Problem Wk.10.1.7: Implementing Operations on Conditional Distributions

---

Part 1: Implement bayesEvidence

Implement the function `bayesEvidence(PBgA, PA, b)`, where `PBgA` specifies *P(B | A)*, `PA` is *P(A)* and `b` is a value for *B*. It should return a `DDist` corresponding to *P(A | B = b)*. This realizes the idea of Bayesian reasoning.

Continuing the example from the previous problem:

```
>>> dist.bayesEvidence(PTgD, disease, 'posTest')
DDist(noDisease: 0.833333, disease: 0.166667)
>>> dist.bayesEvidence(PTgD, disease, 'negTest')
DDist(noDisease: 0.978261, disease: 0.021739)
```

This can be done just putting together the basic operations we have already defined (which are part of the `ddist` module). Your answer can be written in just one line.

`bayesEvidence` is in the `dist` module, so it has access to `JDist` directly. So, in your solution, you do not need to use `dist.JDist`; you can just `JDist`.

```
def bayesEvidence(PBgA, PA, b):
    pass
```

Part 2: Implement totalProbability

Implement the function `totalProbability(PBgA, PA)`, where `PBgA` is a conditional distribution (a function) specifying *P(B | A)*. It should return a `DDist` corresponding to *P(B)*. This implements the idea of Total probability.

Continuing the example from the previous problem:

```
>>> totalProbability(PTgD, disease)
DDist(posTest: 0.540000, negTest: 0.460000)
```

This can be done just putting together the basic operations we have already defined. This one should also need one line.

`totalProbability` is in the `dist` module, so it has access to `JDist` directly. So, in your solution, you do not need to use `dist.JDist`; you can just `JDist`.

```
def totalProbability(PBgA, PA):
   pass
```

6.01SC Introduction to Electrical Engineering and Computer Science

Spring 2011