School of Data and Computer Science, Sun Yat-sen University

E09 Bayesian Network

17341203 Yixin Zhang

November 8, 2019

Contents

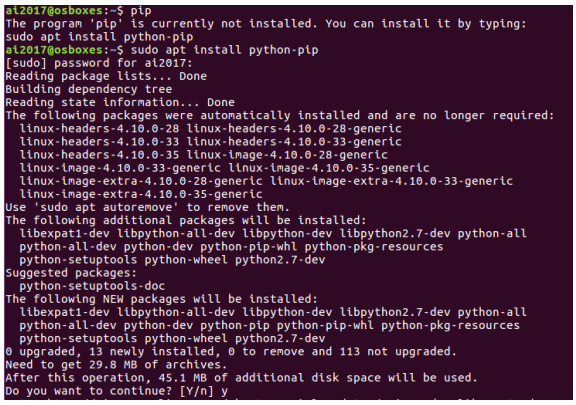
1 Pomegranate Installation

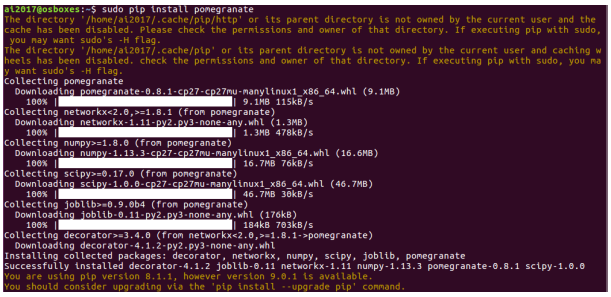
Under Linux:

1. Install `python` first (**python 2**, not python 3).

2. Run `sudo apt-get install python-pip` to install `pip`.

3. Run `sudo pip install pomegranate` to install `pomegranate`.

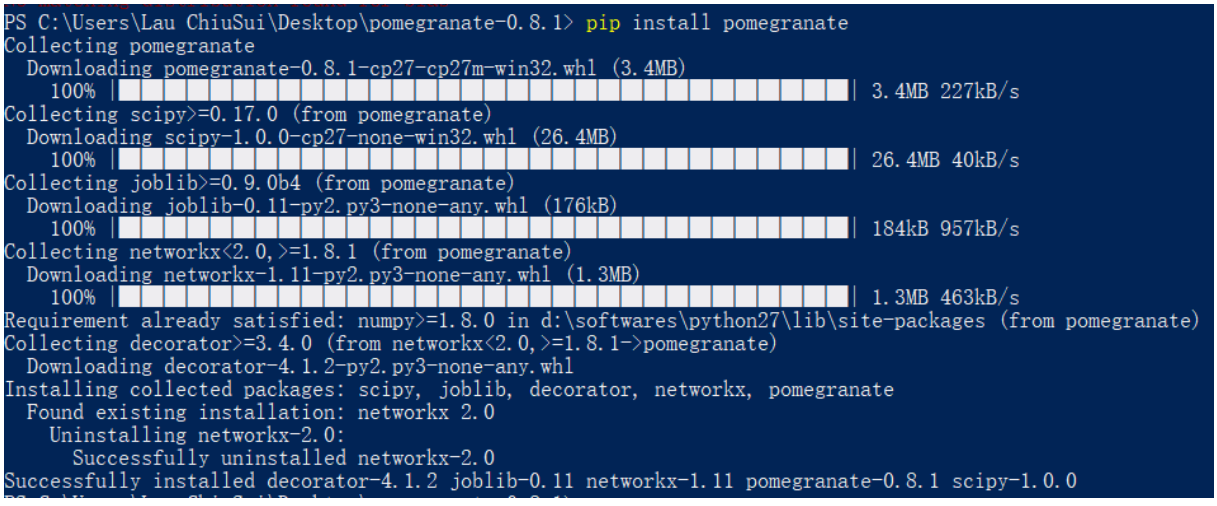




Under Windows

You can also run `pip install pomegranate` if you have installed `pip`. If you don't know how to install `pip`, please click `https://jingyan.baidu.com/article/e73e26c0d94e0524adb6a7ff.html`.

For more, please click the homepage of Pomegranate - `https://github.com/jmschrei/pomegranate` for help.



2 Building Bayesian Network

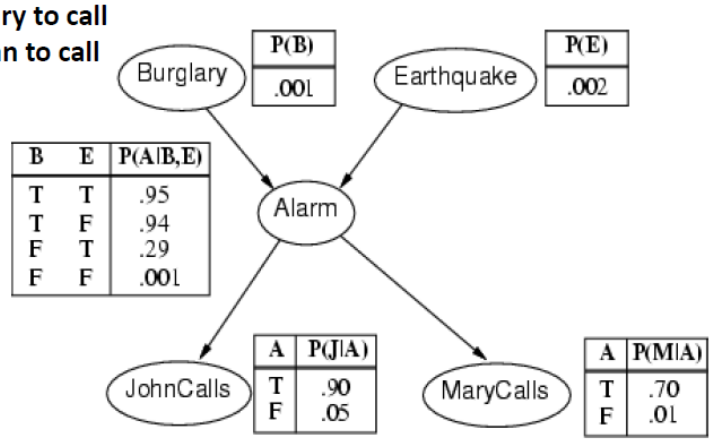
Please refer to `Tutorial_4_Bayesian_Networks.pdf`. I will explain it in class.

3 Tasks

3.1 Burglary

- **A burglary can set the alarm off**
- **An earthquake can set the alarm off**
- **The alarm can cause Mary to call**
- **The alarm can cause John to call**

Note that these tables only provide the probability that Xi is true.
(E.g., Pr(A is true|B,E))
The probability that Xi is false is 1- these values



Please code to calculate:

1. $P(A)$
2. $P(J\overline{M})$
3. $P(A|J\overline{M})$
4. $P(B|A)$
5. $P(B|J\overline{M})$
6. $P(J\overline{M}|\overline{B})$

```
P(Alarm) =
0.002516442

P(J&&~M) =
0.050054875461

P(A |J&&~M) =
0.0135738893313

P(B |A) =
0.373551228282

P(B |J&&~M) =
0.0051298581334

P(J&&~M |~B) =
0.049847949
```

3.2 Diagnosing

Variables and their domais

```
(1)PatientAge:['0-30','31-65','65+']
(2)CTScanResult:['Ischemic Stroke','Hemmorraghic Stroke']
(3)MRIScanResult: ['Ischemic Stroke','Hemmorraghic Stroke']
(4)StrokeType: ['Ischemic Stroke','Hemmorraghic Stroke', 'Stroke Mimic']
(5)Anticoagulants: ['Used','Not used']
(6)Mortality:['True', 'False']
(7)Disability: ['Negligible', 'Moderate', 'Severe']
```

CPTs

Note: [CTScanResult, MRIScanResult,StrokeType] means:

P(StrokeType='...' | CTScanResult='...' $\wedge$ MRIScanResult='...')

```
(1)
[PatientAge]

['0-30', 0.10],
['31-65', 0.30],
['65+', 0.60]

(2)
[CTScanResult]

['Ischemic Stroke',0.7],
[ 'Hemmorraghic Stroke',0.3]

(3)
[MRIScanResult]

['Ischemic Stroke',0.7],
[ 'Hemmorraghic Stroke',0.3]

(4)
[Anticoagulants]

[Used',0.5],
```

```
['Not used',0.5]

(5)
[CTScanResult, MRIScanResult,StrokeType])

['Ischemic Stroke','Ischemic Stroke','Ischemic Stroke',0.8],
['Ischemic Stroke','Hemmorraghic Stroke','Ischemic Stroke',0.5],
[ 'Hemmorraghic Stroke','Ischemic Stroke','Ischemic Stroke',0.5],
[ 'Hemmorraghic Stroke','Hemmorraghic Stroke','Ischemic Stroke',0],

['Ischemic Stroke','Ischemic Stroke','Hemmorraghic Stroke',0],
['Ischemic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke',0.4],
[ 'Hemmorraghic Stroke','Ischemic Stroke','Hemmorraghic Stroke',0.4],
[ 'Hemmorraghic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke',0.9],

['Ischemic Stroke','Ischemic Stroke','Stroke Mimic',0.2],
['Ischemic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1],
[ 'Hemmorraghic Stroke','Ischemic Stroke','Stroke Mimic',0.1],
[ 'Hemmorraghic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1],

(6)
[StrokeType, Anticoagulants, Mortality]

['Ischemic Stroke', 'Used', 'False',0.28],
['Hemmorraghic Stroke', 'Used', 'False',0.99],
['Stroke Mimic', 'Used', 'False',0.1],
['Ischemic Stroke','Not used', 'False',0.56],
['Hemmorraghic Stroke', 'Not used', 'False',0.58],
['Stroke Mimic', 'Not used', 'False',0.05],

['Ischemic Stroke', 'Used' ,'True',0.72],
['Hemmorraghic Stroke', 'Used', 'True',0.01],
['Stroke Mimic', 'Used', 'True',0.9],
['Ischemic Stroke', 'Not used' ,'True',0.44],
['Hemmorraghic Stroke', 'Not used', 'True',0.42 ],
['Stroke Mimic', 'Not used', 'True',0.95]
```

```
(7)
[StrokeType, PatientAge, Disability]

['Ischemic Stroke', '0-30','Negligible', 0.80],
['Hemmorraghic Stroke', '0-30','Negligible', 0.70],
['Stroke Mimic',      '0-30', 'Negligible',0.9],
['Ischemic Stroke',   '31-65','Negligible', 0.60],
['Hemmorraghic Stroke', '31-65','Negligible', 0.50],
['Stroke Mimic',      '31-65', 'Negligible',0.4],
['Ischemic Stroke',   '65+' , 'Negligible',0.30],
['Hemmorraghic Stroke', '65+' , 'Negligible',0.20],
['Stroke Mimic',      '65+' , 'Negligible',0.1],

['Ischemic Stroke',   '0-30' ,'Moderate',0.1],
['Hemmorraghic Stroke', '0-30' ,'Moderate',0.2],
['Stroke Mimic',      '0-30' ,'Moderate',0.05],
['Ischemic Stroke',   '31-65','Moderate',0.3],
['Hemmorraghic Stroke', '31-65','Moderate',0.4],
['Stroke Mimic',      '31-65','Moderate',0.3],
['Ischemic Stroke',   '65+' ,'Moderate',0.4],
['Hemmorraghic Stroke', '65+' ,'Moderate',0.2],
['Stroke Mimic',      '65+' ,'Moderate',0.1],

['Ischemic Stroke',   '0-30' ,'Severe',0.1],
['Hemmorraghic Stroke', '0-30' ,'Severe',0.1],
['Stroke Mimic',      '0-30' ,'Severe',0.05],
['Ischemic Stroke',   '31-65','Severe',0.1],
['Hemmorraghic Stroke', '31-65','Severe',0.1],
['Stroke Mimic',      '31-65','Severe',0.3],
['Ischemic Stroke',   '65+' ,'Severe',0.3],
['Hemmorraghic Stroke', '65+' ,'Severe',0.6],
['Stroke Mimic',      '65+' ,'Severe',0.8]
```

Calculation

Please code to calculate the following probability value:

p1 = P(Mortality='True' | PatientAge='31-65' ∧ CTScanResult='Ischemic Stroke')

p2 = P(Disability='Moderate' | PatientAge='65+' ∧ MRIScanResult='Hemmorraghic Stroke')

p3 = P(StrokeType='Stroke Mimic' | PatientAge='65+' ∧ CTScanResult='Hemmorraghic Stroke' ∧ MRIScanResult='Ischemic Stroke')

p4 = P(Anticoagulants='Not used' | PatientAge='0-30')

```
ai2017@osboxes:~$ python diagnose.py
p1= 0.59485
p2= 0.26
p3= 0.1
p4= 0.5
```

Please solve the 2 tasks and hand in a file named E09_YourNumber.pdf, and send it to ai_201901@foxmail.com

4 Codes and Results

I use pomegranate in Python 3.7.4.

4.1 Burglary

Coding is trivial, except for question 2 and 5. I use chain rule and Bayes rule to convert these two probability into the form that the code can easily handle.

$$P(J\overline{M}) = P(J|\overline{M})P(\overline{M})$$

$$P(J\overline{M}|\overline{B}) = \frac{P(\overline{B}|J\overline{M})P(J\overline{M})}{P(\overline{B})} = \frac{(1 - P(B|J\overline{M}))P(J\overline{M})}{P(\overline{B})}$$

burglary.py

```
from pomegranate import *
burglary = DiscreteDistribution({'T': 0.001, 'F': 0.999})
earthquake = DiscreteDistribution({'T': 0.002, 'F': 0.998})
alarm = ConditionalProbabilityTable([
    ['T', 'T', 'T', 0.95],
    ['T', 'T', 'F', 0.05],
    ['T', 'F', 'T', 0.94],
    ['T', 'F', 'F', 0.06],
    ['F', 'T', 'T', 0.29],
    ['F', 'T', 'F', 0.71],
    ['F', 'F', 'T', 0.001],
```

```
    ['F', 'F', 'F', 0.999],
], [burglary, earthquake])
johncalls = ConditionalProbabilityTable([
    ['T', 'T', 0.90],
    ['T', 'F', 0.10],
    ['F', 'T', 0.05],
    ['F', 'F', 0.95],
], [alarm])
marycalls = ConditionalProbabilityTable([
    ['T', 'T', 0.70],
    ['T', 'F', 0.30],
    ['F', 'T', 0.01],
    ['F', 'F', 0.99],
], [alarm])

sB = State(burglary, name='burglary')  # 0
sE = State(earthquake, name='earthquake') # 1
sA = State(alarm, name='alarm')        # 2
sJ = State(johncalls, name='johncalls') # 3
sM = State(marycalls, name='marycalls') # 4

model = BayesianNetwork('Burglary Problem')
model.add_states(sB, sE, sA, sJ, sM)
model.add_transition(sB, sA)
model.add_transition(sE, sA)
model.add_transition(sA, sJ)
model.add_transition(sA, sM)
model.bake()

result1 = model.predict_proba({})[2].parameters[0]['T']
print('P(A) =', result1)
result2 = model.predict_proba({'marycalls': 'F'})[3].parameters[0]['T'] *
    model.predict_proba({})[4].parameters[0]['F']
print('P(J&&~M) =', result2)
result3 = model.predict_proba({'johncalls': 'T', 'marycalls': 'F'})[2].parameters[0]['T']
print('P(A|J&&~M) =', result3)
result4 = model.predict_proba({'alarm': 'T'})[0].parameters[0]['T']
```

```
print('P(B|A) =', result4)
result5 = model.predict_proba({'johncalls': 'T', 'marycalls': 'F'})[0].parameters[0]['T']
print('P(A|J&&~M) =', result5)
result6 = (result2 * (1-result5)) / model.predict_proba({})[0].parameters[0]['F']
print('P(J&&~M|~B) =', result6)
```

```
(base) $ python burglary.py
P(A) = 0.002516442000000935
P(J&&~M) = 0.050054875461000355
P(A|J&&~M) = 0.01357388933131146
P(B|A) = 0.3735512282818995
P(A|J&&~M) = 0.005129858133403528
P(J&&~M|~B) = 0.049847949000000266
```

4.2 Diagnosing

diagnosing.py

```
from pomegranate import *
PatientAge = DiscreteDistribution({
    '0-30': 0.10,
    '31-65': 0.30,
    '65+': 0.60
})
CTScanResult = DiscreteDistribution({
    'Ischemic Stroke': 0.7,
    'Hemmorraghic Stroke': 0.3
})
MRIScanResult = DiscreteDistribution({
    'Ischemic Stroke': 0.7,
    'Hemmorraghic Stroke': 0.3
})
Anticoagulants = DiscreteDistribution({
    'Used': 0.5,
    'Not used': 0.5
})

StrokeType = ConditionalProbabilityTable([
```

```
    ['Ischemic Stroke','Ischemic Stroke','Ischemic Stroke',0.8],
    ['Ischemic Stroke','Hemmorraghic Stroke','Ischemic Stroke',0.5],
    ['Hemmorraghic Stroke','Ischemic Stroke','Ischemic Stroke',0.5],
    ['Hemmorraghic Stroke','Hemmorraghic Stroke','Ischemic Stroke',0],

    ['Ischemic Stroke','Ischemic Stroke','Hemmorraghic Stroke',0],
    ['Ischemic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke',0.4],
    ['Hemmorraghic Stroke','Ischemic Stroke','Hemmorraghic Stroke',0.4],
    ['Hemmorraghic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke',0.9],

    ['Ischemic Stroke','Ischemic Stroke','Stroke Mimic',0.2],
    ['Ischemic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1],
    ['Hemmorraghic Stroke','Ischemic Stroke','Stroke Mimic',0.1],
    ['Hemmorraghic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1]
], [CTScanResult, MRIScanResult])

Mortality = ConditionalProbabilityTable([
    ['Ischemic Stroke', 'Used', 'False',0.28],
    ['Hemmorraghic Stroke', 'Used', 'False',0.99],
    ['Stroke Mimic', 'Used', 'False',0.1],
    ['Ischemic Stroke','Not used', 'False',0.56],
    ['Hemmorraghic Stroke', 'Not used', 'False',0.58],
    ['Stroke Mimic', 'Not used', 'False',0.05],

    ['Ischemic Stroke', 'Used' ,'True',0.72],
    ['Hemmorraghic Stroke', 'Used', 'True',0.01],
    ['Stroke Mimic', 'Used', 'True',0.9],
    ['Ischemic Stroke', 'Not used' ,'True',0.44],
    ['Hemmorraghic Stroke', 'Not used', 'True',0.42 ],
    ['Stroke Mimic', 'Not used', 'True',0.95]
], [StrokeType, Anticoagulants])

Disability = ConditionalProbabilityTable([
    ['Ischemic Stroke', '0-30','Negligible', 0.80],
    ['Hemmorraghic Stroke', '0-30','Negligible', 0.70],
    ['Stroke Mimic',      '0-30', 'Negligible',0.9],
    ['Ischemic Stroke',   '31-65','Negligible', 0.60],
```

```
    ['Hemmorraghic Stroke', '31-65','Negligible', 0.50],
    ['Stroke Mimic',      '31-65', 'Negligible',0.4],
    ['Ischemic Stroke',   '65+' , 'Negligible',0.30],
    ['Hemmorraghic Stroke', '65+' , 'Negligible',0.20],
    ['Stroke Mimic',      '65+' , 'Negligible',0.1],

    ['Ischemic Stroke',   '0-30' ,'Moderate',0.1],
    ['Hemmorraghic Stroke', '0-30' ,'Moderate',0.2],
    ['Stroke Mimic',      '0-30' ,'Moderate',0.05],
    ['Ischemic Stroke',   '31-65','Moderate',0.3],
    ['Hemmorraghic Stroke', '31-65','Moderate',0.4],
    ['Stroke Mimic',      '31-65','Moderate',0.3],
    ['Ischemic Stroke',   '65+' ,'Moderate',0.4],
    ['Hemmorraghic Stroke', '65+' ,'Moderate',0.2],
    ['Stroke Mimic',      '65+' ,'Moderate',0.1],

    ['Ischemic Stroke',   '0-30' ,'Severe',0.1],
    ['Hemmorraghic Stroke', '0-30' ,'Severe',0.1],
    ['Stroke Mimic',      '0-30' ,'Severe',0.05],
    ['Ischemic Stroke',   '31-65','Severe',0.1],
    ['Hemmorraghic Stroke', '31-65','Severe',0.1],
    ['Stroke Mimic',      '31-65','Severe',0.3],
    ['Ischemic Stroke',   '65+' ,'Severe',0.3],
    ['Hemmorraghic Stroke', '65+' ,'Severe',0.6],
    ['Stroke Mimic',      '65+' ,'Severe',0.8]
], [StrokeType, PatientAge])

sPA = State(PatientAge, name='PatientAge')    # 0
sCT = State(CTScanResult, name='CTScanResult') # 1
sMR = State(MRIScanResult, name='MRIScanResult') # 2
sAN = State(Anticoagulants, name='Anticoagulants') # 3
sDI = State(Disability, name='Disability')    # 4
sST = State(StrokeType, name='StrokeType')    # 5
sMO = State(Mortality, name='Mortality')      # 6

model = BayesianNetwork('Diagnosing')
model.add_states(sPA, sCT, sMR, sAN, sDI, sST, sMO)
```

```
model.add_transition(sPA, sDI)
model.add_transition(sCT, sST)
model.add_transition(sMR, sST)
model.add_transition(sAN, sMO)
model.add_transition(sST, sDI)
model.add_transition(sST, sMO)
model.bake()

result1 = model.predict_proba({'PatientAge': '31-65', 'CTScanResult': 'Ischemic
    Stroke'})[6].parameters[0]['True']
print('p1 = {:.5f}'.format(result1))
result2 = model.predict_proba({'PatientAge': '65+', 'MRIScanResult': 'Hemmorraghic
    Stroke'})[4].parameters[0]['Moderate']
print('p2 = {:.2f}'.format(result2))
result3 = model.predict_proba({'PatientAge': '65+', 'CTScanResult': 'Hemmorraghic Stroke',
    'MRIScanResult': 'Ischemic Stroke'})[5].parameters[0]['Stroke Mimic']
print('p3 = {:.1f}'.format(result3))
result4 = model.predict_proba({'PatientAge': '0-30'})[3].parameters[0]['Not used']
print('p4 = {:.1f}'.format(result4))
```

```
(base) $ python diagnosing.py
p1 = 0.59485
p2 = 0.26
p3 = 0.1
p4 = 0.5
```