SciComp With Py

Conditional Probability

Vladimir Kulyukin
Department of Computer Science
Utah State University

Outline

- Conditional Probability

Events and Probabilities

- A and B are events
- P(A) is the probability of event A occurring
- P(B) is the probability of event B occurring
- P(A, B) is the probability of A and B occurring
- P(B | A) is the probability of B occurring given that A has occurred

Conditional Probability

$$P(B \mid A) = P(A, B)/P(A)$$

Problem

A professor gives his students two tests. 80% of his students pass the 1st test. 70% of his students pass both tests. What percentage of students who pass the 1st test pass the 2nd test?

Solution

- A = student passes 1st test
- B = student passes 2nd test
- P(A, B) = 70%
- P(A) = 80%
- P(B | A) = P(A, B)/P(A) = 0.7/0.8 = 0.84

Problem

Generate 100,000 persons and randomly place them into the following age groups: 20's, 30's, 40's, 50's, 60's, and 70's. Generate purchasing probabilities in such a way that the younger a person is the less likely the person is to make a purchase. Compute conditional probabilities of a person making a purchase given that that person is in a specific age group.

Solution: Initializing Data Structures

```
from numpy import random
random.seed(0)
## number of people in each age group
peopleInAgeGroup = {20:0, 30:0, 40:0, 50:0, 60:0, 70:0}
## number of purchases in each age group
purchasesInAgeGroup = {20:0, 30:0, 40:0, 50:0, 60:0, 70:0}
## total number of purchases
numOfPurchases = 0
## total number of people
numOfPeople = 100000
```

source in cond_prob.py

Solution: Generating Data

```
for _ in xrange(numOfPeople):
    ## randomly choose an age group
    ageGroup = random.choice([20, 30, 40, 50, 60, 70])
    ## the younger you are the less likely you are to buy stuff
    purchaseProbability = float(ageGroup) / 100.0
    ## modify the number of people in ageGroup
    peopleInAgeGroup[ageGroup] += 1
    ## if the purchase probability > random
    if (random.random() < purchaseProbability):
        numOfPurchases += 1
        purchasesInAgeGroup[ageGroup] += 1
```

source in cond_prob.py

Solution: Implementing Standard Probabilities

```
## P(AgeGroup=x)
def probOfAgeGroup(x):
    return float(peopleInAgeGroup[x])/numOfPeople
# P(Purchase) = prob of buying something
def probOfPurchase():
    return float(numOfPurchases)/numOfPeople
## P(Purchase, AG=x)
def probOfPurchaseAndAgeGroup(x):
    return float(purchasesInAgeGroup[x])/numOfPeople
```

source in cond_prob.py

Solution: Implementing Conditional Probabilities

```
## P(Purchase | AgeGroup = x)
def probOfPurchaseGivenAgeGroup(x):
    return float(purchasesInAgeGroup[x])/peopleInAgeGroup[x]


## P(Purchase | AgeGroup = x) = P(Purchase, AgeGroup=x)/P(AgeGroup=x)
def condProbOfPurchaseGivenAgeGroup(x):
    return probOfPurchaseAndAgeGroup(x)/probOfAgeGroup(x)
```

source in cond_prob.py

Solution: Computing Probabilities

```
# display P(Purchase, AgeGroup=x)
for ag in xrange(20, 80, 10):
    print('P(Purchase, AG=%d) = %f' % (ag, probOfPurchaseAndAgeGroup(ag)))
# display P(AG=x)
for ag in xrange(20, 80, 10):
    print('P(AG=%d) = %f' % (ag, probOfAgeGroup(ag)))
# display two ways to compute P(Purchage | AgeGroup=x)
for ag in xrange(20, 80, 10):
    p = probOfPurchaseGivenAgeGroup(ag)
    cp = condProbOfPurchaseGivenAgeGroup(ag)
    print('p = %f; cp = %f' % (p, cp))
    assert(p == cp)
```

source in cond_prob.py

Sample Call: Computing Independent Probabilities

```
$ python cond_prob.py -nofp 100000 -pp 0.4 -pd 0.0001 -dp 0
Running independent experiment
Purchase and AgeGroup=20 are dependent
P(Purchase)=0.402180; P(Purchase|AgeGroup=20)=0.397965
Purchase and AgeGroup=30 are dependent
P(Purchase)=0.402180; P(Purchase|AgeGroup=30)=0.398851
Purchase and AgeGroup=40 are dependent
P(Purchase)=0.402180; P(Purchase|AgeGroup=40)=0.403520
Purchase and AgeGroup=50 are dependent
P(Purchase)=0.402180; P(Purchase|AgeGroup=50)=0.405516
Purchase and AgeGroup=60 are dependent
P(Purchase)=0.402180; P(Purchase|AgeGroup=60)=0.400829
Purchase and AgeGroup=70 are dependent
P(Purchase)=0.402180; P(Purchase|AgeGroup=70)=0.406332
```

source in cond_prob.py

Sample Call: Computing Dependent Probabilities

```
$ python cond_prob.py -nofp 100000 -pp 0.4 -pd 0.0001 -dp 1
Running dependent experiment
Purchase and AgeGroup=20 are dependent
P(Purchase)=0.451270; P(Purchase|AgeGroup=20)=0.202906
Purchase and AgeGroup=30 are dependent
P(Purchase)=0.451270; P(Purchase|AgeGroup=30)=0.301282
Purchase and AgeGroup=40 are dependent
P(Purchase)=0.451270; P(Purchase|AgeGroup=40)=0.396986
Purchase and AgeGroup=50 are dependent
P(Purchase)=0.451270; P(Purchase|AgeGroup=50)=0.500000
Purchase and AgeGroup=60 are dependent
P(Purchase)=0.451270; P(Purchase|AgeGroup=60)=0.603288
Purchase and AgeGroup=70 are dependent
P(Purchase)=0.451270; P(Purchase|AgeGroup=70)=0.700018
```

source in cond_prob.py