CS158 Midterm 1

Collins Munene Kariuki

TOTAL POINTS

27.25 / 34

QUESTION 1

1 T/F 7 / 10

- **0 pts** Correct
- **1 pts** One T/F incorrect
- **2 pts** Two T/F incorrect

✓ **- 3 pts** *Three T/F incorrect*

- **4 pts** Four T/F incorrect

1. F
2. T
3. T

QUESTION 2

2 Perceptron learning 6 / 6

✓ **- 0 pts** *Correct*

- **2 pts** (d)
- **0.75 pts** (d) missing b value
- **1.5 pts** (b) no change since we got the prediction correct
- **1.5 pts** (c) prediction is +1
- **1 pts** (d) w1 wrong

QUESTION 3

3 Bad hair day 3.75 / 6

- **0 pts** Correct

✓ **- 0.75 pts** *(a) not DT*

- **0.75 pts** (a) not k-NN

✓ **- 0.75 pts** *(b) not k-NN*

✓ **- 0.75 pts** *(b) decision trees*

- **0.25 pts** (c) perceptron is also amenable to adding additional examples
- **0.75 pts** (c) k-NN
- **0.75 pts** (a) perceptron
- **0.75 pts** (c) not DT

QUESTION 4

4 Trees 7 / 7

✓ **- 0 pts** *Correct*

- **1 pts** (a) denominator is 6 not 5
- **1 pts** (b) Don't split on procrastinate = S.
- **1 pts** (a) procrastinate 1/6
- **1 pts** (b) split on procrastinate first
- **1 pts** (a) do problems is 2/6
- **1 pts** (b) wrong predictions for procrastinate = N and Y
- **0.25 pts** (a) Training error, not accuracy
- **0.5 pts** (a) Training errors, so less is better. Procrastinate should be chosen.

QUESTION 5

5 Feature normalization 3.5 / 5

- **0 pts** Correct

✓ **- 1.5 pts** *Use the _training data_ means on the test data.*

1. **[10 points]** T/F: For each of the statements below state whether they are true or false. You do not need to justify your answer.

F Without depth limiting or pruning (i.e. building the full tree) decision trees will always achieve 0% training error.

T When doing proper experimentation, hyperparameters should be tuned on the development data set.

T 1 If the number of labels > 4 AVA always takes longer to classify than OVA since it involves classifying with more binary classifiers.

F If we're utilizing mean centering with variance scaling as our feature pre-processor, then the mean of each feature of the test data will be 0 after pre-processing.

T If our features are booleans the height of the decision tree can be no larger than the number of features.

F 2 If we let k = the number of training examples for k-NN, then we will always predict the overall majority class label.

F In most situations k-NN is faster to classify than Decision Trees.

F 3 The average perceptron algorithm uses on the order of the same magnitude of memory during training as the perceptron algorithm.

F If the recall of a classifier is 100% on a test set *and* the test set contains at least 1 positive example, then the precision will also be 100%.

T For gradient descent with regularization, it does not make sense to have set $\lambda < 0$.

2. **[6 points]** Perceptron learning

prediction $= w_1 f_1 + w_2 f_2$

Given initial weights $w = [1, -1]$ and $b = 0$ for a linear model over two features:

(a) What is the prediction (+1 or −1) for the example: $[1, 0]$ with *label* $= +1$?

$w_1 = 1$ $f_1 = 1$ $1(1) + (-1)(0) = 1 + 0 = 1$ → correct prediction

$w_2 = -1$ $f_2 = 0$ so prediction $= +1$

(b) If this example was your only training example, what would w and b be if you started at the initial weights above after **one** iteration of the perceptron learning algorithm.

- Given that the initial weights resulted in the correct prediction, we should not alter the initial weights.

Answer: $b = 0$, $w = [1, -1]$

(c) What is the prediction (+1 or −1) for the example: $[0, -1]$ with *label* $= -1$ based on the *initial weights* (i.e., $w = [1, -1]$ and $b = 0$, **NOT** the weights you found in part 2b)?

prediction $= w_1 f_1 + w_2 f_2$ $(1)(0) + (-1)(-1) = 0 + 1 = 1$

$w_1 = 1$ and $w_2 = -1$

$f_1 = 0$ and $f_2 = -1$ prediction $= +1$ (the wrong prediction)

(d) If this example was your only training example, what would w and b be if you started at the *initial weights* above after **one** iteration of the perceptron learning algorithm.

- The prediction is wrong so we need to change the weights and bias according to:
- for each w_i:

  $w_i = w_i + f_i * \text{label}$ and $b = 0 + (-1) = -1$ (We do this only once, not for every weight)

  so $w_1 = 1 + (0 * -1) = 1$

  and $w_2 = -1 + (-1 * -1) = -1 + 1 = 0$
- Thus, finally: $w = [1, 0]$ and $b = -1$

1 T/F **7 / 10**

- **0 pts** Correct
- **1 pts** One T/F incorrect
- **2 pts** Two T/F incorrect

✓ **- 3 pts** *Three T/F incorrect*

- **4 pts** Four T/F incorrect

1. F
2. T
3. T

F 3 The average perceptron algorithm uses on the order of the same magnitude of memory during training as the perceptron algorithm.

F If the recall of a classifier is 100% on a test set *and* the test set contains at least 1 positive example, then the precision will also be 100%.

T For gradient descent with regularization, it does not make sense to have set $\lambda < 0$.

2. **[6 points]** Perceptron learning

prediction $= w_1 f_1 + w_2 f_2$

Given initial weights $w = [1, -1]$ and $b = 0$ for a linear model over two features:

(a) What is the prediction (+1 or −1) for the example: $[1, 0]$ with *label* $= +1$?

$w_1 = 1$ $f_1 = 1$ $1(1) + (-1)(0) = 1 + 0 = 1$ → correct prediction

$w_2 = -1$ $f_2 = 0$ so prediction $= +1$

(b) If this example was your only training example, what would w and b be if you started at the initial weights above after **one** iteration of the perceptron learning algorithm.

- Given that the initial weights resulted in the correct prediction, we should not alter the initial weights.

Answer: $b = 0$, $w = [1, -1]$

(c) What is the prediction (+1 or −1) for the example: $[0, -1]$ with *label* $= -1$ based on the *initial weights* (i.e., $w = [1, -1]$ and $b = 0$, **NOT** the weights you found in part 2b)?

prediction $= w_1 f_1 + w_2 f_2$ $(1)(0) + (-1)(-1) = 0 + 1 = 1$

$w_1 = 1$ and $w_2 = -1$

$f_1 = 0$ and $f_2 = -1$ prediction $= +1$ (the wrong prediction)

(d) If this example was your only training example, what would w and b be if you started at the *initial weights* above after **one** iteration of the perceptron learning algorithm.

- The prediction is wrong so we need to change the weights and bias according to:
- for each w_i:

  $w_i = w_i + f_i * \text{label}$ and $b = 0 + (-1) = -1$ (We do this only once, not for every weight)

  so $w_1 = 1 + (0 * -1) = 1$

  and $w_2 = -1 + (-1 * -1) = -1 + 1 = 0$
- Thus, finally: $w = [1, 0]$ and $b = -1$

2 Perceptron learning 6 / 6

✓ **- 0 pts** *Correct*

- 2 pts (d)

- 0.75 pts (d) missing b value

- 1.5 pts (b) no change since we got the prediction correct

- 1.5 pts (c) prediction is +1

- 1 pts (d) w1 wrong

3. **[6 points]** Bad hair day

You're developing a new mobile app that takes a picture of you and determines whether or not you're having a good hair day. To do this, you've collected over a million examples each consisting of around 10,000 features. You're trying to decide which of our three machine learning approaches will work out for you. You're going to train the models on a normal computer, but all of the classifying needs to happen on the mobile device.

(a) **[2 points]** Since you're working on a mobile device, memory is constrained to within an order of magnitude of the number of features. You talk to an expert and, unfortunately, most of the features are important. Given this, which of the following three classifiers are still applicable if you're only concerned with the memory required for classification (circle all that are appropriate):

- decision trees
- perceptron
- k-NN

(b) **[2 points]** You decide that run-time is the most important factor and you need it to be roughly linear in the size of the input example. Based only on classification run-time, which of the following approaches would work for you (circle all that are appropriate):

- decision trees
- perceptron
- k-NN

(c) **[2 points]** You decide that you want to add the ability for users to take and label their own pictures and then update the underlying model on the fly. You don't want to have to do the full training of the model every time the user adds new data. Assuming you can store all of the training data, which of the following approaches would allow you to support on the fly updates with minimal computation that would end up with roughly the same performance as if you'd trained the model from scratch on all of the data (circle all that are appropriate):

- decision trees
- perceptron → Assuming the new data will make the larger dataset (as a whole) separable.
- k-NN

Remember to read book. do problems

3 Bad hair day **3.75 / 6**

- **0 pts** Correct

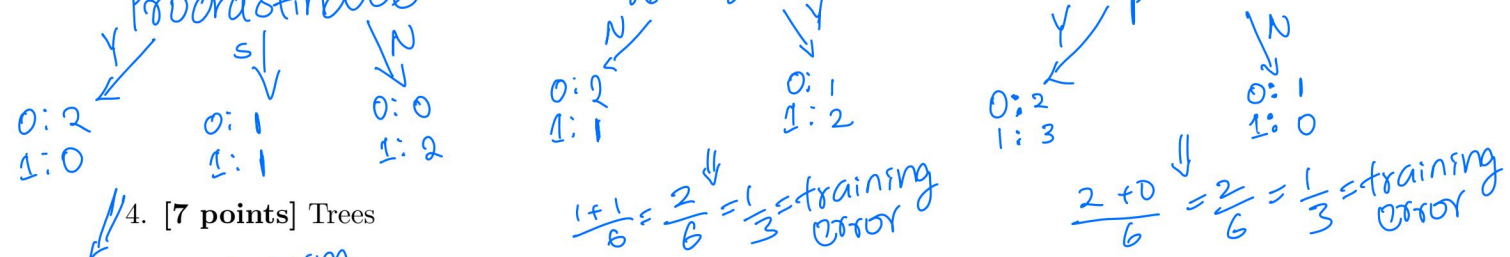
✓ **- 0.75 pts** *(a) not DT*

- **0.75 pts** (a) not k-NN

✓ **- 0.75 pts** *(b) not k-NN*

✓ **- 0.75 pts** *(b) decision trees*

- **0.25 pts** (c) perceptron is also amenable to adding additional examples
- **0.75 pts** (c) k-NN
- **0.75 pts** (a) perceptron
- **0.75 pts** (c) not DT



4. **[7 points]** Trees

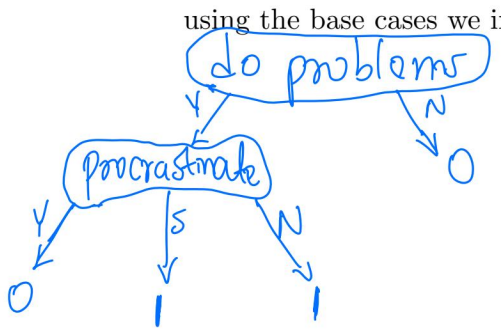
procrastinate	read book	do problems	pass ML
Y	N	Y	0
S	Y	Y	1
N	N	Y	1
S	N	N	0
N	Y	Y	1
Y	Y	Y	0

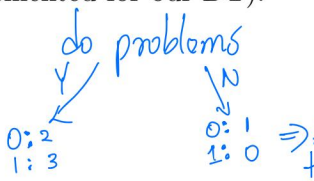
(a) **[3 points]** For the training data above, fill in the table below with the training error if we split on each feature individually. For the `procrastinate` feature assume that the tree would do a ternary split. *Circle the feature that would be chosen for the root of the decision tree?* (For partial credit, show your work.)

→work is shown above.

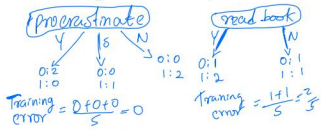
procrastinate	1/6
read book	1/3
do problems	1/3

(b) **[3 points]** Draw the decision tree that would be learned if we selected `do problems` as the root and used training error as the splitting criterion after that (with no depth limit, using the base cases we implemented for our DT).





⇒All examples have the same label.



↑ when we split when do problems = Y.

→The reason we can't split after this is because all data belong to the same class for all of Y, S, and N for procrastinate.

(c) **[1 point]** What would this decision tree classify `[procrastinate=N, read book=Y, do problems=N]` as?

Answer = 0

4 Trees 7 / 7

✓ **- 0 pts** *Correct*

- 1 pts (a) denominator is 6 not 5

- 1 pts (b) Don't split on procrastinate = S.

- 1 pts (a) procrastinate 1/6

- 1 pts (b) split on procrastinate first

- 1 pts (a) do problems is 2/6

- 1 pts (b) wrong predictions for procrastinate = N and Y

- 0.25 pts (a) Training error, not accuracy

- 0.5 pts (a) Training errors, so less is better. Procrastinate should be chosen.

$\text{mean}(f_1) = \frac{1+0+0+0+1+1}{6} = \frac{3}{6} = 0.5$

$\text{mean}(f_2) = \frac{10+0+5+10+20+15}{6} = \frac{60}{6} = 10$

$\text{mean}(f_1) = \frac{0+1}{2} = 0.5$

$\text{mean}(f_2) = \frac{20+10}{2} = \frac{30}{2} = 15$

5. **[5 points]** Feature normalization

Training data

f_1	f_2	label
1	10	1
0	0	-1
0	5	1
0	10	1
1	20	-1
1	15	-1

$10-10=0$, $5-10=-5$, $20-10=10$, $15-10=5$

$1-0.5=0.5$, $0-0.5=-0.5$

Test data

f_1	f_2	label
0	20	1
1	10	-1

$0-0.5=-0.5$, $1-0.5=0.5$

$20-15=5$, $10-15=-5$

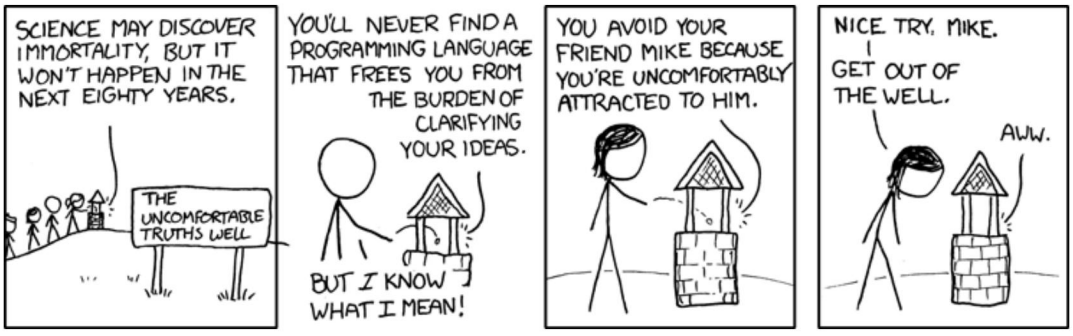
Fill in the tables below for the resulting data set after applying mean centering.

Training data

f_1	f_2	label
0.5	0	1
-0.5	-10	-1
-0.5	-5	1
-0.5	0	1
0.5	10	-1
0.5	5	-1

Test data

f_1	f_2	label
-0.5	5	1
0.5	-5	-1



http://xkcd.com/568/

5 Feature normalization 3.5 / 5

- **0 pts** Correct

✓ **- 1.5 pts** *Use the _training data_ means on the test data.*

CS158 - Midterm Exam 1
To be done by *11:59pm* on Friday, October 6

Name: Collins Kariuki

1	2	3	4	5	Total :
10	6	6	7	5	37

- You may use up to **2 hours** on this exam. Please time yourself and record your start and end times below:

  Start time: 3:53 pm **End time:** 5.49 pm

- You must finish this exam by **Friday, October 6th at 11:59 pm**.
- You must submit your exam to gradescope within **2.5 hours** after downloading it.
- You may upload your answers in any reasonable format.
- You *may* use your notes, the class notes, the class book, and your assignments.
- You *may NOT* use any other resources on the web or search for things on the web.
- You may not discuss this exam with **anyone** until after Friday.
- If there is any ambiguity on a question, state your assumptions clearly.