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

E09 Bayesian Network

17341137 Zhenpeng song

November 9, 2019

Contents

1 Pomegranate Installation **2**

2 Building Bayesian Network **2**

3 Tasks **3**

3.1 Burglary 3

3.2 Diagnosing 4

4 Codes and Results **8**

4.1 Burglary 8

4.1.1 Codes 8

4.1.2 Results 11

4.2 Diagnosing 12

4.2.1 Codes 12

4.2.2 Results 16

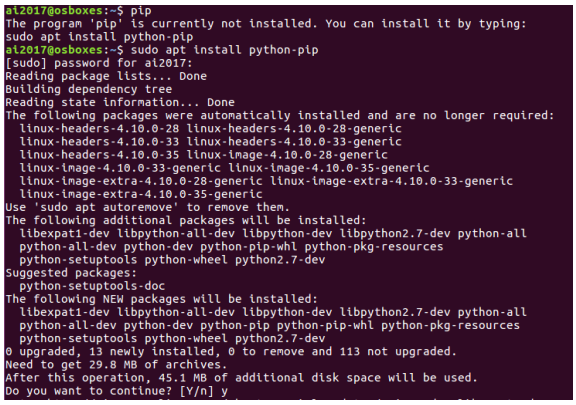
1 Pomegranate Installation

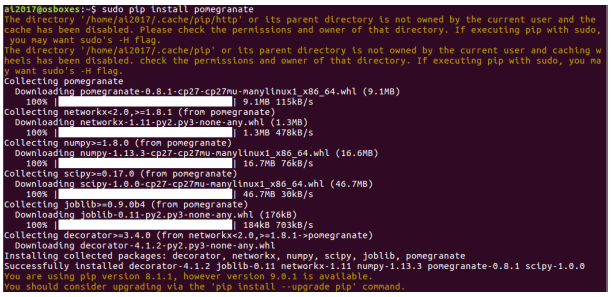
Under Linux:

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`.

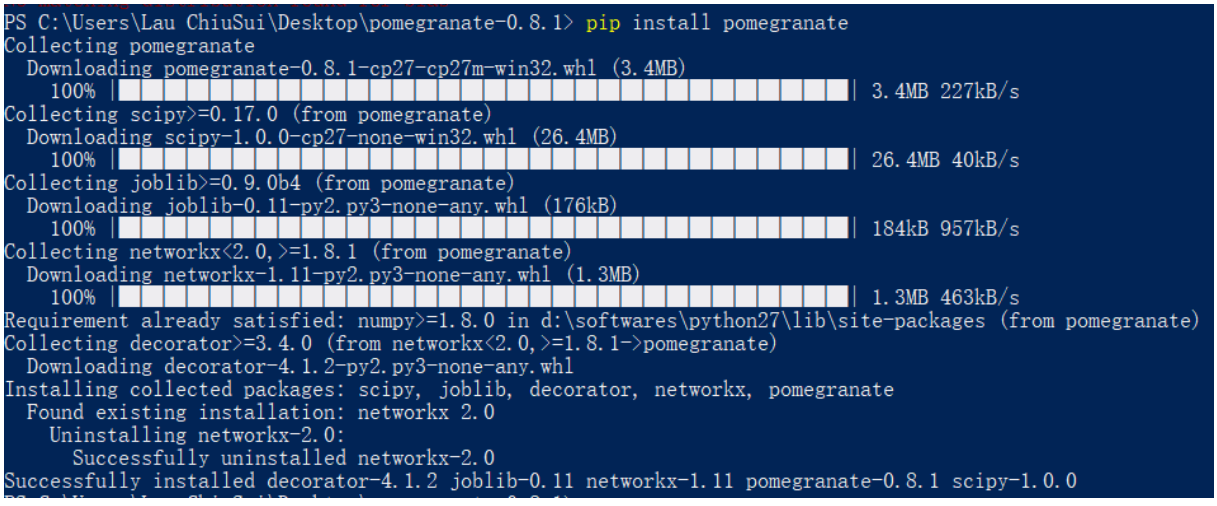




Under Windows

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

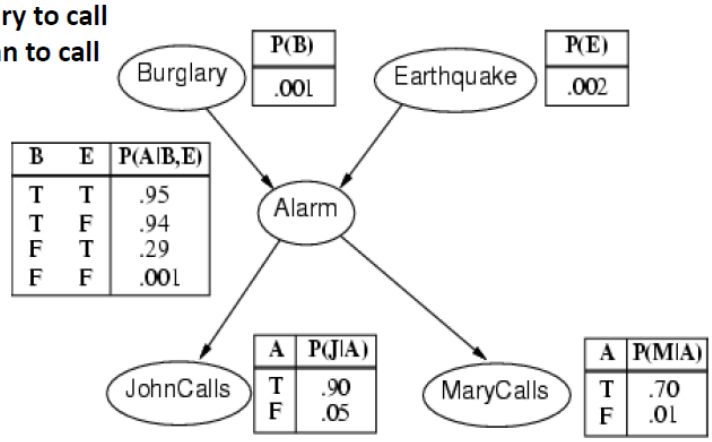
Please refer to `Tutorial_4_Bayesian_Networks.pdf`. I will explain it in class.

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],
```

```
60
61 ['Ischemic Stroke',  'Used' ,'True',0.72],
62 ['Hemmorraghic Stroke', 'Used', 'True',0.01],
63 ['Stroke Mimic', 'Used', 'True',0.9],
64 ['Ischemic Stroke',  'Not used' ,'True',0.44],
65 ['Hemmorraghic Stroke', 'Not used', 'True',0.42 ],
66 ['Stroke Mimic', 'Not used', 'True',0.95]
67
68 (7)
69 [StrokeType, PatientAge, Disability]
70
71 ['Ischemic Stroke',   '0-30','Negligible', 0.80],
72 ['Hemmorraghic Stroke', '0-30','Negligible', 0.70],
73 ['Stroke Mimic',        '0-30', 'Negligible',0.9],
74 ['Ischemic Stroke',     '31-65','Negligible', 0.60],
75 ['Hemmorraghic Stroke', '31-65','Negligible', 0.50],
76 ['Stroke Mimic',        '31-65', 'Negligible',0.4],
77 ['Ischemic Stroke',     '65+'  , 'Negligible',0.30],
78 ['Hemmorraghic Stroke', '65+'  , 'Negligible',0.20],
79 ['Stroke Mimic',        '65+'  , 'Negligible',0.1],
80
81 ['Ischemic Stroke',     '0-30' ,'Moderate',0.1],
82 ['Hemmorraghic Stroke', '0-30' ,'Moderate',0.2],
83 ['Stroke Mimic',        '0-30' ,'Moderate',0.05],
84 ['Ischemic Stroke',     '31-65','Moderate',0.3],
85 ['Hemmorraghic Stroke', '31-65','Moderate',0.4],
86 ['Stroke Mimic',        '31-65','Moderate',0.3],
87 ['Ischemic Stroke',     '65+'  ,'Moderate',0.4],
88 ['Hemmorraghic Stroke', '65+'  ,'Moderate',0.2],
89 ['Stroke Mimic',        '65+'  ,'Moderate',0.1],
90
91 ['Ischemic Stroke',     '0-30' ,'Severe',0.1],
92 ['Hemmorraghic Stroke', '0-30' ,'Severe',0.1],
```

```
93 ['Stroke Mimic',          '0-30' ,'Severe',0.05],
94 ['Ischemic Stroke',       '31-65','Severe',0.1],
95 ['Hemmorraghic Stroke',  '31-65','Severe',0.1],
96 ['Stroke Mimic',          '31-65','Severe',0.3],
97 ['Ischemic Stroke',       '65+'  ,'Severe',0.3],
98 ['Hemmorraghic Stroke',  '65+'  ,'Severe',0.6],
99 ['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

4.1 Burglary

4.1.1 Codes

```
from pomegranate import *
import pandas as pd

Burglary = DiscreteDistribution(
    {
        'T': 0.001,
        'F': 0.999
    }
)

EarthQuake = DiscreteDistribution(
    {
        'T': 0.002,
        'F': 0.998
    }
)

Alarm = ConditionalProbabilityTable(
    [
        ['T', 'T', 'T', 0.95],
        ['T', 'F', 'T', 0.94],
        ['F', 'T', 'T', 0.29],
        ['F', 'F', 'T', 0.001],

        ['T', 'T', 'F', 0.05],
        ['T', 'F', 'F', 0.06],
        ['F', 'T', 'F', 0.71],
        ['F', 'F', 'F', 0.999],
```

```
    ],
    [Burglary, EarthQuake]
)

JohnC = ConditionalProbabilityTable(
    [
        ['T', 'T', 0.9],
        ['T', 'F', 0.1],
        ['F', 'T', 0.05],
        ['F', 'F', 0.95],
    ],
    [Alarm]
)

MaryC = ConditionalProbabilityTable(
    [
        ['T', 'T', 0.7],
        ['T', 'F', 0.3],
        ['F', 'T', 0.01],
        ['F', 'F', 0.99],
    ],
    [Alarm]
)


s1 = Node(Burglary, name="Burglary")
s2 = Node(EarthQuake, name="EarthQuake")
s3 = Node(Alarm, name="Alarm")
s4 = Node(JohnC, name="JohnC")
s5 = Node(MaryC, name="MaryC")

model = BayesianNetwork("Buglary Problem")

```

```
62 model.add_states(s1, s2, s3, s4, s5)
63
64 model.add_edge(s1, s3)
65 model.add_edge(s2, s3)
66 model.add_edge(s3, s4)
67 model.add_edge(s3, s5)
68
69 model.bake()
70
71 idx = ['P(Alarm)', 'P(J && ~M)', 'P(B | A)', 'P(A | J && ~M)',
72        'P(B | J && ~M)', 'P(J && ~M | ~B)']
73
74 df = pd.DataFrame(index=idx, columns=['Probability'])
75
76 # P(A)
77 df['Probability']['P(Alarm)'] = str(
78     model.predict_proba({})[2].parameters[0]['T'])
79
80 # P(J && ~M)
81 PJ = model.predict_proba({'MaryC': 'F'})[3].parameters[0]['T']
82 PM = model.predict_proba({'JohnC': 'T'})[4].parameters[0]['F']
83 df['Probability']['P(J && ~M)'] = str(PJ * PM)
84
85 # P(A | J && ~M)
86 df['Probability']['P(A | J && ~M)'] = str(model.predict_proba(
87     {'JohnC': 'T', 'MaryC': 'F'})[2].parameters[0]['T'])
88
89 # P(B | A)
90 df['Probability']['P(B | A)'] = str(
91     model.predict_proba({'Alarm': 'T'})[0].parameters[0]['T'])
92
93 # P(B | J && ~M)
94 df['Probability']['P(B | J && ~M)'] = str(model.predict_proba(
```

```
95      {'JohnC': 'T', 'MaryC': 'F'})[0].parameters[0]['T'])
96
97  # P(J && ~M | ~B)
98  PJ = model.predict_proba(
99      {'MaryC': 'F', 'Burglary': 'F'})[3].parameters[0]['T']
100 PM = model.predict_proba(
101     {'JohnC': 'T', 'Burglary': 'F'})[4].parameters[0]['F']
102 df['Probability']['P(J && ~M | ~B)'] = str(PJ * PM)
103
104 print(df)
```

4.1.2 Results

```
λ python Sol_E09_Burglary.py
                            Probability
P(Alarm)          0.002516442000000935
P(J && ~M)        0.048624757853553476
P(B | A)            0.3735512282818995
P(A | J && ~M)     0.013573889331311 46
P(B | J && ~M)    0.005129858133403527
P(J && ~M | ~B)    0.048940729652763 58
```

4.2 Diagnosing

4.2.1 Codes

```
from pomegranate import *

PatientAge = DiscreteDistribution(
    {
        'A': 0.10,
        'B': 0.30,
        'C': 0.60
    }
)

CTScanResult = DiscreteDistribution(
    {
        'IS': 0.7,
        'HS': 0.3
    }
)

MRIScanResult = DiscreteDistribution(
    {
        'IS': 0.7,
        'HS': 0.3
    }
)

Anticoagulants = DiscreteDistribution(
    {
        'T': 0.5,
        'F': 0.5
    }
)
```

```
31
32 StrokeType = ConditionalProbabilityTable(
33     [
34         ['IS', 'IS', 'IS', 0.8],
35         ['IS', 'HS', 'IS', 0.5],
36         ['HS', 'IS', 'IS', 0.5],
37         ['HS', 'HS', 'IS', 0.0],
38
39         ['IS', 'IS', 'HS', 0.0],
40         ['IS', 'HS', 'HS', 0.4],
41         ['HS', 'IS', 'HS', 0.4],
42         ['HS', 'HS', 'HS', 0.9],
43
44         ['IS', 'IS', 'SM', 0.2],
45         ['IS', 'HS', 'SM', 0.1],
46         ['HS', 'IS', 'SM', 0.1],
47         ['HS', 'HS', 'SM', 0.1],
48     ],
49     [CTScanResult, MRIScanResult]
50 )
51
52 Mortality = ConditionalProbabilityTable(
53     [
54         ['IS', 'T', 'F', 0.28],
55         ['HS', 'T', 'F', 0.99],
56         ['SM', 'T', 'F', 0.10],
57         ['IS', 'F', 'F', 0.56],
58         ['HS', 'F', 'F', 0.58],
59         ['SM', 'F', 'F', 0.05],
60         ['IS', 'T', 'T', 0.72],
61         ['HS', 'T', 'T', 0.01],
62         ['SM', 'T', 'T', 0.90],
63         ['IS', 'F', 'T', 0.44],
```

```
        ['HS', 'F', 'T', 0.42],
        ['SM', 'F', 'T', 0.95],
    ],
    [StrokeType, Anticoagulants]
)

Disability = ConditionalProbabilityTable(
    [
        ['IS', 'A', 'N', 0.80],
        ['HS', 'A', 'N', 0.70],
        ['SM', 'A', 'N', 0.90],
        ['IS', 'B', 'N', 0.60],
        ['HS', 'B', 'N', 0.50],
        ['SM', 'B', 'N', 0.40],
        ['IS', 'C', 'N', 0.30],
        ['HS', 'C', 'N', 0.20],
        ['SM', 'C', 'N', 0.10],

        ['IS', 'A', 'M', 0.10],
        ['HS', 'A', 'M', 0.20],
        ['SM', 'A', 'M', 0.05],
        ['IS', 'B', 'M', 0.30],
        ['HS', 'B', 'M', 0.40],
        ['SM', 'B', 'M', 0.30],
        ['IS', 'C', 'M', 0.40],
        ['HS', 'C', 'M', 0.20],
        ['SM', 'C', 'M', 0.10],

        ['IS', 'A', 'S', 0.10],
        ['HS', 'A', 'S', 0.10],
        ['SM', 'A', 'S', 0.05],
        ['IS', 'B', 'S', 0.10],
        ['HS', 'B', 'S', 0.10],
```

```
            ['SM', 'B', 'S', 0.30],
            ['IS', 'C', 'S', 0.30],
            ['HS', 'C', 'S', 0.60],
            ['SM', 'C', 'S', 0.80],
        ],
        [StrokeType, PatientAge]
)


s1 = Node(PatientAge, name="PatientAge")
s2 = Node(CTScanResult, name="CTScanResult")
s3 = Node(MRIScanResult, name="MRIScanResult")
s4 = Node(Anticoagulants, name="Anticoagulants")
s5 = Node(StrokeType, name="StrokeType")
s6 = Node(Mortality, name="Mortality")
s7 = Node(Disability, name="Disability")

model = BayesianNetwork("Diagnosing Problem")

model.add_states(s1, s2, s3, s4, s5, s6, s7)

model.add_edge(s2, s5)
model.add_edge(s3, s5)

model.add_edge(s5, s6)
model.add_edge(s4, s6)

model.add_edge(s5, s7)
model.add_edge(s1, s7)

model.bake()


```

```
130 # P(1)
131 print('P(1) = ', model.predict_proba(
132     {'PatientAge': 'B', 'CTScanResult': 'IS'})
133         [5].parameters[0]['T'])
134
135 # P(2)
136 print('P(2) = ', model.predict_proba(
137     {'PatientAge': 'C', 'MRIScanResult': 'HS'})
138         [6].parameters[0]['M'])
139
140 # P(3)
141 print('P(3) = ', model.predict_proba(
142     {'PatientAge': 'C', 'CTScanResult': 'HS',
143     'MRIScanResult': 'IS'})[4].parameters[0]['SM'])
144
145 # P(4)
146 print('P(4) = ', model.predict_proba(
147     {'PatientAge': 'A'})[3].parameters[0]['F'])
```

4.2.2 Results

```
λ python Sol_E09_Diagnosing.py
P(1) =  0.5948499999999999
P(2) =  0.2600000000000001
P(3) =  0.10000000000000044
P(4) =  0.5
```