Homework 10

Due Friday, April 26 at 5:00 PM

"Fish are friends, not food." - *Finding Nemo (2003)*

Handing In

To hand in a homework, go to the directory where your work is saved and run `cs0160_handin hwX` where X is the number of the homework. Make sure that your written work is saved as a .pdf file, and any Python problems are completed in the same directory or a subdirectory. You can re-handin any work by running the handin script again. We'll only grade your most recent submission. To install stencil Python files for a homework, run `cs0160_install hwX`. **You will lose points if you do not hand in your written work as a .pdf file.**

1 Written Problems

Racing Horses

Ignorant Bet Person (IBP) is watching 2 horses (A and B) race against each other with a group of experts - Darth Vader (DV), Linkin Park (LP), Voldemort (V), and the Brittany Spears (BS). True to the name, IBP wants to bet on which horse will win each race, but sadly knows nothing about the horses that are racing. IBP instead decides to bet based on the advice that the experts provide.

Ideally, in an offline situation, IBP would only listen to the expert who knows the most about the horses. However, and once more true to the name, IBP doesn't know who that is! Instead, to deal with this online situation, IBP decides to bet using a weighted majority vote based on the advice given by the experts. IBP initially considers each of the other members equally with respective weights of 1. When an expert gives bad advice, IBP weighs all of their subsequent advice exactly one third of the previous weight. If there is a tie, IBP will always bet on horse A winning the race. Given the table below of advice and actual winner, fill in IBP's bet for each race. The first bet has been filled out for you.

How much weight do each of the experts have after the end of the fourth race?

Race	DV	LP	V	BS	IBP	Actual
1	B	A	A	B	A	A
2	A	B	A	A		A
3	B	B	B	A		A
4	A	A	B	B		B

Computability

In class, we talked about classifications of computability (P, NP, NP-Hard, NP-Complete).

Part 1:

For the following problems, state which complexity class each of the problem falls under, and explain your answer in one to two sentences. For problem 3, in three or four sentences, describe the algorithm to solve this problem.

1. Sorting a list of numbers
2. Creating a seating arrangement of n people at k tables, such that no two people who hate each other are seated at the same table. You can assume that you know how two arbitrary people k_A and k_B feel about each other.
3. Determining whether it's possible to assign each of a graph's vertices one of two colors, such that no two vertices that share an edge have the same color.

Part 2:

In class, we talked about P and NP as if they were known to be different complexity classes. In reality, it's not known whether P = NP, although in most cases we assume P != NP.

Read through this article http://news.mit.edu/2009/explainer-pnp

Consider that RSA encryption (one of the most common encryption techniques) is in NP. In one to two sentences, explain what it would mean with regards to this if it was proven that P = NP.

Machine Learning

1. Consider the following two applications for a neural network and state how large the final layer should be:
   (a) Given an input vector of numbers, we want to return a True or False classification.
   (b) Given an input black and white image of a handwritten digit, we can convert this image to a 1D vector of 1s and 0s corresponding to whether each pixel is black or white. We want to find the digit (0-9) represented in the image given this input 1D vector.
2. Given a list of input values, describe how a perceptron would calculate an output value.

Algorithmic Fairness

We learned about machine learning and bias in algorithms during class.

If someone writes a machine learning algorithm that is biased in its results (it's incredibly difficult to avoid *all* bias...), who do you think is responsible? And why? Some examples of individuals involved may be the folks who provided the data that the algorithm was trained on, the programmer who wrote the algorithm, the owner of the computer on which the computations were performed, the company or organization that the programmer works for... but there are many more!

What are two real world instances where algorithmic bias has been/become an issue?

There's no correct answer to this problem - we're just interested in your thoughts!

2 Python Problems

Functional Programming Practice

In class we talked about two higher order functions called `map` and `reduce`. Solve each of the following problems using only python's built in `map` or `reduce` (it is up to you to decide which one is appropriate for the problem). The functions you pass into `map` or `reduce` must be anonymous functions. Your solutions should go in the stencil `functional.py`. You might have to think about each of these for a while, but the solution to each one is a single line of code (besides the error handling).

Part 1: apply_all

Fill in the function `apply_all` in the stencil. This function takes in a list of unary (one argument) functions and a number. It returns a new list with each of those functions applied to that number. In other words, if you pass in the list of functions `[f(x), g(x), h(x)]` and the number `n`, `apply_all` should produce `[f(n), g(n), h(n)]`.

Example

- `apply_all([lambda x: x+1, lambda x: x+2, lambda x: x+3], 4)` → `[5, 6, 7]`

Part 2: compose

Fill in the function `compose` in the stencil. This function takes in a list of unary functions and a number `n`. `Compose` should compose all of the functions in the list and apply to `n` to produce a single number. The inner-most function in the composition will be the first function in the input list, and the outer-most function in the composition will be the last function in the input list. In other words, if you pass in the list of fuctions `[f(x), g(x), h(x)]` and the number `n`, `compose` should produce `h(g(f(n)))`.

Example

- `compose([lambda x: x+1, lambda x: x+2, lambda x: x+3], 4)` → `10`.

Part 3: list_compose_steps

Fill in `list_compose_steps`, which should take in a list of unary functions and a number n. It should return a list with each of the intermediate values produced by `compose`. In other words, if you pass in the list of functions `[f(x), g(x), h(x)]` and the number `n`, `list_compose_steps` should produce `[n, f(n), g(f(n)), h(g(f(n)))]`. Please note that using `append` will not work in your lambda expression because `append` doesn't return anything. To append some number x to a list, you should write `list+[x]` instead of `list.append(x)`.

Example

- `list_compose_steps([lambda x: x+1, lambda x: x+2, lambda x: x+3], 4)` → `[4, 5, 7, 10]`.

Keep in mind that you should never call `compose` and that you can only use `map` or `reduce`. Also remember that the number n should be the first element in your output list.

All of these should raise an `InvalidInputException` if any of the inputs are `None`.

Testing

Write your test cases in `functional_test.py`. A few examples have already been filled in for you.