CS 3430: S22: Scientific Computing
Midterm 03

Vladimir Kulyukin
Department of Computer Science
Utah State University

April 28, 2022

Instructions

1. This exam has 11 problems worth 25 points. Your solutions are due in Canvas by 11:59pm on April 29, 2022.

2. You're welcome to use your solutions to your previous homework problems in this exam, your class notes, the Canvas lecture materials, your Python IDE's documentation; **you may not use any other materials while working on your solutions**.

3. You'll type and code your solutions in the included file `cs3430_s22_exam03.py` and submit it in Canvas. **Remember to write your name and A-number at the top of the file**, which will make it easier for us to grade your work.

4. You may not talk to anyone during this exam orally, digitally, in writing, by phone, by sign language, or any other form of communication (including telepathy).

5. **If you use your homework solutions, you must include all the homework files into your submission**. For example, if, in `cs3430_s22_exam03.py`, you use your code from `cs3430_s22_hw_x`, put the following import statement in `cs3430_s22_exam03.py`

   ```
   from cs3430_s22_hw_x import y
   ```

   and include `cs3430_s22_hw_x.py` into your submission. If you fail to include your former submission and your code doesn't pass a unit test due to a failed import, I'll give only partial credit at best. There won't be any back and forth on resubmissions for the final, because I must submit the grades 72 hours after the final exam. Take this note seriously.

6. Problems 1, 7, 8, and 11 do not require any coding. You'll type your answers in multi-line comments in `cs3430_s22_exam03.py`. The other problems require coding solutions. The file `cs3430_s22_exam03_uts.py` has 44 unit tests I've written to help you verify your solutions to the coding problems (Problems 2, 3, 4, 5, 6, 9, and 10).

7. You solutions in `cs3430_s22_exam03.py` are due in Canvas **by 11:59pm on April 29, 2022**. Again, if you have multiple files to submit, because of your imports, zip them all up in `cs3430_s22_exam_03.zip` and submit the zip.

8. I thank you all once again for taking CS 3430: S22 and wish you all best of luck on this exam and, as always, Happy Hacking!

Problem 1 (3 points)

Find an equivalence class whose elements give the remainders 7, 13, 17, 59 when divided by 29, 31, 37, 53, respectively. State your solution in the form $[a]_m$ (e.g., $[60]_{131}$). Recall that the notation $[a]_m$ is the equivalence class of a modulo m.

Problem 2 (2 points)

Write the function `solve_cong_system_with_crt(m_ary, a_ary)` that implements the Chinese Remainder Theorem (CRT) algorithm. The parameter `m_ary` is an array of pairwise relatively prime positive integers (i.e., $[m_1, m_2, ..., m_k]$), the parameter `a_ary` is an array of integers (i.e., $[a_1, a_2, ..., a_k]$). Remember that `m_ary` and `a_ary` must be of the same length. This function returns an equivalence class implemented as a Python generator that solves the congruence system $x \equiv a_i \pmod{m_i}$, $1 \leq i \leq k$. Save your code in `cs3430_s22_exam03.py`.

Problem 3 (2 points)

Write the function `solve_cong_with_xeuc(a, b, m)` that uses the Extended Euclid algorithm to find a solution to the congruence $ax \equiv b \pmod{m}$, if it exists, and returns an equivalence class implemented as a Python generator that solves the congruence. If there is no solution, the function returns `None`. Save your code in `cs3430_s22_exam03.py`.

Problem 4 (2 points)

Implement the function `rand_lcg(a, b, m, n, x0=0)` that generates random numbers with the LCG method. This method takes the parameters `a`, `b`, `m`, where `a` is the multiplier, `b` is the increment, `b` is the modulus, and $x0$ is the seed. The parameter `n` specifies how many numbers are generated. This method returns a Python generator that generates `n` random numbers. Save your code in `cs3430_s22_exam03.py`.

Problem 5 (2 points)

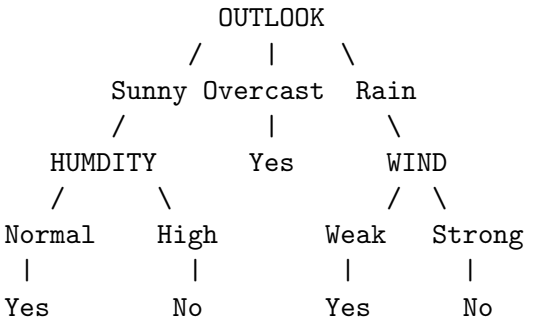
Implement the function `rand_xorshift(a, b, c, n, x0=1)` that generates random numbers with the 32 xorshit method. This method takes the parameters where `a`, `b`, `c` define the xorshift triple. The parameter `n` specifies how many numbers are generated and the keyword `x0` defines the seed. This method returns a Python generator to generate `n` random numbers. Save your code in `cs3430_s22_exam03.py`.

Problem 6 (3 points)

Implement `equidistrib_test(seq, n, lower_bound, upper_bound)` that executes the Equidistribution Test. This method returns two values: the `V` statistic and its p value. The argument seq is the sequence of `n` random numbers in the range `[lower_bound, upper_bound]`, where `lower_bound=0` and `upper_bound` is a positive integer. Save your code in `cs3430_s22_exam03.py`.

Problem 7 (1 point)

Consider the following decision tree.



Use this decision tree to classify the following examples with respect to the binary target `PlayTennis`. Write your answers in a multi-line comment for this problem in `cs3430_s22_exam03.py`. You don't have to write any code for this problem.

1. { `Wind=Weak`, `Humidity=High`, `Outlook=Sunny`, `Temperature=Hot` };
2. { `Humidity=High`, `Temperature=Mild`, `Outlook=Rain`, `Wind=Weak` };
3. { `Temperature=Cool`, `Humidity=Normal`, `Wind=Strong`, `Outlook=Rain` };
4. { `Temperature=Mild`, `Humidity=High`, `Wind=Weak`, `Outlook=Sunny` };
5. { `Wind=Strong`, `Humidity=High`, `Temperature=Mild`, `Outlook=Rain` }.

Problem 8 (2 points)

a) Compute the entropy of the following dataset with respect to the target `PLAY_TENNIS`.

DAY	OUTLOOK	TEMPERATURE	HUMIDITY	WIND	PLAY_TENNIS
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes

b) Compute the entropy of the following dataset with respect to the target `PLAY_TENNIS`.

DAY	OUTLOOK	TEMPERATURE	HUMIDITY	WIND	PLAY_TENNIS
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

c) Compute the gain of `WIND` of the following dataset given that the target is `PLAY_TENNIS`.

DAY	OUTLOOK	TEMPERATURE	HUMIDITY	WIND	PLAY_TENNIS
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes

d) Compute the gain of `HUMIDITY` of the following dataset given that the target is `PLAY_TENNIS`.

DAY	OUTLOOK	TEMPERATURE	HUMIDITY	WIND	PLAY_TENNIS
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes

Write your answers in a multi-line comment for this problem in `cs3430_s22_exam03.py`. You don't have to write any code for this problem.

Problem 9 (3 points)

Use the files `train_data.csv` and `test_data.csv` for this problem. The file `train_data.csv` contains 123 training examples generated from the IRIS dataset. This is one of the standard machine learning datasets that consists of examples of IRIS flowers. The file `test_data.csv` contains 27 examples generated from this dataset. The target attribute for this dataset is called `Class`. The attributes `SepLen` and `SepWid` denote the septal length and width, respectfully, for each flower example. The attributes `PetLen` and `PetWid` denote the petal length and width, respectfully, for each flower example.

a) Implement the function `learn_bin_id3_dt_from_csv_file(csv_fp, target_attrib)` that takes a path to a csv file of training examples, applies the binary ID3 algorithm to them with the target attribute specified by the second argument `target_attrib`, and returns the root of the learned binary ID3 decision tree (i.e., the object of the `id3_node` class in `bin_id3.py` you used in your homework).

b) Implement the function `classify_csv_file_with_bin_id3_dt(dt_root, csv_fp, target_attrib)` that takes the root of the binary ID3 tree returned by `learn_bin_id3_dt_from_csv_file()` (`dt_root`), a path to a csv file with test examples (`csv_fp`) and the target attribute `target_attrib`, and returns the average classification accuracy of `dt_root` on the test examples in `csv_fp`.

Here's a test run of `learn_bin_id3_dt_from_csv_file()` and my output. Your tree may look different but shouldn't be deeper.

```
>>> dt_root = learn_bin_id3_dt_from_csv_file('train_data.csv', 'Class')
>>> assert not dt_root is None
>>> display_bin_id3_node(dt_root)

PetLen
        PetLen1
                No
        PetLen2
                No
        PetLen0
                Yes
        PetLen4
                No
        PetLen3
                No
```

Your classification accuracy doesn't have to be 100%, but it should definitely be $\geq 90\%$, because it's a pretty simple dataset. Save your code in `cs3430_s22_exam03.py`.

Problem 10 (2 points)

a) Write the function `build_huffman_tree_from_text(txtstr)` that takes a string `txtstr` and builds a Huffman tree from the frequencies of each character in the string `txtstr`. Save your code in `cs3430_s22_exam03.py`.

b) Write the function `encode_moby_dick_ch03()` that produces the Huffman tree binary encoding of Chapter 3 from "Moby Dick" given in `moby_dick_ch03.txt` and saves the encoding in the current directory. This function should produce two files – `moby_dick_ch03.bin` and `moby_dick_ch03_pb.txt`. In your comments next to `encode_moby_dick_ch03()`, subtract the sum of the sizes of `moby_dick_ch03.bin` and `moby_dick_ch03_pb.txt` from the size of `moby_dick_ch03.txt` and state how many bytes your encoding saved. Save your code in `cs3430_s22_exam03.py`.

Problem 11 (3 points)

Compute the correlation coefficients matrix between the following two matrices.

$A =$

10	0	0
0	1	0
0	0	0

$B =$

0	0	10
0	2	0
0	0	0

Type your correlation matrix in a multi-line comment in `cs3430_s22_exam03.py`.

What to Submit

Submit `cs3430_s22_exam03.py` and all imports required to run your functions in `cs3430_s22_exam03.py`. Zip all your files in `cs3430_s22_exam_03.zip` and submit it in Canvas.