HOMEWORK B
DATA STRUCTURES, RECURSION, DYNAMIC PROGRAMMING[1]

CMU 10-607: COMPUTATIONAL FUNDAMENTALS FOR MACHINE LEARNING (FALL 2018)

https://piazza.com/cmu/fall2018/10606607

OUT: Nov. 08, 2018

DUE: Nov. 20, 2018 11:59 PM

START HERE: Instructions

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., "Jane explained to me what is asked in Question 2.1"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only. See the Academic Integrity Section on the course site for more information: http://www.cs.cmu.edu/~mgormley/courses/606-607-f18/about.html#7-academic-integrity-policies
- **Late Submission Policy:** See the late submission policy here: http://www.cs.cmu.edu/~mgormley/courses/606-607-f18/about.html#6-general-policies
- **Submitting your work to Gradescope:** For written problems such as short answer, multiple choice, derivations, proofs, or plots, we will be using Gradescope (https://gradescope.com/). Please use the provided template. Submissions can be handwritten onto the template, but should be labeled and clearly legible. If your writing is not legible, you will not be awarded marks. Alternatively, submissions can be written in LaTeX. Regrade requests can be made, however this gives the TA the opportunity to regrade your entire paper, meaning if additional mistakes are found then points will be deducted. Each derivation/proof should be completed on a separate page. For short answer questions you **should not** include your work in your solution. If you include your work in your solutions, your assignment may not be graded correctly by our AI assisted grader.

For multiple choice or select all that apply questions, shade in the box or circle in the template document corresponding to the correct answer(s) for each of the questions. For LaTeXusers, use ■ and ●for shaded boxes and circles, and don't change anything else.

[1]Compiled on Friday 16th November, 2018 at 15:51

Instructions for Specific Problem Types

For "Select One" questions, please fill in the appropriate bubble completely:

Select One: Who taught this course?

● Matt Gormley

○ Marie Curie

○ Noam Chomsky

If you need to change your answer, you may cross out the previous answer and bubble in the new answer:

Select One: Who taught this course?

● Matt Gormley

○ Marie Curie

⊗ Noam Chomsky

For "Select all that apply" questions, please fill in all appropriate squares completely:

Select all that apply: Which are scientists?

■ Stephen Hawking

■ Albert Einstein

■ Isaac Newton

□ I don't know

Again, if you need to change your answer, you may cross out the previous answer(s) and bubble in the new answer(s):

Select all that apply: Which are scientists?

■ Stephen Hawking

■ Albert Einstein

■ Isaac Newton

⊠ I don't know

For questions where you must fill in a blank, please make sure your final answer is fully included in the given space. You may cross out answers or parts of answers, but the final answer must still be within the given space.

Fill in the blank: What is the course number?

10-606	10-~~7~~06

1 Perceptrons (Programming) [12 pts]

In the accompanying file `perceptron.py`, fill out code to complete the provided functions for perceptron prediction and weight updates.

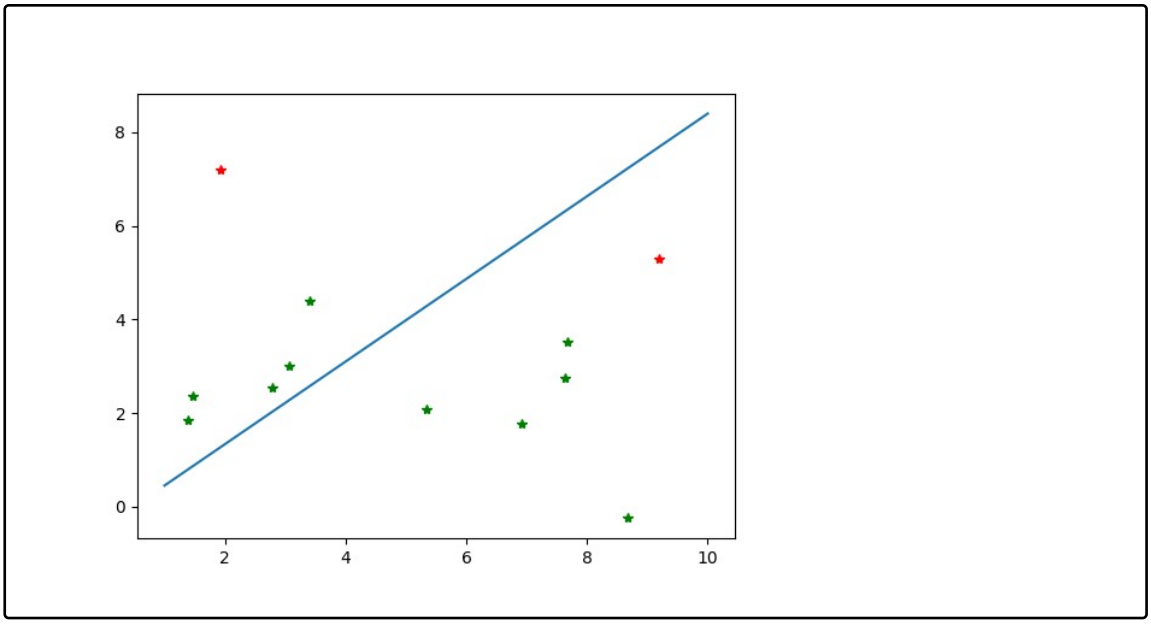
1. **[1 pts]** How many epochs does it take for the perceptron training error (which can be read from console output of `perceptron.py`) to converge to 0.0 on the linearly separable training data i.e. `train_separable`?

   3

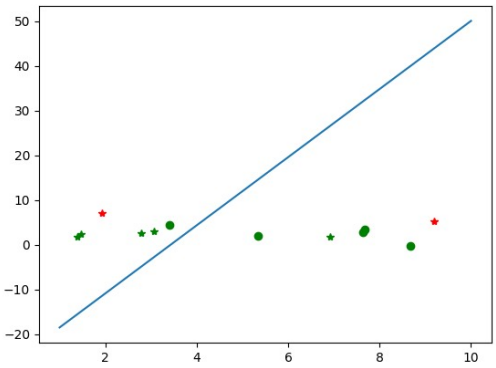
2. **[1 pts]** After training on the linearly separable dataset, how many errors does the perceptron make on the test data `test`?

   0

3. **[3 pts]** Plot the two-dimensional data with the training datapoints `train_separable` in green and test datapoints `test` in red. Also include in the plot a line indicating the separating decision boundary learned by the perceptron training process.



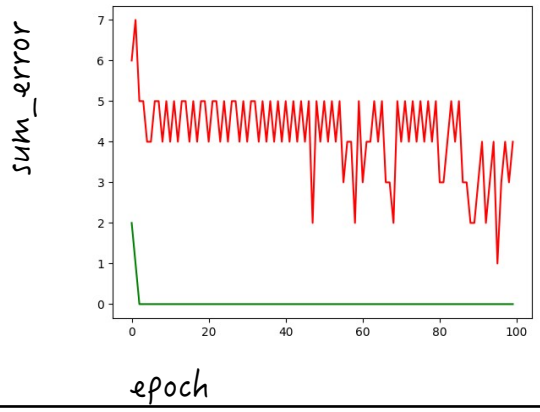
4. **[3 pts]** Plot the two-dimensional data with the training datapoints `train_nonseparable` in green and test datapoints `test` in red. Also include in the plot a line indicating the separating decision boundary learned by the perceptron training process for the non-separable training data.



5. **[1 pts]** In our implementation, you will notice that we have allowed the algorithm to run for a specific maximum number of epochs. In theory i.e. without any implementation-specific bound on the maximum number of epochs, will the perceptron algorithm converge to a training error rate of 0 for the training data `train_nonseparable` that is not linearly separable? If yes, how many epochs will it take for the algorithm to converge?

No, it will not converge

6. **[3 pts]** Plot the training error versus training epoch for the two datasets `train_separable` and `train_nonseparable` in a single plot. You can get these quantities from the console output when you run the `perceptron.py` file after coding the perceptron predict and update rules.



Red line is non-separable training set, while green line is separable training set.

2 Computational Complexity [21 pts]

1. **[1 pts]** Is $4n^3 + n^2 + n$ in $O(n^2)$? **Select one:**

   ○ Yes

   ● No

2. **[1 pts]** Is $4n^3 + n^2 + n$ in $O(n^3)$? **Select one:**

   ● Yes

   ○ No

3. **[1 pts]** Is $4n^3 + n^2 + n$ in $O(n^4)$? **Select one:**

   ● Yes

   ○ No

4. **[4 pts]** Prove that n^4 is in $O(n^5)$.

   1. Pick C = 1
   2. $F(x) = n^5/n^4 = n$
   3. when $n > 1$, $F(n) > 1$, $n^5 > n^4$
   4. So n^4 is in $O(n^5)$

5. **[4 pts]** Prove that $4n^4 + 3n^3 + 2n^2 + n \log n$ is in $O(n^4)$?

   1. Pick C = 10, when $n > 1$,
      $4n^4 + 3n^3 + 2n^2 + n < 4n^4 + 3n^4 + 2n^4 + n^4 = 10n^4$
   2. So $4n^4 + 3n^3 + 2n^2 + n^1$ is in $O(n^4)$.

6. **[4 pts]** Prove that $n^2 \log n$ is NOT in $O(n^2)$.

1. Assume exist n_0 and C s.t. when $n > n_0$, $n^2 \log n$ is in $O(n^2)$
2. Thus $n^2 \log n < Cn^2$
   -> $\log n < C$ ($n < e^c$)
3. 1&2 contradict, n_0 does not exit.

So the assumption is not valid.

7. **[2 pts]** What is the time complexity of following code?

```
a = 0
for i in range(N):
    for j in range(N, i, -1):
        a = a + i + j
```

Select one:

- ○ $O(n)$
- ○ $O(n \log n)$
- ○ $O(n\sqrt{n})$
- ● $O(n^2)$

8. **[2 pts]** What is the worst-case time complexity of following code?

```
for index in range(1,len(alist)):

    currentvalue = alist[index]
    position = index

    while position>0 and alist[position-1]>currentvalue:
        alist[position]=alist[position-1]
        position = position-1

    alist[position]=currentvalue
```

Select one:

- ○ $O(1)$
- ○ $O(n)$
- ● $O(n^2)$
- ○ $O(n \log n)$

9. **[2 pts]** What is the worst-case time complexity of following code?

```
for passnum in range(len(alist)-1,0,-1):
    for i in range(passnum):
        if alist[i]>alist[i+1]:
            temp = alist[i]
            alist[i] = alist[i+1]
            alist[i+1] = temp
```

Select one:

- ○ $O(1)$
- ○ $O(n)$
- ● $O(n^2)$
- ○ $O(n \log n)$

3 Dynamic Programming with Markov Chains (Theory) [13 pts]

Consider four discrete random variables X_1, X_2, X_3, and X_4. Each of them can take 3 possible values $1, 2, 3$. These random variables are linked in a chain: $X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_4$. Such a chain is called a Markov chain because each random variable in the chain is conditionally independent of all its ancestors conditioned on its parent random variable. For example, $X_4 \perp\!\!\!\perp X_1, X_2 \mid X_3$ i.e. $P(X_4 \mid X_1, X_2, X_3) = P(X_4 \mid X_3)$. You are further given that

$$
\begin{aligned}
P(X_1) &= [0.1, 0.3, 0.6] \\
P(X_{i+1}|X_i = 1) &= [0.3, 0.5, 0.2] \qquad \text{for } i = 1, 2, 3 \\
P(X_{i+1}|X_i = 2) &= [0.37, 0.33, 0.3] \quad \text{for } i = 1, 2, 3 \\
P(X_{i+1}|X_i = 3) &= [0.3, 0.5, 0.2] \qquad \text{for } i = 1, 2, 3
\end{aligned}
$$

1. **[2 pts]** Compute $P(X_1 = 1, X_2 = 3, X_3 = 2, X_4 = 1)$. **Hint:** Use chain rule of probability.

   Show your work:

   P = P(X1=1) x P(X2=3 | X1=1) x P(X3=2 | X2=3) x P(X4=1 | X3=2)
   = 0.1 x 0.2 x 0.5 x 0.37
   = 0.0037

   Write the final answer:

   0.0037

2. **[2 pts]** Compute $P(X_1 = 1, X_2 = 3, X_3 = 2, X_4 = 2)$. Notice that a substantial part of the computation overlaps with the previous question.

   Show your work:

   $P = P(X1=1) \times P(X2=3 \mid X1=1) \times P(X3=2 \mid X2=3) \times P(X4=2 \mid X3=2)$
   $= 0.1 \times 0.2 \times 0.5 \times 0.33$
   $= 0.0033$

   Write the final answer:

   0.0033

3. **[3 pts]** Compute $P(X_1 = 1, X_2 = 3, X_3 = 2)$. **Hint:** $P(X_4|X_3 = 2)$ is a function of X_4 alone. The complicated way is to calculate $P(X_1 = 1, X_2 = 3, X_3 = 2, X_4 = 1)$, $P(X_1 = 1, X_2 = 3, X_3 = 2, X_4 = 2)$, and $P(X_1 = 1, X_2 = 3, X_3 = 2, X_4 = 3)$ and add the three together to marginalize out X_4. A simpler way is to observe that $P(X_1 = 1, X_2 = 3, X_3 = 2) = \sum_{x_4} P(X_1 = 1, X_2 = 3, X_3 = 2, X_4 = x_4) = \sum_{x_4} P(X_1 = 1)P(X_2 = 3 \mid X_1 = 1)P(X_3 = 2 \mid X_1 = 1, X_2 = 3)P(X_4 = x_4 \mid X_1 = 1, X_2 = 3, X_3 = 2) = \sum_{x_4} P(X_1 = 1)P(X_2 = 3 \mid X_1 = 1)P(X_3 = 2 \mid X_2 = 3)P(X_4 = x_4 \mid X_3 = 2)$ The last result is from the conditional independence since we are dealing with a Markov chain. Finally, $\sum_{x_4} P(X_1 = 1)P(X_2 = 3 \mid X_1 = 1)P(X_3 = 2 \mid X_2 = 3)P(X_4 = x_4 \mid X_3 = 2) = P(X_1 = 1)P(X_2 = 3 \mid X_1 = 1)P(X_3 = 2 \mid X_2 = 3) \sum_{x_4} P(X_4 = x_4 \mid X_3 = 2)$ As a result, $P(X_1 = 1)P(X_2 = 3 \mid X_1 = 1)P(X_3 = 2 \mid X_2 = 3)$ can be calculated just once and multiplied with the three entries in $P(X_4 = x_4 \mid X_3 = 2)$ resulting in fewer computations.

Show your work:

$P = P(X1=1) \times P(X2=3 \mid X1=1) \times P(X3=2 \mid X2=3)$
$= 0.1 \times 0.2 \times 0.5$

Write the final answer:

0.01

4. **[3 pts]** Compute $P(X_1 = 1, X_3 = 2)$. **Hint:** Note that $P(X_1 = 1, X_3 = 2) = \sum_{x_2} P(X_1 = 1, X_2 = x_2, X_3 = 2)$. Thus, for each of the three value of X_2, you can calculate $P(X_1 = 1, X_2 = x_2, X_3 = 2)$ as in the previous question and add the three resultant values together. Again, note that this is much lesser computation than calculating $P(X_1 = 1, X_2 = x_2, X_3 = 2, X_4 = x_4)$ for each of the 9 combinations of x_2, x_4 values and then adding these joint probabilities together. We are essentially decomposing the problem and reusing our work instead of dealing with the original problem of calculating $P(X_1 = x_1, X_2 = x_2, X_3 = x_3, X_4 = x_4)$ for each combination of x_1, x_2, x_3, x_4 values we care about.

   Show your work:

$$P = P(X1=1) \times \sum_{i=1,2,3} (P(X2=i \mid X1=1) \times P(X3=2 \mid X2=i))$$
$$= 0.1 \times (0.3 \times 0.5 + 0.5 \times 0.33 + 0.2 \times 0.5)$$
$$= 0.0416$$

   Write the final answer:

   0.0416

5. **[3 pts]** Compute $P(X_3 = 2)$. **Hint:** Calculate $P(X_1 = x_i, X_3 = 2)$ for $x_i = 1, 2, 3$ as in the preceding question, and add the three values together to obtain the answer.

   Show your work:

$$P(X3 = 2)$$

$$= \sum_{\substack{i=1,2,3 \\ j=1,2,3}} P(X1=i) \times P(X2=j \mid X1=i) \times P(X3=2 \mid X2=j)$$

$$= 0.42367$$

   Write the final answer:

0.42367

4 Dynamic Programming with Markov Chains (Programming) [3 pts]

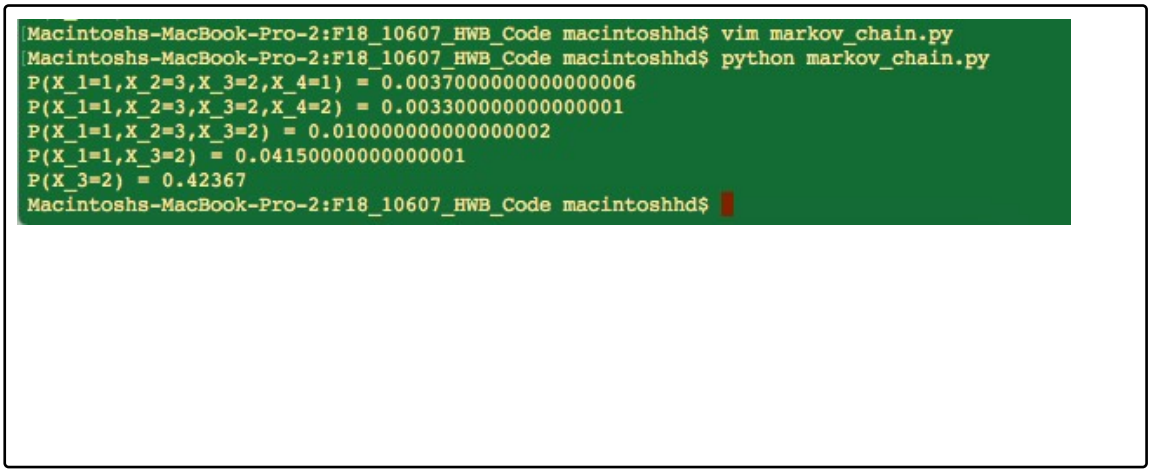
In the accompanying file `markov_chain.py`, fill out code to complete the provided function `compute_probability`.

1. **[1 pts]** Do the outputs from the implementation match your manually calculated answers to the questions in your previous section? **Select one:**

   ● Yes

   ○ No

2. **[2 pts]** Include a legible screenshot of your console output here.

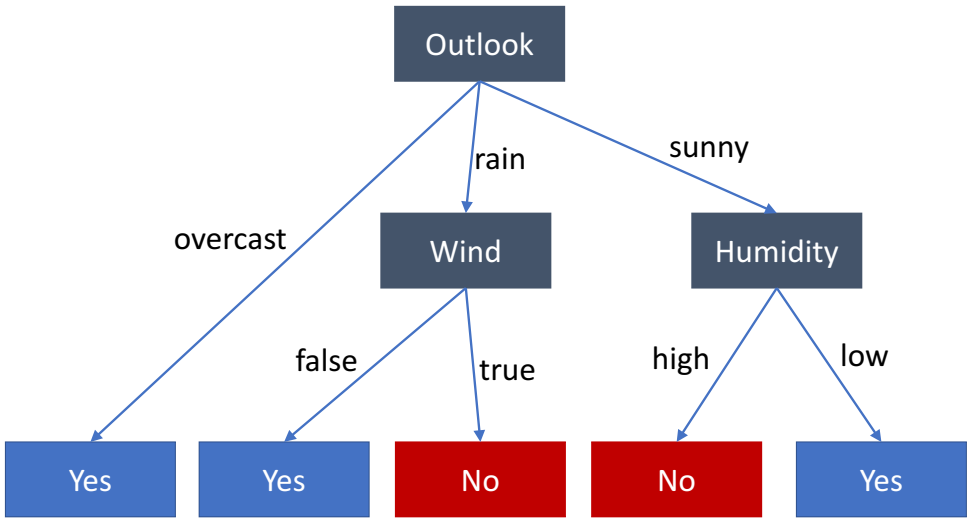


5 Data Structures and Recursion [0 pts]

THIS QUESTION IS NO LONGER A PART OF HW-B. NEITHER THE QUESTION BELOW NOR THE CODE SUBMISSION WILL BE GRADED IN HW-B. IT WILL BE INCORPORATED IN HW-C. PLEASE INCLUDE CODE AND ANSWERS TO THIS QUESTION IN HW-C.

Implement a decision tree data structure in Python. The easiest way to do this is to use a Python dictionary where the condition at a node can be used as a string key to traverse to the children of the node.

Using your tree implementation, create the following decision tree in Python code:



Use depth-first search of the tree to read out each possible rule from the root of the tree to each leaf in the tree. The output of your code should resemble the following:

```
If Outlook=overcast, Golf=yes
If Outlook=rain and Wind=false, Golf=yes
If Outlook=rain and Wind=true, Golf=no
If Outlook=sunny and Humidity=high, Golf=no
If Outlook=sunny and Humidity=low, Golf=yes
```

All your code for this question should be in a Python file named `tree.py`. Running the file using the command `python3 tree.py` should generate the console output provided above.

1. **[0 pts]** Include a legible screenshot of your console output here.

6 Submission [1 pts]

1. **[1 pts]** In addiiton to the PDF submission with your written answers and plots, please zip the files `perceptron.py` and `markov_chain.py` directly (do NOT place them in a folder and then zip), name the zipped file `<andrewid>-10607-hwb.zip`, and submit the zipped folder on Gradescope under the assignment `Homework B (Programming)`. Your code file submissions should follow the provided code template files. In particular, they should NOT include any additional Python imports such as `matplotlib` which might make it fail on the Autograder. Please maintain any code for creating your plots in separate files. You do not have to submit your plotting code on Gradescope. Have you made the code submission on Gradescope? **Select one:**

   ◉ Yes

   ○ No

7 Collaboration Policy

After you have completed all other components of this assignment, report your answers to the collaboration policy questions detailed in the Academic Integrity Policies found here.

1. Did you receive any help whatsoever from anyone in solving this assignment? If so, include full details including names of people who helped you and the exact nature of help you received.

2. Did you give any help whatsoever to anyone in solving this assignment? If so, include full details including names of people you helped and the exact nature of help you offered.

3. Did you find or come across code that implements any part of this assignment? If so, include full details including the source of the code and how you used it in the assignment.