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

E08 Bayesian Network

16337188 Runyu Ouyang

October 26, 2018

Contents

1 Pomegranate Installation **3**

2 Building Bayesian Network **3**

3 Tasks **4**

3.1 Burglary 4

3.2 Diagnosing 5

4 Codes and Results **8**

4.1 Burglary 8

4.2 Diagnosing 11

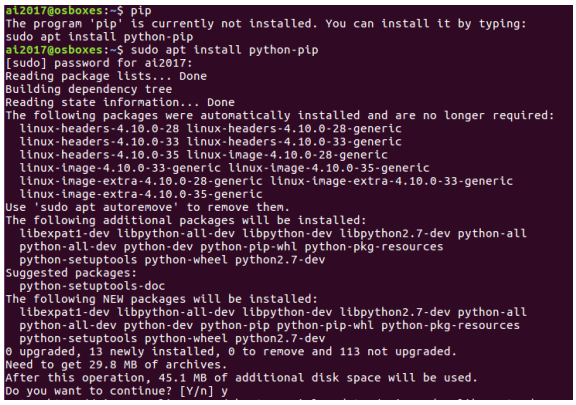
1 Pomegranate Installation

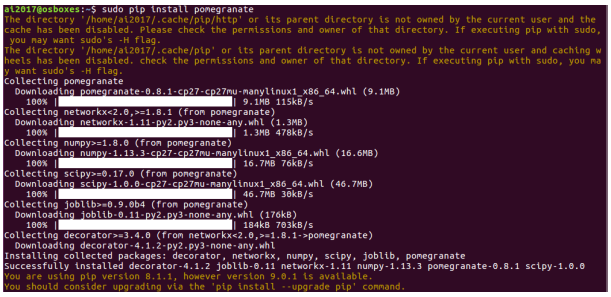
Under Linux:

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`.

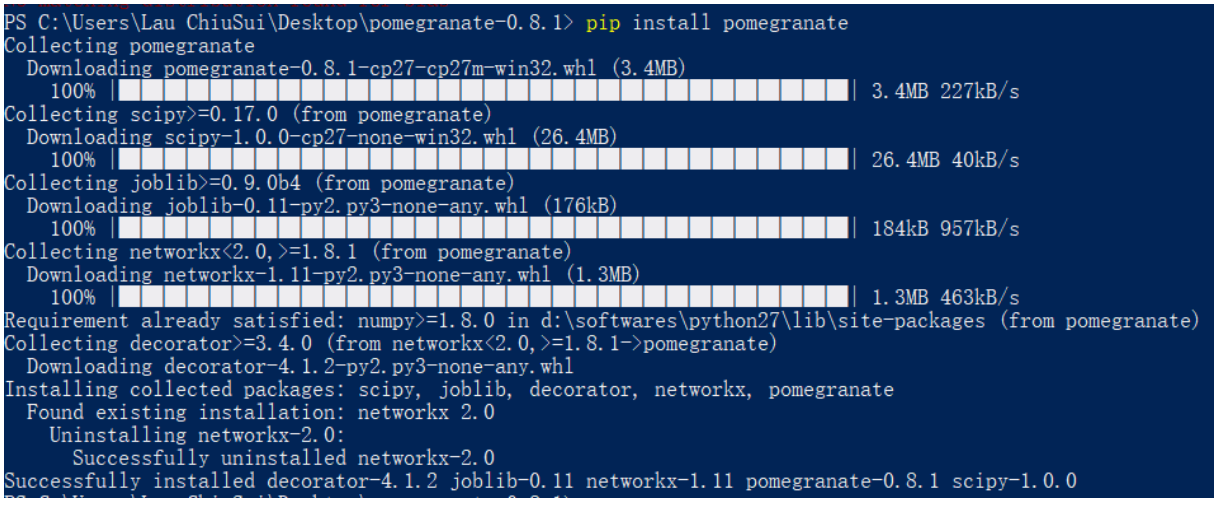




Under Windows

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

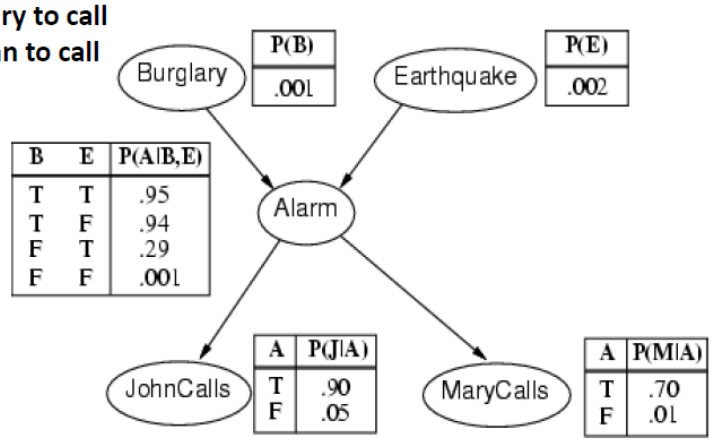
Please refer to `Tutorial_4_Bayesian_Networks.pdf`. I will explain it in class.

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

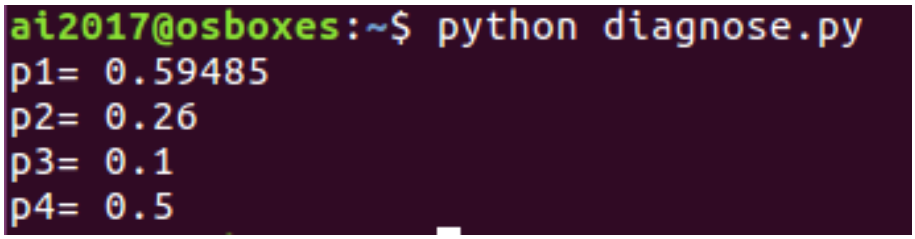
Please code to calculate the following probability value:

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')



Please solve the 2 tasks and hand in a file named E08_YourNumber.pdf, and send it to ai_2018@foxmail.com

4 Codes and Results

4.1 Burglary

- Please see the code in Bayes_Burglary.py

Bayes_Burglary.py

```
from pomegranate import *

# build the network
Burglary = DiscreteDistribution({'B': 0.001, '~B': 0.999})
Earthquake = DiscreteDistribution({'E': 0.002, '~E': 0.998})

Alarm = ConditionalProbabilityTable(
    [['B', 'E', 'A', 0.95],
    ['B', 'E', '~A', 0.05],
    ['B', '~E', 'A', 0.94],
    ['B', '~E', '~A', 0.06],
    ['~B', 'E', 'A', 0.29],
    ['~B', 'E', '~A', 0.71],
    ['~B', '~E', 'A', 0.001],
    ['~B', '~E', '~A', 0.999]], [Burglary, Earthquake])
JohnCalls = ConditionalProbabilityTable(
    [['A', 'J', 0.9],
    ['A', '~J', 0.1],
    ['~A', 'J', 0.05],
    ['~A', '~J', 0.95]], [Alarm])
MaryCalls = ConditionalProbabilityTable(
    [['A', 'M', 0.7],
    ['A', '~M', 0.3],
    ['~A', 'M', 0.01],
    ['~A', '~M', 0.99]], [Alarm])

S1 = Node(Burglary, name = 'Burglary')
S2 = Node(Earthquake, name = 'Earthquake')
S3 = Node(Alarm, name = 'Alarm')
S4 = Node(JohnCalls, name = 'JohnCalls')
S5 = Node(MaryCalls, name = 'MaryCalls')
```

```
model = BayesianNetwork("Burglary")
model.add_states(S1, S2, S3, S4, S5)
model.add_edge(S1, S3)
model.add_edge(S2, S3)
model.add_edge(S3, S4)
model.add_edge(S3, S5)
model.bake()

# P(A)
T1 = model.predict_proba({})[2].parameters[0]['A']
# P(A | J~M)
T3 = model.predict_proba({'JohnCalls':'J', 'MaryCalls':'~M'})[2].
    parameters[0]['A']
# P(A | B)
T4 = model.predict_proba({'Alarm':'A'})[0].parameters[0]['B']
# P(B | J~M)
T5 = model.predict_proba({'JohnCalls':'J', 'MaryCalls':'~M'})[0].
    parameters[0]['B']
# P(AJ~M)
P = model.probability([['B', 'E', 'A', 'J', '~M'],
                       ['~B', 'E', 'A', 'J', '~M'],
                       ['B', '~E', 'A', 'J', '~M'],
                       ['~B', '~E', 'A', 'J', '~M']]).sum()
# P(J~M) = P(AJ~M) / P(A | J~M)
T2 = P / T3
# P(~B)
P = model.predict_proba({})[0].parameters[0]['~B']
# P(J~M | ~B) = (1 - P(B | J~M)) * P(J~M) / P(~B)
T6 = (1 - T5) * T2 / P

print("P(A)=")
print(T1)
print("\nP(J~M)=")
```

```
print(T2)
print("\nP(A|J~M)=")
print(T3)
print("\nP(B|A)=")
print(T4)
print("\nP(B|J~M)=")
print(T5)
print("\nP(J~M|~B)=")
print(T6)
```

```
PS C:\Users\abc810343087\Desktop\学习\人工智能\E08> python Bayes_Burglary.py
P(A) =
0.002516442000000935

P(J && ~M) =
0.05005487546098589

P(A | J && ~M) =
0.01357388933131146

P(B | A) =
0.3735512282818995

P(B | J && ~M) =
0.005129858133403527

P(J && ~M | ~B) =
0.04984794899998586
```

Figure 1: The result of Burglary.

4.2 Diagnosing

- Please see the code in Bayes_Diagnosing.py

```
# build the network
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])

S1 = Node(PatientAge, name = 'PatientAge')
S2 = Node(CTScanResult, name = 'CTScanResult')
S3 = Node(MRIScanResult, name = 'MRIScanResult')
S4 = Node(StrokeType, name = 'StrokeType')
S5 = Node(Anticoagulants, name = 'Anticoagulants')
S6 = Node(Mortality, name = 'Mortality')
S7 = Node(Disability, name = 'Disability')

model = BayesianNetwork("Burglary")
model.add_states(S1, S2, S3, S4, S5, S6, S7)
model.add_edge(S2, S4)
model.add_edge(S3, S4)
model.add_edge(S4, S6)
model.add_edge(S5, S6)
model.add_edge(S1, S7)
model.add_edge(S4, S7)
model.bake()

P1 = model.predict_proba({'PatientAge': '31-65',
    'CTScanResult': 'Ischemic Stroke'})[5].parameters[0]['True']
```

```
print("P1 = ", P1)

P2 = model.predict_proba({'PatientAge':'65+',
    'MRIScanResult': 'Hemmorraghic Stroke'})[6].parameters[0]['Moderate'
        ]
print("P2 = ", P2)

P3 = model.predict_proba({'PatientAge':'65+',
    'CTScanResult': 'Hemmorraghic Stroke',
    'MRIScanResult': 'Ischemic Stroke'})[3].parameters[0]['Stroke Mimic'
        ]
print("P3 = ", P3)

P4 = model.predict_proba({'PatientAge': '0-30'})[4].parameters[0]['Not
    used']
print("P4 = ", P4)
```

```
PS C:\Users\abc810343087\Desktop\学习\人工智能\E08> python Bayes_Diagnosing.py
P1 =  0.5948499999999999
P2 =  0.2600000000000001
P3 =  0.10000000000000045
P4 =  0.5
```

Figure 2: The result of Diagnosing.