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

E09 Bayesian Network

孙新梦 1834149

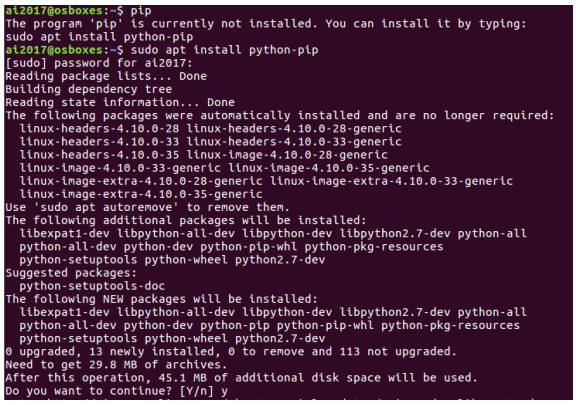
October 26, 2020

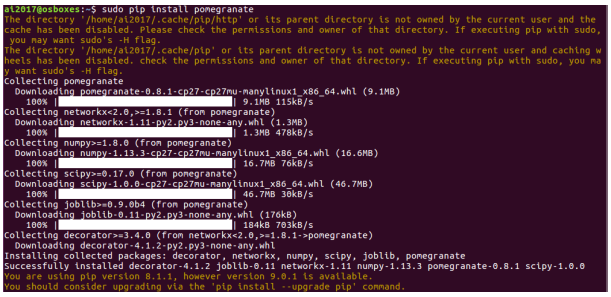
目录

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`.

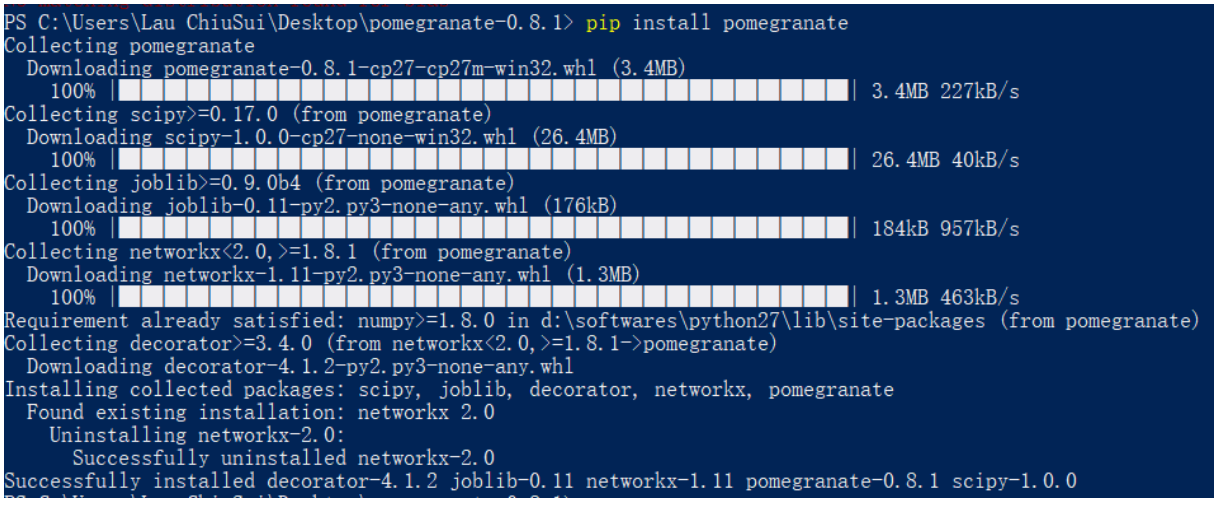




Under Windows

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

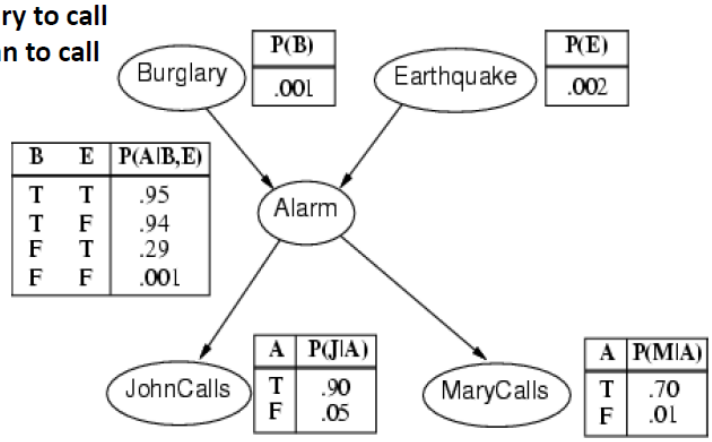
Please refer to `Tutorial_4_Bayesian_Networks.pdf`. I will explain it in class.

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]
```

```
26
27 (4)
28 [Anticoagulants]
29
30 ['Used',0.5],
31 ['Not used',0.5]
32
33 (5)
34 [CTScanResult, MRIScanResult,StrokeType])
35
36 ['Ischemic Stroke','Ischemic Stroke','Ischemic Stroke',0.8],
37 ['Ischemic Stroke','Hemmorraghic Stroke','Ischemic Stroke',0.5],
38 ['Hemmorraghic Stroke','Ischemic Stroke','Ischemic Stroke',0.5],
39 ['Hemmorraghic Stroke','Hemmorraghic Stroke','Ischemic Stroke
       ',0],
40
41 ['Ischemic Stroke','Ischemic Stroke','Hemmorraghic Stroke',0],
42 ['Ischemic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke
       ',0.4],
43 ['Hemmorraghic Stroke','Ischemic Stroke','Hemmorraghic Stroke
       ',0.4],
44 ['Hemmorraghic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke
       ',0.9],
45
46 ['Ischemic Stroke','Ischemic Stroke','Stroke Mimic',0.2],
47 ['Ischemic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1],
48 ['Hemmorraghic Stroke','Ischemic Stroke','Stroke Mimic',0.1],
49 ['Hemmorraghic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1],
50
51 (6)
52 [StrokeType, Anticoagulants, Mortality]
53
54 ['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

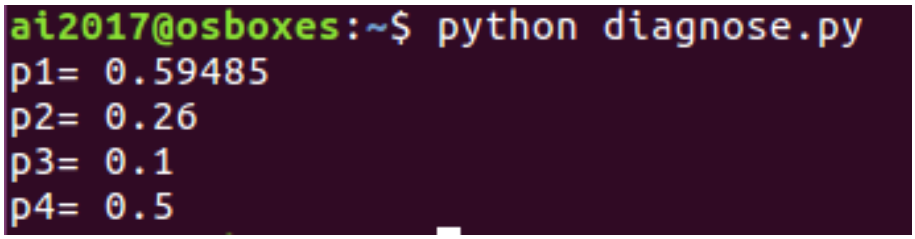
Please code to calculate the following probability value:

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_2020@foxmail.com

4 Codes and Results

4.1 1.Burglary

Code

```
from pomegranate import *
import pandas as pd
```

```
102
103 Burglary = DiscreteDistribution(
104     {
105         'T': 0.001,
106         'F': 0.999
107     }
108 )
109
110 EarthQuake = DiscreteDistribution(
111     {
112         'T': 0.002,
113         'F': 0.998
114     }
115 )
116
117 Alarm = ConditionalProbabilityTable(
118     [
119         ['T', 'T', 'T', 0.95],
120         ['T', 'F', 'T', 0.94],
121         ['F', 'T', 'T', 0.29],
122         ['F', 'F', 'T', 0.001],
123
124         ['T', 'T', 'F', 0.05],
125         ['T', 'F', 'F', 0.06],
126         ['F', 'T', 'F', 0.71],
127         ['F', 'F', 'F', 0.999],
128     ],
129     [Burglary, EarthQuake]
130 )
131
132 JohnC = ConditionalProbabilityTable(
133     [
134         ['T', 'T', 0.9],
```

```
135             ['T', 'F', 0.1],
136             ['F', 'T', 0.05],
137             ['F', 'F', 0.95],
138         ],
139         [Alarm]
140 )
141
142 MaryC = ConditionalProbabilityTable(
143         [
144             ['T', 'T', 0.7],
145             ['T', 'F', 0.3],
146             ['F', 'T', 0.01],
147             ['F', 'F', 0.99],
148         ],
149         [Alarm]
150 )
151
152
153 s1 = Node(Burglary, name="Burglary")
154 s2 = Node(EarthQuake, name="EarthQuake")
155 s3 = Node(Alarm, name="Alarm")
156 s4 = Node(JohnC, name="JohnC")
157 s5 = Node(MaryC, name="MaryC")
158
159 model = BayesianNetwork("Buglary Problem")
160
161 model.add_states(s1, s2, s3, s4, s5)
162
163 model.add_edge(s1, s3)
164 model.add_edge(s2, s3)
165 model.add_edge(s3, s4)
166 model.add_edge(s3, s5)
167
```

```
168 model.bake()
169
170 idx = ['P(Alarm)', 'P(J && ~M)', 'P(B | A)', 'P(A | J && ~M)',
171         'P(B | J && ~M)', 'P(J && ~M | ~B)']
172
173 df = pd.DataFrame(index=idx, columns=['Probability'])
174
175 # P(A)
176 df['Probability']['P(Alarm)'] = str(
177     model.predict_proba({})[2].parameters[0]['T'])
178
179 # P(J && ~M)
180 PJ = model.predict_proba({'MaryC': 'F'})[3].parameters[0]['T']
181 PM = model.predict_proba({'JohnC': 'T'})[4].parameters[0]['F']
182 df['Probability']['P(J && ~M)'] = str(PJ * PM)
183
184 # P(A | J && ~M)
185 df['Probability']['P(A | J && ~M)'] = str(model.predict_proba(
186     {'JohnC': 'T', 'MaryC': 'F'})[2].parameters[0]['T'])
187
188 # P(B | A)
189 df['Probability']['P(B | A)'] = str(
190     model.predict_proba({'Alarm': 'T'})[0].parameters[0]['T'])
191
192 # P(B | J && ~M)
193 df['Probability']['P(B | J && ~M)'] = str(model.predict_proba(
194     {'JohnC': 'T', 'MaryC': 'F'})[0].parameters[0]['T'])
195
196 # P(J && ~M | ~B)
197 PJ = model.predict_proba({'MaryC': 'F', 'Burglary': 'F'})[3].
    parameters[0]['T']
198 PM = model.predict_proba({'JohnC': 'T', 'Burglary': 'F'})[4].
    parameters[0]['F']
```

```
df['Probability']['P(J && ~M | ~B)'] = str(PJ * PM)

print(df)
```

Result

```
                                Probability
P(Alarm)          0.002516442000000935
P(J && ~M)        0.048624757853553476
P(B | A)            0.3735512282818995
P(A | J && ~M)     0.01357388933131146
P(B | J && ~M)    0.005129858133403527
P(J && ~M | ~B)    0.04894072965276358
```

4.2 2.Diagnosis

Code

```

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
    }
)

StrokeType = ConditionalProbabilityTable(
    [
        ['IS', 'IS', 'IS', 0.8],
        ['IS', 'HS', 'IS', 0.5],
        ['HS', 'IS', 'IS', 0.5],
        ['HS', 'HS', 'IS', 0.0],

        ['IS', 'IS', 'HS', 0.0],
        ['IS', 'HS', 'HS', 0.4],
        ['HS', 'IS', 'HS', 0.4],
        ['HS', 'HS', 'HS', 0.9],

        ['IS', 'IS', 'SM', 0.2],
        ['IS', 'HS', 'SM', 0.1],
        ['HS', 'IS', 'SM', 0.1],
        ['HS', 'HS', 'SM', 0.1],
    ],
```

```
    [CTScanResult, MRIScanResult]
)

Mortality = ConditionalProbabilityTable(
    [
        ['IS', 'T', 'F', 0.28],
        ['HS', 'T', 'F', 0.99],
        ['SM', 'T', 'F', 0.10],
        ['IS', 'F', 'F', 0.56],
        ['HS', 'F', 'F', 0.58],
        ['SM', 'F', 'F', 0.05],
        ['IS', 'T', 'T', 0.72],
        ['HS', 'T', 'T', 0.01],
        ['SM', 'T', 'T', 0.90],
        ['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")
```

```
318
319 model.add_states(s1, s2, s3, s4, s5, s6, s7)
320
321 model.add_edge(s2, s5)
322 model.add_edge(s3, s5)
323
324 model.add_edge(s5, s6)
325 model.add_edge(s4, s6)
326
327 model.add_edge(s5, s7)
328 model.add_edge(s1, s7)
329
330 model.bake()
331
332
333 # P(1)
334 print('P(1) = ', model.predict_proba(
335     {'PatientAge': 'B', 'CTScanResult': 'IS'})[5].parameters[0]['T
        ']))
336
337 # P(2)
338 print('P(2) = ', model.predict_proba(
339     {'PatientAge': 'C', 'MRIScanResult': 'HS'})[6].parameters[0]['
        M']))
340
341 # P(3)
342 print('P(3) = ', model.predict_proba(
343     {'PatientAge': 'C', 'CTScanResult': 'HS', 'MRIScanResult': 'IS
        '})[4].parameters[0]['SM']))
344
345 # P(4)
346 print('P(4) = ', model.predict_proba(
347     {'PatientAge': 'A'})[3].parameters[0]['F'])
```

```
348
349 # Helper
350 print('P1 = ', model.predict_proba(
351     {})[5].parameters[0]['T'])
352
353 print('P2 = ', model.predict_proba(
354     {'PatientAge': 'B'})[1].parameters[0]['IS'])
355
356 print('P3 = ', model.predict_proba(
357     {'PatientAge': 'C', 'CTScanResult': 'HS', 'MRIScanResult': 'IS
        '})[4].parameters[0]['HS'])
358
359 print('P4 = ', model.predict_proba(
360     {})[6].parameters[0]['N'])
```

Result

```
P(1) =  0.5948499999999999
P(2) =  0.2600000000000001
P(3) =  0.10000000000000044
P(4) =  0.5
```