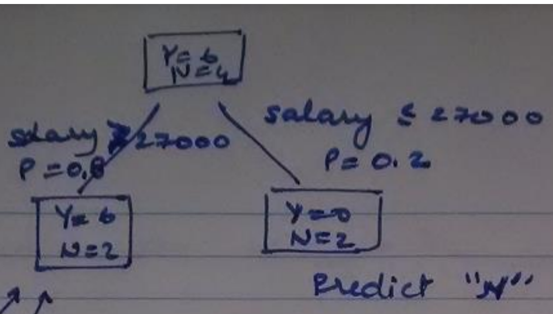
predict "N"

Step 2:

Now we have the following intuitive splits ⇒

① age ≥ 43

② salary ≥ 65000



$H(Y \mid salary > 27000) = 0.649022.$

$H(Y_j) \Rightarrow -1\left(\frac{2}{8}\log \frac{2}{8} + \frac{6}{8}\log \frac{6}{8}\right) = (0.811)$

IG for choice1 $\Rightarrow H(Y_j) - \left[0.5 \times 0 + (0.5(\frac{1}{2}\log_2 0.5 + 0.5 \log_2 0.5))\right]$

$\Rightarrow H(Y_j) - 0.5$

$\Rightarrow 0.311$

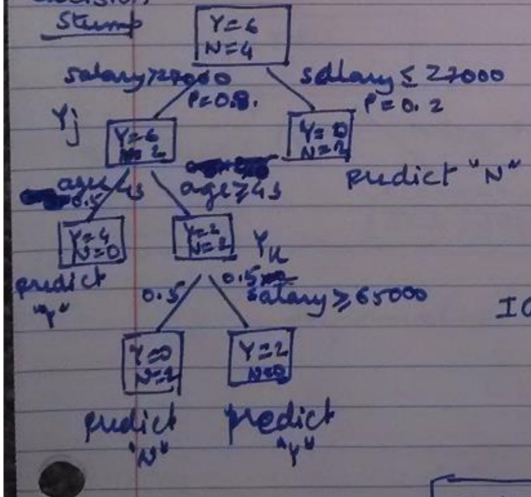
IG for choice 2 $\Rightarrow H(Y_j) - \left[-\frac{3}{8} \times 0 - \frac{5}{8}\left(\frac{2}{5}\log(2/5) + 3/5 \log(3/5)\right)\right]$

$\Rightarrow H(Y_j) - 0.6068$

$\Rightarrow 0.2042.$

Choice 1 has higher information gain $\Rightarrow \boxed{0.311}$

Decision Stump



Step 3:

After step 2 : there is just one intuitive ~~step~~ choice

→ salary ≥ 65000

$H(Y_k) = -1\left(\frac{2}{4}\log \frac{2}{4} + \frac{2}{4}\log \frac{2}{4}\right)$

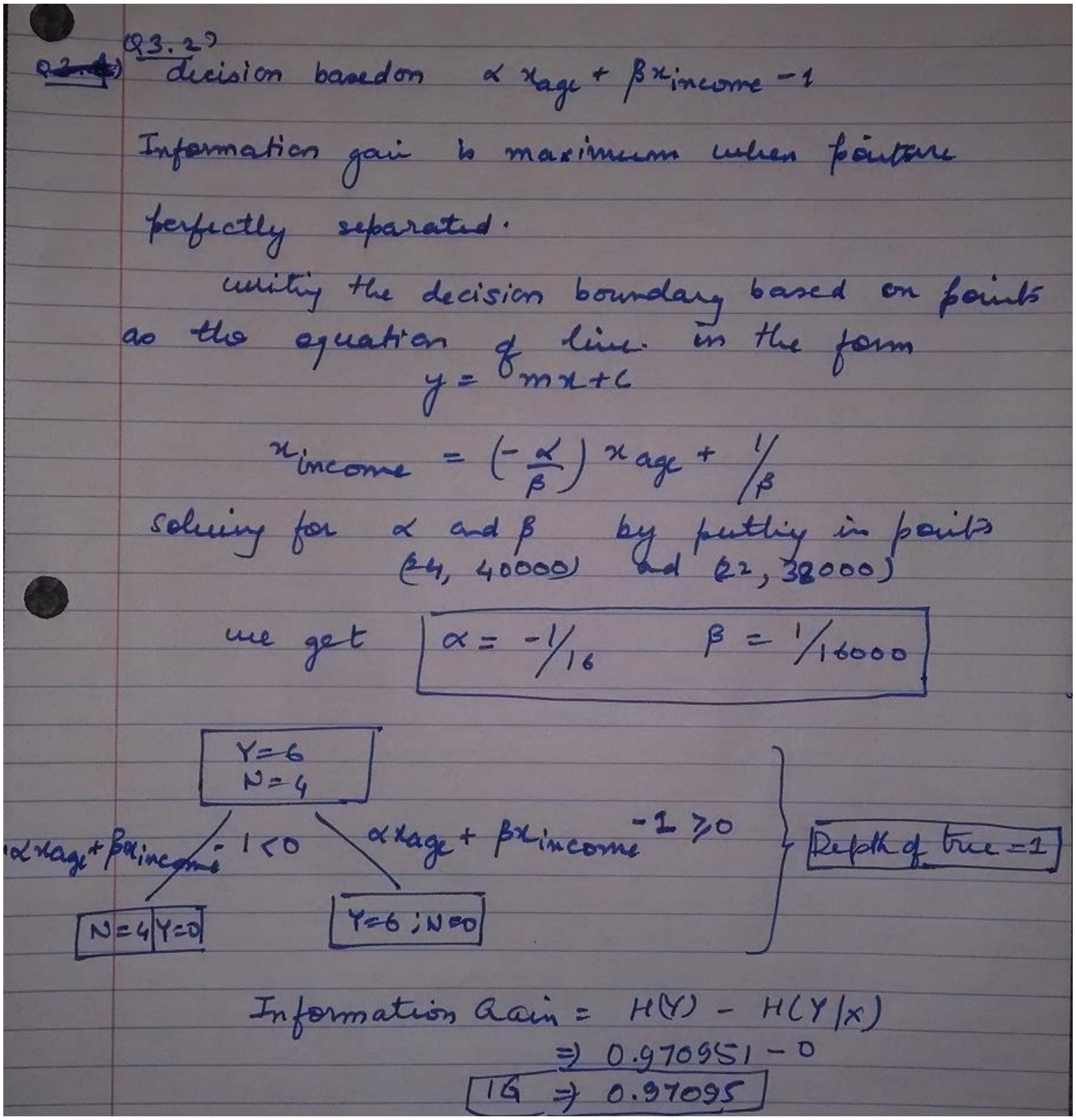
$\Rightarrow 1$

IG for the only choice

$\Rightarrow H(Y_k) - \left[\frac{1}{2} \times 0 + \frac{1}{2} \times 0\right]$

$\Rightarrow (1 - 0) . \Rightarrow \boxed{1}$

$\boxed{\text{Depth of Tree = 3}}$



Q3.2)

~~Q3.4)~~ decision basedon $\alpha x_{age} + \beta x_{income} - 1$

Information gain is maximum when feature perfectly separated.

writing the decision boundary based on points as the equation of line in the form

$$y = mx + c$$

$$x_{income} = \left(-\frac{\alpha}{\beta}\right) x_{age} + \frac{1}{\beta}$$

solving for α and β by putting in points (24, 40000) and (22, 38000)

we get $\boxed{\alpha = -1/16 \qquad \beta = 1/16000}$

Y=6
N=4

$\alpha x_{age} + \beta x_{income} - 1 < 0$

$\alpha x_{age} + \beta x_{income} - 1 \geq 0$

N=4 | Y=0

Y=6 ; N=0

Depth of tree = 1

Information Gain = $H(Y) - H(Y|x)$

$\Rightarrow 0.970951 - 0$

$\boxed{IG \Rightarrow 0.97095}$

Ques 3.3

Advantages of multivariate decision trees:

- More than one feature can be tested for per decision, this helps getting a way better predictor when the data points are linearly separable.
- Training error is lower as more complex models are allowed (decision based on multiple variables and rather than a single variable).
- Multivariate decision trees will have, on an average, smaller tree size.

Disadvantages of multivariate decision trees:

- The computation cost (CPU Time) of the learning algorithm is very high (imagine running linear regression again and again for each decision node), as a lot of features need to be tested to find out the maximum information gain per step. While the univariate algorithm requires considering only one feature at a time.

- On smaller datasets, multivariate trees tend to over-fit a lot because of standard deviation in the data points. This leads to higher test errors. Multivariate trees require a lot of data-points which might or might not be present.

Ques 4.4.1

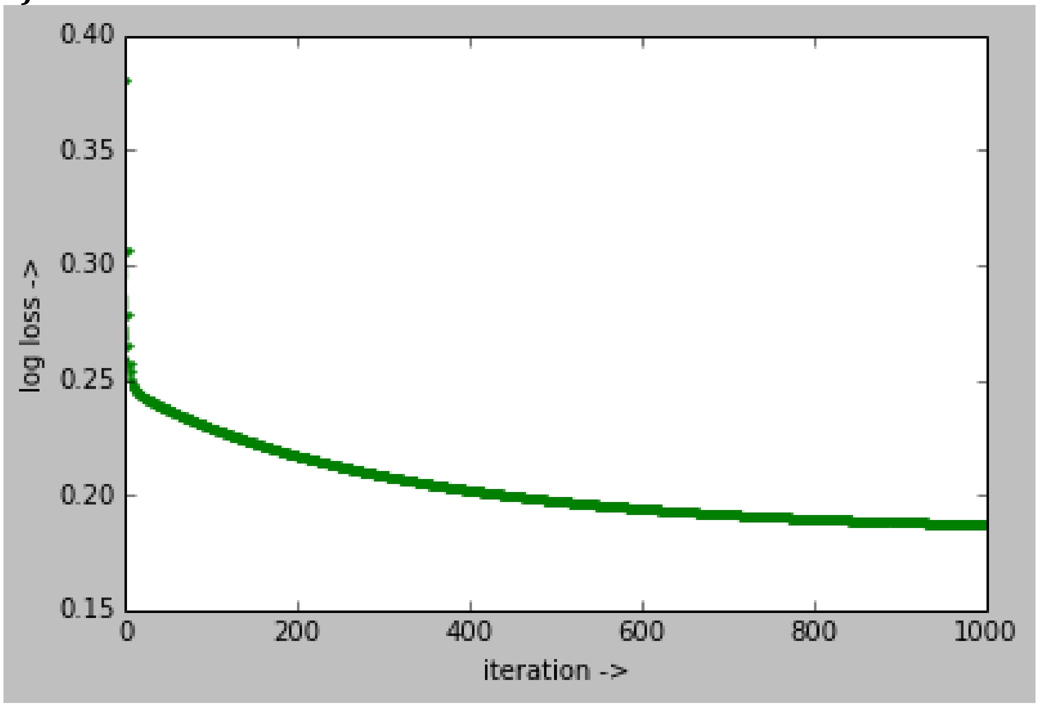
Actual weight update step:

$$w_i^{t+1} = w_i^t + \eta \left\{ -\lambda w_i^t + \frac{1}{N}\left(\sum_{j=1}^{N} x_i^j \left[y^j - \frac{e^{w_0+\sum w_i x_i^j}}{1 - e^{w_0+\sum w_i x_i^j}} \right] \right) \right\}$$

Weight update step as in python (Y is Nx1, X is Nx(D+1), W is (D+1)x1):

```
yj_minus_p = Y – (exp(X.dot(W))/(1+ exp(X.dot(W))))
W[0] = W[0] + (eta / N) * sum(yj_minus_p)
W[1:] = (1 - eta * lmbd) * W[1:] + eta * (array(yj_minus_p).dot(X[:,1:]) / N)
```

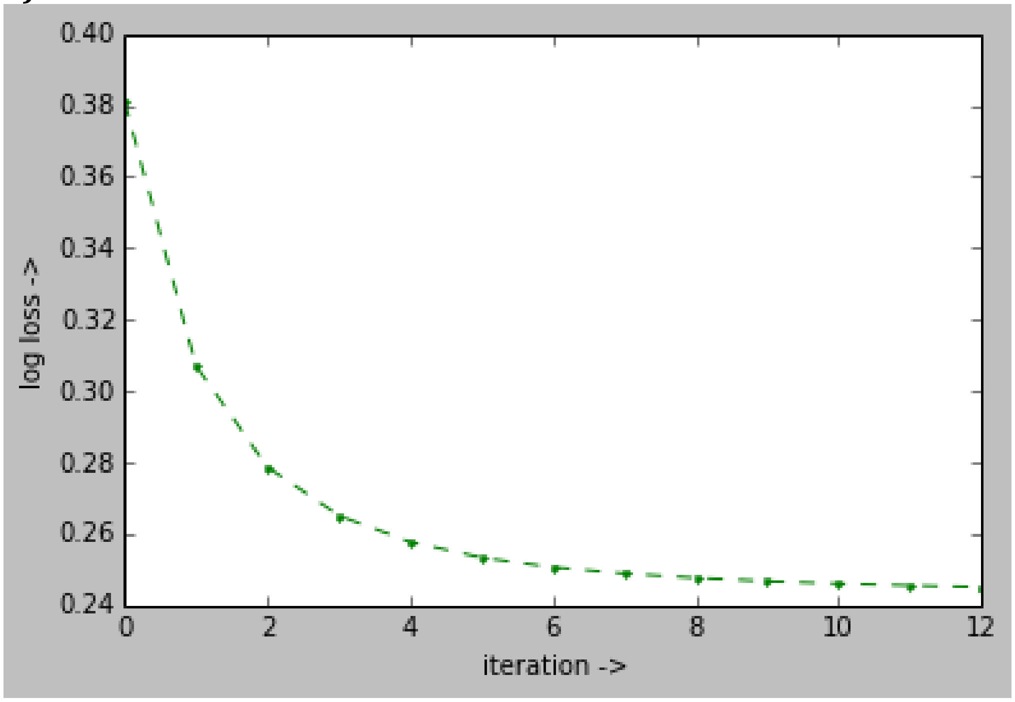
Ques 4.4.2
a)



b) SSE for batch gradient descent with 1000 iterations: 54.0

Ques 4.4.3 a) Number of iterations with the stopping criteria: 13

b)



c) SSE for batch gradient descent with the stop criteria: 54.0

Ques 4.5.1

Actual weight update step:

$$w_i^{t+1} = \ w_i^t + \eta \left\{ -\lambda w_i^t + x_i^j \left[y^j - \frac{e^{w_0 + \sum w_i x_i^j}}{1 - \ e^{w_0 + \sum w_i x_i^j}} \right] \right\}$$

Weight update step as in python (Y is Nx1, X is Nx(D+1), W is (D+1)x1):

```
yj_minus_p = Y[j] - (1-1/(1+exp(X[j,:].dot(W))))
W[0] = W[0] + (eta) * yj_minus_p
W[1:] = (1 - eta * lmbd) * W[1:] + eta * (yj_minus_p * (X[j,1:]))
```

Ques 4.5.2

a) L_2 Norm for lambda = 0 is: 1.92503456434
L_2 Norm for lambda = 0.3 is: 0.283520413611

b) SSE for lambda = 0.3 is: 54.0

c) Feature Weights for INTERCEPT: -3.10616785425
DEPTH: 0.109353101677
POSITION: -0.006094751226

Ques 4.5.3
After 5 iterations, log loss with:
Stochastic Descent: 0.197392978738
Gradient Descent: 0.257743788383

Stochastic Descent seems to converge faster.

Ques 4.6.1
For predictions made by SGD running with one pass over data and lambda = 0.3 and eta = 0.1:

Precision and Recall for class 0:	0.946,	1.0
Precision and Recall for class 1:	0,	0.0

Ques 4.6.2
For predictions made by batch gradient descent running for 10000 iterations over the oversampled data and with lambda = 0.3 and eta = 0.01:

Precision and Recall for class 0:	0.94140625	0.509513742072
Precision and Recall for class 1:	0.049180327	0.444444444444