CSC 4780/6780
Fall 2022
Homework 05

September 14, 2022

This homework is due at 11:59 pm on Sunday, Sept 18. It must be uploaded to iCollege by then. No credit will be given for late submissions. A solution will be released by noon on Monday, Sept 19.

Once again: it is a good idea to get this done and turn it in early.

1 What are we doing?

Using Bayes' Rule to make inferences based on evidence is crucial skill for data scientists. Bayes' Rule lets us find the best explanation for what we see, and lets us quantify our confidence in that explanation.

And you get to write some code!

2 Do inference

Your friend has two pairs of dice. She rolls the first pair and notes their sum (But does not tell you what that sum is!) She repeatedly rolls the second pair and tells you if the new sum is higher than (H), lower than(L), or equal to (E) the first pair. You are trying to guess what the sum of the first pair of dice is.

Based on the squence of H, L, and E, you need to figure out the probability of each possible sum.

For example, if she tells you HHL, you would compute the following probabilities (to six places after the decimal point):

Sum	Probability
2	0
3	0.025953
4	0.096518
5	0.193322
6	0.262744
7	0.241295
8	0.125116
9	0.044613
10	0.009652
11	0.000786
12	0

So your best guess for this round is 6, but there is only about a 26% chance that you are right.

The file `input.txt` has a round of the game on each line. (For each round, she re-rolls the first pair of dice.)

```
H
L
E
HL
HHL
EEEEEE
HHHHHHHH
LLLLLLLL
HHHHHHHHHHHH
LLLLLLLLLLLL
HHHHHHHHHHHHE
LLLLLLLLLLLLE
```

Your program should read `input.txt` and generate a CSV that lists the input (truncated to 7 characters, if necessary), the probability for each possible original roll, and your best guess at what the original sum was:

```
input,P(d=2),P(d=3),P(d=4),P(d=5),P(d=6),P(d=7),P(d=8),P(d=9),P(d=10),P(d=11),P(d=12),guess
H,0.06087,0.11478,0.15652,0.18087,0.18261,0.15652,0.08696,0.04174,0.01565,0.00348,0.00000,6
L,0.00000,0.00348,0.01565,0.04174,0.08696,0.15652,0.18261,0.18087,0.15652,0.11478,0.06087,8
E,0.00685,0.02740,0.06164,0.10959,0.17123,0.24658,0.17123,0.10959,0.06164,0.02740,0.00685,7
HL,0.00000,0.01229,0.05028,0.11620,0.19553,0.25140,0.19553,0.11620,0.05028,0.01229,0.00000,7
HHL,0.00000,0.02595,0.09652,0.19332,0.26274,0.24130,0.12512,0.04461,0.00965,0.00079,0.00000,6
EEEEEE,0.00000,0.00027,0.00462,0.03460,0.16496,0.59110,0.16496,0.03460,0.00462,0.00027,0.00000,7
HHHH...,0.27893,0.34841,0.24380,0.10347,0.02342,0.00190,0.00006,0.00000,0.00000,0.00000,0.00000,3
LLLL...,0.00000,0.00000,0.00000,0.00000,0.00006,0.00190,0.02342,0.10347,0.24380,0.34841,0.27893,11
HHHH...,0.38714,0.38217,0.18266,0.04373,0.00421,0.00009,0.00000,0.00000,0.00000,0.00000,0.00000,2
LLLL...,0.00000,0.00000,0.00000,0.00000,0.00000,0.00009,0.00421,0.04373,0.18266,0.38217,0.38714,12
HHHH...,0.20419,0.40313,0.28902,0.09226,0.01111,0.00028,0.00000,0.00000,0.00000,0.00000,0.00000,3
LLLL...,0.00000,0.00000,0.00000,0.00000,0.00000,0.00028,0.01111,0.09226,0.28902,0.40313,0.20419,11
```

You should not assume a number of dice or a number of sides of the dice, those will be supplied on the command-line.

You will need to use Bayes' Law, which says "The probability that the original sum was d given a sequence s is given by

$$P(d|s) = \frac{P(s|d)P(d)}{P(s)}$$

To calculate $P(s)$, you will use:

$$P(s) = \sum_{d \in D} P(s|d)P(d)$$

3 Criteria for success

If your name is Fred Jones, you will turn in a zip file called `HW04_Jones_Fred.zip` of a directory called `HW05_Jones_Fred`. It will contain:

- `dice.py`
- `input.txt`
- `output26.csv`
- `output57.csv`

Be sure to format your python code with black before you submit it.

We will run your code like this:

```
cd HW04_Jones_Fred
python3 dice.py 2 6 input.txt output26.csv
python3 dice.py 5 7 input.txt output57.csv
```

Your program will read the `input.txt`. The game was played with 2 6-sided dice. Output your inferences. On the second run, same thing, but the game was played with 5 7-sided dice.

Do this work by yourself. Stackoverflow is OK. A hint from another student is OK. Looking at another student's code is *not* OK.

4 6780 Only: Infer on long sequences using logs

If your `dice.py` is used for very long sequences of rolls, it will underflow the floating-point numbers on your computer. Try it:

```
> python3 dice.py 2 6 long_input.txt long_output26.csv
```

This probably results in:

```
input,P(d=2),P(d=3),P(d=4),P(d=5),P(d=6),P(d=7),P(d=8),P(d=9),P(d=10),P(d=11),P(d=12),guess
HHEH...,nan,nan,nan,nan,nan,nan,nan,nan,nan,nan,nan,0
LLLL...,nan,nan,nan,nan,nan,nan,nan,nan,nan,nan,nan,0
LLLL...,nan,nan,nan,nan,nan,nan,nan,nan,nan,nan,nan,0
```

You are going to use logarithms to enable your code to deal with very long sequences like these.

Copy `dice.py` to `long_dice.py`

Your lookup table will contain the log of the likelihoods. It will add those instead of multiplying them. At the end, just before dividing by the marginal likelihood, you will exponentiate them (e^x) so that you get exactly the same results as the previous exercise. Before exponentiating them, you can add a large positive constant to all the log likelihooods.

Here is the code:

```
unnormed_posteriors = np.exp(unnormed_log_posteriors np.max(unnormalized_log_posteriors))
normalized_posteriors = unnormed_posteriors / np.sum(unnormed_posteriors)
```

You can test your code on `long_input.txt`:

```
> python3 long_dice.py 2 6 long_input.txt  long_output26.csv
```

It should look like this:

```
input,P(d=2),P(d=3),P(d=4),P(d=5),P(d=6),P(d=7),P(d=8),P(d=9),P(d=10),P(d=11),P(d=12),guess
HHEH...,0.00000,0.00000,0.00000,0.00000,1.00000,0.00000,0.00000,0.00000,0.00000,0.00000,0.00000,6
LLLL...,0.00000,0.00000,0.00000,0.00000,0.00000,0.00000,0.00000,0.00000,1.00000,0.00000,0.00000,10
```

(Yes, with long sequences you get pretty confident.)

Your zip file should include:

- `long_dice.py`
- `long_input.txt`
- `long_output26.csv`