PS10

Graded

Student

Chetan Hiremath

Total Points

90 / 100 pts

Question 1

10.1 — **10** / 10 pts

1.1 (a) — **2** / 2 pts

+ **+ 2 pts** Correct

1.2 (b) — **3** / 3 pts

+ **+ 3 pts** Correct

1.3 (c) — **2** / 2 pts

+ **+ 2 pts** Correct

1.4 (d) — **3** / 3 pts

+ **+ 3 pts** Correct

Question 2

10.2 — **10** / 10 pts

2.1 (a) — **3** / 3 pts

+ **+ 3 pts** Correct

2.2 (b) — **7** / 7 pts

+ **+ 7 pts** Correct

Question 3

10.3 **15** / 20 pts

3.1 (a) **5** / 5 pts

✔ **+ 5 pts** Correct

3.2 (b) **In Review** **10** / 15 pts

✔ **+ 15 pts** Correct

✔ **− 5 pts** Calculation incomplete/missing

Regrade Request **Submitted on: Apr 23**

What missing or incomplete calculations are you talking about? If my calculations are right, then I should get full points. Will you regrade this question?

Edit Request

Question 4

10.4 **5** / 5 pts

✔ **+ 5 pts** Correct

Question 5

10.5 **15** / 15 pts

5.1 (a) **5** / 5 pts

✔ **+ 5 pts** Correct

5.2 (b) **5** / 5 pts

✔ **+ 5 pts** Correct

5.3 (c) **5** / 5 pts

✔ **+ 5 pts** Correct

Question 6

10.6 — **35** / 40 pts

6.1 **(a)** — **10** / 10 pts

- ✔ **+ 10 pts** Correct

6.2 **(b)** — **10** / 10 pts

- ✔ **+ 10 pts** Correct

6.3 **(c)** — **In Review** — **15** / 20 pts

- ✔ **+ 20 pts** Correct
- ✔ **− 5 pts** Missing split/votes calculation/Incorrect

Regrade Request — **Submitted on: Apr 23**

How I have lost 5 points if my calculations are correct? Will you regrade this question?

Edit Request

1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

2a. Entropy of this set of training examples $= B\left(\frac{3}{3+3}\right) = B\left(\frac{3}{6}\right) = B\left(\frac{1}{2}\right) = -\left(\left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) + \left(1-\frac{1}{2}\right)\log_2\left(1-\frac{1}{2}\right)\right) = -\left(\frac{1}{2}(-1) + \frac{1}{2}(-1)\right) = -\left(-\frac{1}{2} - \frac{1}{2}\right) = -(-1) = 1.$

Question assigned to the following page: 1.1

1d. A and (not B) $\equiv A \wedge (\neg B)$

0 - No
1 - Yes

A	B	$A \wedge (\neg B)$
0	0	0
0	1	0
1	0	1
1	1	0

Question assigned to the following page: 5.1

5a. 1st-order Linear Langrange interpolating polynomial for points (1,2) and (3,5).

$$f(x) = \frac{(x-x_1)}{(x_0-x_1)} f(x_0) + \frac{(x-x_0)}{(x_1-x_0)} f(x_1)$$

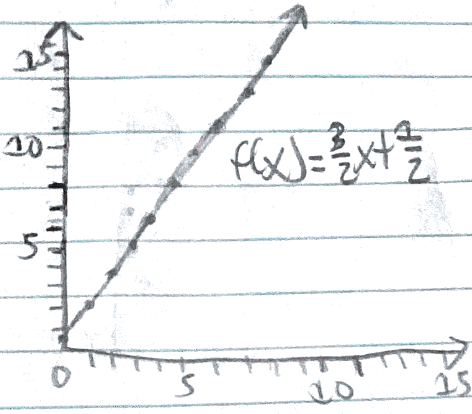
$$= \frac{x-3}{1-3}(2) + \frac{x-1}{3-1}(5)$$

$$= \frac{2(x-3)}{-2} + \frac{5(x-1)}{2}$$

$$= \frac{-3x-1}{-2}$$

$$= \frac{3x+1}{2}$$

$$= \frac{3}{2}x + \frac{1}{2}$$



Question assigned to the following page: [6.1]

6a.

Sources- This code is used, borrowed, and modified from DecisionStumpBase.ipynb's Python code and R&N Textbook's Remainder Sub-Routine Algorithm.

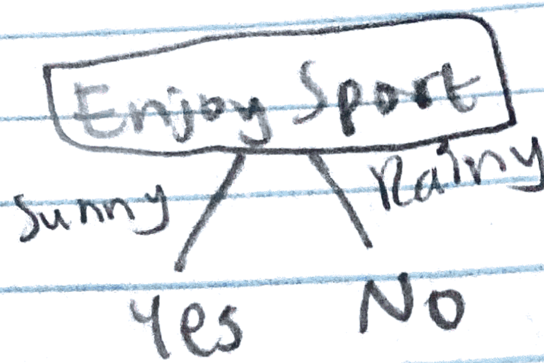
Remainder(A)'s Sub-Routine's Python Code-

```
pos1 = [ sum([table[j][0]==1 and table[j][i]==1   for j in
range(len(table))]) for i in classatt ]
neg1 = [ sum([table[j][0]==0 and table[j][i]==1   for j in
range(len(table))]) for i in classatt ]

remarr = np.array([])
for i in range(17):
    remainder = (pos[i]/N * B(pos1[i]/pos[i])) + (neg[i]/N *
B(neg1[i]/pos[i]))
    remarr = np.append(remarr, remainder)
print("Remainder values:")
print(remarr)
```

This sub-routine program calculates the remainder values of the Attributes 1-17 and includes them in the array. Attribute 1's remainder value is 0 since Attribute 1 is the class name with the entropy that is calculated without the remainder formula, but Attributes 2-17's values are the remainder values that are needed to calculate their respective information gain values. The modified program will be explained in the last part.

Question assigned to the following page: 3.1

3a.

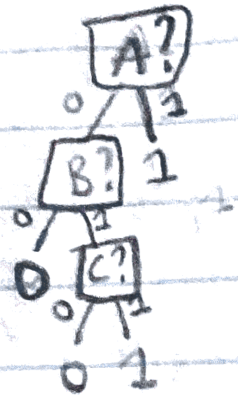


Decision tree

Question assigned to the following page: 1.2

b. A or (B and C) = $A \vee (B \wedge C)$

0-No
1-Yes



A	B	C	$A \vee (B \wedge C)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Question assigned to the following page: 5.2

b. 2^{nd}-order/Quadratic Langrange interpolating polynomial for points $(-1, 0.1)$, $(0, 0)$, and $(1, 0.1)$.

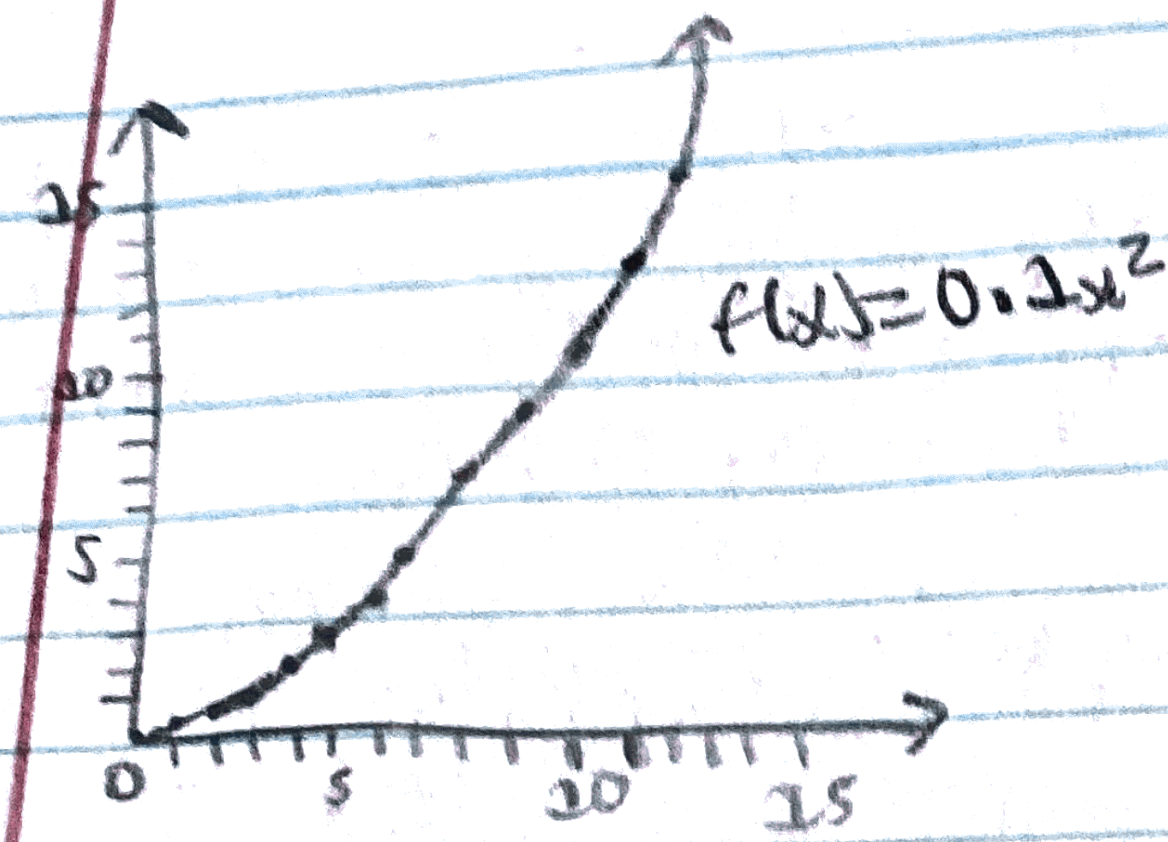
$$f(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1) + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} f(x_2)$$

$$= \frac{(x-0)(x-1)}{(-1-0)(-1-1)}(0.1) + \frac{(x-(-1))(x-1)}{(0-(-1))(0-1)}(0) + \frac{(x-(-1))(x-0)}{(1-(-1))(1-0)}(0.1)$$

$$= \frac{x(x-1)}{2}(0.1) + \frac{x(x+1)}{2}(0.1)$$

$$= \frac{0.1x(x-1) + 0.1x(x+1)}{2} = \frac{0.1x^2 + 0.1x^2}{2} = \frac{2(0.1x^2)}{2} = 0.1x^2$$

Question assigned to the following page: 5.2



$f(x) = 0.1x^2$
0
5
10
15
5
10
15

Question assigned to the following page: 6.2

b.

Sources- This code is used, borrowed, and modified from DecisionStumpBase.ipynb's Python code and R&N Textbook's Gain Sub-Routine Algorithm.

Gain(A)'s Sub-Routine's Python Code-

```
pos1 = [ sum([table[j][0]==1 and table[j][i]==1   for j in 
range(len(table))]) for i in classatt ]
neg1 = [ sum([table[j][0]==0 and table[j][i]==1   for j in 
range(len(table))]) for i in classatt ]

gainarr = np.array([])
for i in range(17):
    gain = B(pos[0]/N) - (pos[i]/N * B(pos1[i]/pos[i])) - (neg[i]/N * 
B(neg1[i]/pos[i]))
    gainarr = np.append(gainarr, gain)
print("Gain values:")
print(gainarr)
```

This sub-routine program calculates the gain values of the Attributes 1-17 and includes them in the array. The gain value of an attribute = the entropy - the remainder value of an attribute. Attribute 1's gain value is the entropy of this data set since Attribute 1 is the class name, but Attributes 2-17's values are the gain values that are calculated by remainder values of their respective attributes. The modified program will be explained in the last part.

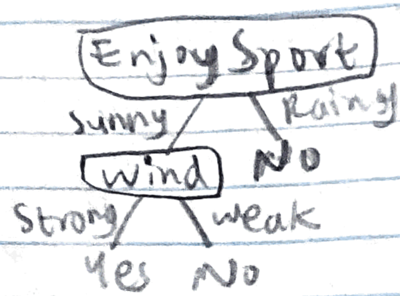
Question assigned to the following page: 2.2

b. $\text{Gain}(a2) = 1 - \frac{4}{6}B\left(\frac{2}{2+2}\right) - \frac{2}{6}B\left(\frac{1}{1+1}\right) = 1 - \frac{4}{6}B\left(\frac{1}{2}\right) - \frac{2}{6}B\left(\frac{1}{2}\right)$

$= 1 - \frac{4}{6}(1) - \frac{2}{6}(1) = 1 - \frac{4}{6} - \frac{2}{6} = 1 - 1 = 0.$

Question assigned to the following page: 3.2

b.



Decision Tree with a new training example

$Entropy = B\left(\frac{3}{3+2}\right) = B\left(\frac{3}{5}\right) = -\left(\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) + \left(1-\frac{3}{5}\right)\log_2\left(1-\frac{3}{5}\right)\right) = 0.971$

$Gain(Sky) = 0.971 - \frac{4}{5}B\left(\frac{3}{3+1}\right) - \frac{1}{5}B\left(\frac{1}{1}\right) = 0.971 - \frac{4}{5}B\left(\frac{3}{4}\right) - \frac{1}{5}B(1)$
$= 0.971 - \frac{4}{5}\left(-\left(\frac{3}{4}\right)\log_2\left(\frac{3}{4}\right) + \left(1-\frac{3}{4}\right)\log_2\left(1-\frac{3}{4}\right)\right) = 0.322$

$Gain(AirTemp) = 0.971 - \frac{4}{5}B\left(\frac{3}{3+1}\right) - \frac{1}{5}B\left(\frac{1}{1}\right) = 0.971 - \frac{4}{5}B\left(\frac{3}{4}\right) - \frac{1}{5}B(1)$
$= 0.971 - \frac{4}{5}\left(-\left(\frac{3}{4}\right)\log_2\left(\frac{3}{4}\right) + \left(1-\frac{3}{4}\right)\log_2\left(1-\frac{3}{4}\right)\right) = 0.322$

$Gain(Humidity) = 0.971 - \frac{3}{5}B\left(\frac{2}{2+1}\right) - \frac{2}{5}B\left(\frac{1}{1+1}\right) = 0.971 - \frac{3}{5}B\left(\frac{2}{3}\right) - \frac{2}{5}B\left(\frac{1}{2}\right)$
$= 0.971 - \frac{3}{5}\left(-\left(\frac{2}{3}\right)\log_2\left(\frac{2}{3}\right) + \left(1-\frac{2}{3}\right)\log_2\left(1-\frac{2}{3}\right)\right) - \frac{2}{5}(1) = 0.020$

$Gain(Wind) = 0.971 - \frac{4}{5}B\left(\frac{3}{3+1}\right) - \frac{1}{5}B\left(\frac{1}{1}\right) = 0.971 - \frac{4}{5}B\left(\frac{3}{4}\right) - \frac{1}{5}B(1)$
$= 0.971 - \frac{4}{5}\left(-\left(\frac{3}{4}\log_2\left(\frac{3}{4}\right) + \left(1-\frac{3}{4}\right)\log_2\left(1-\frac{3}{4}\right)\right)\right) = 0.322$

$Gain(Water) = 0.971 - \frac{4}{5}B\left(\frac{2}{2+2}\right) - \frac{1}{5}B\left(\frac{1}{1}\right) = 0.971 - \frac{4}{5}B\left(\frac{1}{2}\right) - \frac{1}{5}B(1)$
$= 0.971 - \frac{4}{5}(1) = 0.171$

$Gain(Forecast) = 0.971 - \frac{3}{5}B\left(\frac{2}{2+1}\right) - \frac{2}{5}B\left(\frac{1}{1+1}\right) = 0.971 - \frac{3}{5}B\left(\frac{2}{3}\right) - \frac{2}{5}B\left(\frac{1}{2}\right)$
$= 0.971 - \frac{3}{5}\left(-\left(\frac{2}{3}\right)\log_2\left(\frac{2}{3}\right) + \left(1-\frac{2}{3}\right)\log_2\left(1-\frac{2}{3}\right)\right) - \frac{2}{5}(1) = 0.020$

Choose Sky, AirTemp, or Wind as one of the best attributes since they have the same information gain value or 0.322 in the decision tree.

Question assigned to the following page: 6.3

c.

Sources- This code is used, borrowed, and modified from DecisionStumpBase.ipynb's Python code and R&N Textbook's Entropy, Remainder, and Gain Algorithms.

Decision Stump's Modified Python Code-

```
import numpy as np
from math import log2

N=232
def B(q):
  if q==0 or q==1:
    return 0
  return -(q*log2(q)+(1-q)*log2(1-q))

# test the entropy function, see p. 704 of R&N
for q in [0, 1, 0.5, 0.99, 9/(9+5)]:
  print("q = %.3f, B(q) = %.3f" % (q, B(q)) )
print()

classatt = range(17)
# count the positive, negative examples for classication and each attribute
pos=[ sum([table[j][i]    for j in range(len(table))]) for i in classatt ]
neg=[ sum([table[j][i]==0 for j in range(len(table))]) for i in classatt ]
```

Question assigned to the following page: 6.3

```
for a in classatt:
    print("%d: \t%d \t%d \t%.4f" % (a+1, pos[a], neg[a],
B(pos[a]/(pos[a]+neg[a]))))

#These upcoming portions are used to modify this program.
pos1 = [ sum([table[j][0]==1 and table[j][i]==1   for j in
range(len(table))]) for i in classatt ]
neg1 = [ sum([table[j][0]==0 and table[j][i]==1   for j in
range(len(table))]) for i in classatt ]

print()
print("Positive examples:")
print(pos1)
print("Negative examples:")
print(neg1)
print()

remarr = np.array([])
gainarr = np.array([])
```

Question assigned to the following page: 6.3

```
for i in classatt:
    remainder = (pos[i]/N * B(pos1[i]/pos[i])) + (neg[i]/N *
B(neg1[i]/pos[i]))
    gain = B(pos[0]/N) - (pos[i]/N * B(pos1[i]/pos[i])) - (neg[i]/N *
B(neg1[i]/pos[i]))
    remarr = np.append(remarr, remainder)
    gainarr = np.append(gainarr, gain)
print("Remainder values:")
print(remarr)
print("Gain values:")
print(gainarr)
print()
print("Decision Stump:")
for i in classatt:
    print("%d: \t%.i \t%.i \t%.4f \t%.4f" % (i+1, pos1[i], neg1[i],
remarr[i], gainarr[i]))
print("The Attribute Number with the highest gain value", max(gainarr[1:]),
"is", (np.argmax(gainarr[1:]) + 2))
```

Question assigned to the following page: 6.3

Result-

```
q = 0.000, B(q) = 0.000
q = 1.000, B(q) = 0.000
q = 0.500, B(q) = 1.000
q = 0.990, B(q) = 0.081
q = 0.643, B(q) = 0.940

1:    108   124   0.9966
2:    96    136   0.9784
3:    107   125   0.9957
4:    123   109   0.9974
5:    113   119   0.9995
6:    128   104   0.9923
7:    149   83    0.9408
8:    124   108   0.9966
9:    119   113   0.9995
10:   113   119   0.9995
11:   128   104   0.9923
12:   80    152   0.9294
13:   108   124   0.9966
14:   127   105   0.9935
15:   149   83    0.9408
16:   86    146   0.9512
17:   189   43    0.6916

Positive examples:
[108, 23, 51, 17, 107, 103, 94, 29, 16, 15, 62, 17, 92, 91, 106, 12, 72]
Negative examples:
[0, 73, 56, 106, 6, 25, 55, 95, 103, 98, 66, 63, 16, 36, 43, 74, 117]
```

Question assigned to the following page: 6.3

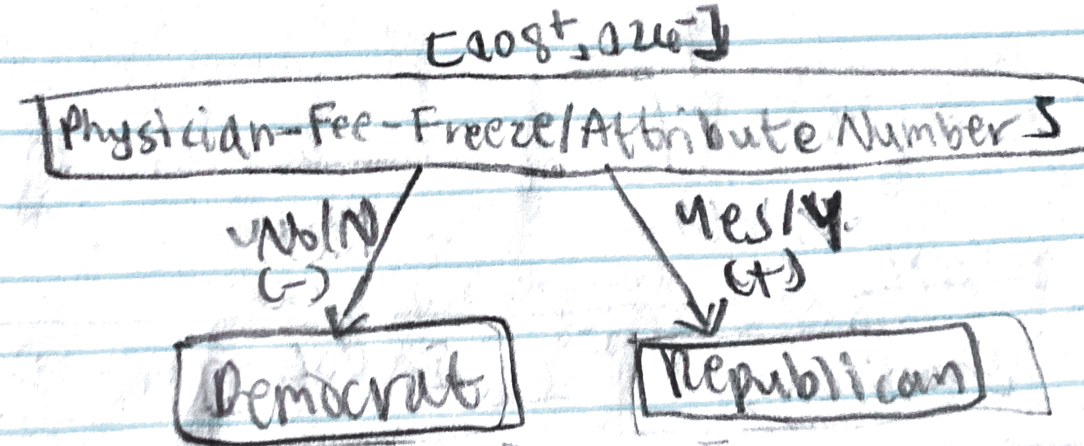
```
Remainder values:
[0.         0.79434669 0.99842429 0.57953535 0.29941133 0.71245344
 0.94999982 0.78469847 0.56952739 0.56491415 0.99929544 0.7462343
 0.60518658 0.86012868 0.86689367 0.58301942 0.95871188]
Gain values:
[ 0.99656637  0.20221968 -0.00185792  0.41703102  0.69715504  0.28411293
  0.04656655  0.2118679   0.42703898  0.43165222 -0.00272907  0.25033207
  0.39137979  0.13643769  0.1296727   0.41354695  0.03785449]

Decision Stump:
1:    108   0     0.0000      0.9966
2:    23    73    0.7943      0.2022
3:    51    56    0.9984      -0.0019
4:    17    106   0.5795      0.4170
5:    107   6     0.2994      0.6972
6:    103   25    0.7125      0.2841
7:    94    55    0.9500      0.0466
8:    29    95    0.7847      0.2119
9:    16    103   0.5695      0.4270
10:   15    98    0.5649      0.4317
11:   62    66    0.9993      -0.0027
12:   17    63    0.7462      0.2503
13:   92    16    0.6052      0.3914
14:   91    36    0.8601      0.1364
15:   106   43    0.8669      0.1297
16:   12    74    0.5830      0.4135
17:   72    117   0.9587      0.0379
The Attribute Number with the highest gain value 0.6971550420914682 is 5
```

Question assigned to the following page: [6.3]

This modified program calculates the remainder and the gain values of the Attributes 1-17 and includes them in separate array. Then, these values are calculated for the decision stump that is shown on the result. The first column has the attribute numbers, the second column has the positive examples/republicans of the respective attributes, the third column has the negative examples/democrats of the respective attributes, the fourth column has the remainder values of the respective attributes, and the fifth and the last column has the gain values of the respective attributes. The table variable that is big and long is not shown in this solution since it has 232 instances with the Boolean values, but I have used the table variable in the modified program to calculate the remainder and the gain values for the decision stump. The attribute with the highest gain value is Attribute 5/Physician-Fee-Freeze since its value is 0.6972, and Attribute 1 is not considered since it is a class name with the entropy that is used to calculate the gain values of the remaining attributes.

Question assigned to the following page: 6.3

6c.


[208+, 226-]
Physician-Fee-Freeze/Attribute Number 5
No/N
(-)
Yes/Y
(+)
Democrat
Republican

Question assigned to the following page: 5.3

c. 1st order / Linear least squares fit for points (-1, 0.1), (0, 0), and (1, 0.1).

x	$y=f(x)$	x^2	xy
-1	0.1	1	-0.1
0	0	0	0
1	0.1	1	0.1

$\Sigma x = -1+0+1 = 0$ $\quad \Sigma x^2 = 1+0+1 = 2$

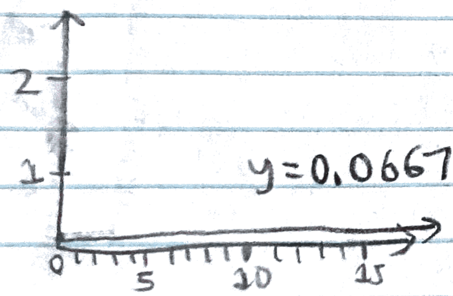
$\Sigma y = 0.1+0+0.1 = 0.2$ $\quad \Sigma xy = -0.1+0+0.1 = 0$

$n = 3$

$$m = \frac{n\Sigma xy - (\Sigma x \Sigma y)}{n\Sigma x^2 - (\Sigma x)^2} = \frac{3(0) - ((0)(0.2))}{3(2) - (0)^2} = 0$$

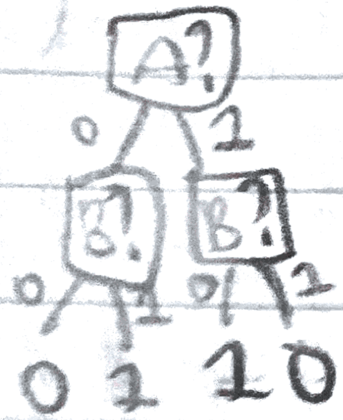
$$b = \frac{\Sigma y - m\Sigma x}{n} = \frac{0.2 - 0(0)}{3} = 0.0667$$

$y = mx + b = 0.0667$



Question assigned to the following page: 1.3

C. A xor $B = A \oplus B$



0-No
1-Yes

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Question assigned to the following page: 4

4. $P(error) = \sum_{x=\frac{k+1}{2}}^{k} \binom{k}{x} \varepsilon^x (1-\varepsilon)^{k-x}$ when k is the number of learned hypotheses, and ε is the error of one hypothesis. $\binom{k}{x} = \frac{k!}{x!(k-x)!}$ shows the number of distinct ways to choose k objects from x objects.

Use $k=5$, and $\varepsilon=0.1$ for this formula

$$P(error) = \sum_{x=\frac{k+1}{2}}^{k} \binom{k}{x} \varepsilon^x (1-\varepsilon)^{k-x}$$

$$= \sum_{x=3}^{5} \binom{5}{x} 0.1^x (1-0.1)^{5-x}$$

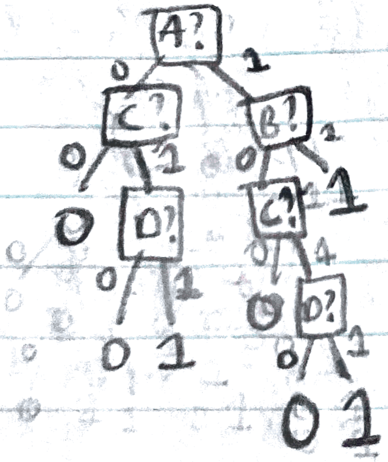
$$= \left(\binom{5}{3} 0.1^3 (1-0.1)^{5-3}\right) + \left(\binom{5}{4} 0.1^4 (1-0.1)^{5-4}\right) + \left(\binom{5}{5} 0.1^5 (1-0.1)^{5-5}\right)$$

$$= 0.00810 + 0.00045 + 0.00001$$

$$= 0.00856$$

Question assigned to the following page: 1.4

d. (A and B) or (C and D) = $(A \wedge B) \vee (C \wedge D)$



0 - No
1 - Yes

A	B	C	D	$(A \wedge B) \vee (C \wedge D)$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1