Coin EM Implementation Assignment
deadline: 4pm Mon Oct 15

October 5, 2018

please read all of the following carefully (don't be put off by the fact that it runs to 2.5 pages; you have to write only a little code)

the code you need to modify is in */shared/teaching/CSLL/4CSLL5_18_19/CoinEMAssignment*. If logged into one of the lab machines in LG12 you should be able to find this. You need to make a copy of all its contents into some directory of your own. You should be able to get the code to compile with

```
make make_gamma
```

and then you should be able to execute

```
./make_gamma coin_data_notes_order
```

and you should see:

```
processing coin_data_notes_order
read all data
total amount of extracted data is: 9
CHOICE: A
TOSSES: HHHHHHHHTT H:8 T:2
CHOICE: B
TOSSES: TTHTTTHTTT H:2 T:8
CHOICE: A
TOSSES: HTHHTHHHHT H:7 T:3
CHOICE: A
TOSSES: HTHHHTHHHH H:8 T:2
CHOICE: B
TOSSES: TTTTTTHTTT H:1 T:9
CHOICE: A
TOSSES: HHTHHHHHHH H:9 T:1
CHOICE: A
TOSSES: THHTHHHHHT H:7 T:3
CHOICE: A
TOSSES: HHHHHHTHHH H:9 T:1
CHOICE: B
TOSSES: HHTTTTTHTT H:3 T:7
1: A(0)   B(0)
2: A(0)   B(0)
3: A(0)   B(0)
4: A(0)   B(0)
5: A(0)   B(0)
```

```
6: A(0)   B(0)
7: A(0)   B(0)
8: A(0)   B(0)
9: A(0)   B(0)
```

where the first part of the output is just a display of the data after the program has read it in, and the second part is a display of all the values in a 2-dimension data structure `gamma`, corresponding to the $\gamma_d(Z)$ from the slides. Currently these displayed values are not correct.

what you must do The aim is to make the `gamma` table contain correct values. In *make_gamma.cpp* between the positions where is says `BEGIN INSERT` and `END INSERT` you have to insert some code.

Once the code has been modified it should display, for each d, and for each possible value (A and B) of the coin choice variable Z, the **conditional probability** $P(Z|\boldsymbol{X}^d)$, where $\boldsymbol{X}^d$ is the sequence of H and T outcomes for the d^{th} data item.

In order to easily make data structures to store probabilities, the code is designed around the idea of mapping the possible values of variables to integers, counting from 0. So the A and B alternatives for the coin choice are represented with 0 and 1. The H and T alternatives for a coin toss are represented with 0 and 1.

So `gamma[d][0]` will represent $\gamma_d(Z = A)$, and `gamma[d][1]` will represent $\gamma_d(Z = B)$.

Clearly it is only possible to work out these probabilities under some setting of the *parameters*. The code stores the coin choice parameters in an size 2 array `chce_probs` and the correspondence to the treatment in the slides is:

slides	meaning	code
θ_a	$P(Z = a)$	`chce_probs[0]`
θ_b	$P(Z = b)$	`chce_probs[1]`

The heads/tails probs for two possible coins to choose are stored in a 2x2 array `ht_probs`, where the first dimension is for which coin is chosen (ie. A vs B), second dimension is the coin toss outcome (ie. H vs T), so the correspondence to the treatment in the slides is:

slides	meaning	code
$\theta_{h\|a}$	$P(h\|a)$ ie. the head prob of coin A	`ht_probs[0][0]`
$\theta_{t\|a}$	$P(t\|a)$ ie. the tail prob of coin A	`ht_probs[0][1]`
$\theta_{h\|b}$	$P(h\|b)$ ie. the head prob of coin B	`ht_probs[1][0]`
$\theta_{t\|b}$	$P(t\|b)$ ie. the tail prob of coin B	`ht_probs[1][1]`

In the *make_gamma.cpp* the values for these probabilities are set as follows:

```
chce_probs[0] = 0.2; chce_probs[1] = 0.8;
ht_probs[0][0] = 0.4;   ht_probs[0][1] = 0.6;
ht_probs[1][0] = 0.3;   ht_probs[1][1] = 0.7;
```

note: these values are different to the values assumed in the slides.

desired output Once the code is completed, the display of the γ values under these parameters, for this data set should be:

```
1: A(0.647224)   B(0.352776)
2: A(0.114646)   B(0.885354)
3: A(0.541164)   B(0.458836)
4: A(0.647224)   B(0.352776)
5: A(0.0768477)   B(0.923152)
6: A(0.740524)   B(0.259476)
```

```
7: A(0.541164)   B(0.458836)
8: A(0.740524)   B(0.259476)
9: A(0.16766)   B(0.83234)
```

further aspects of the code The source code consists of

make_gamma.cpp
prob_tables_coin(.cpp/.h)
CoinTrial(.cpp/.h)
Variable(.cpp/.h)
SymTable(.cpp/.h)

there is documentation in the code, and there is web-browsable documentation in *CoinEM-Docs*. You should not need to change any code except the code in *make_gamma.cpp*, with the other code taking care of reading the data and representing it.

Basically each line from the input file *coin_data_notes_order*, eg. `A H H H H H H H H T T` is represented by a *CoinTrial* object whose `outcomes` member represents the sequence of H and T outcomes.

A function `process_corpus` takes care of reading all these lines and stores the results in

`vector<CoinTrial> data`

You should only need to write code referring to `data`, `gamma` and the probability tables `chce_probs` and `ht_probs`