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

E09 Bayesian Network

17341015 Hongzheng Chen

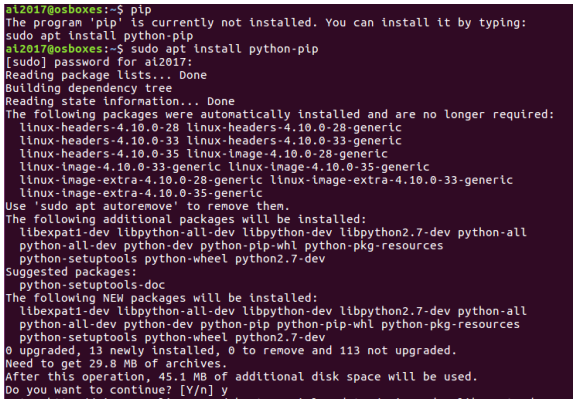
November 7, 2019

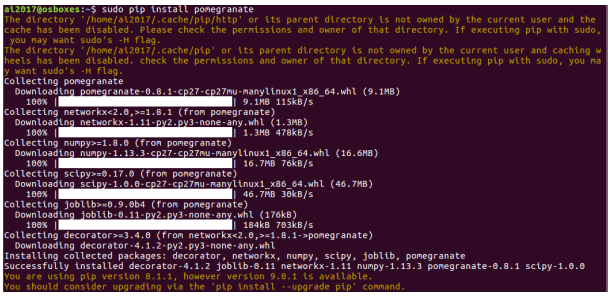
Contents

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`.

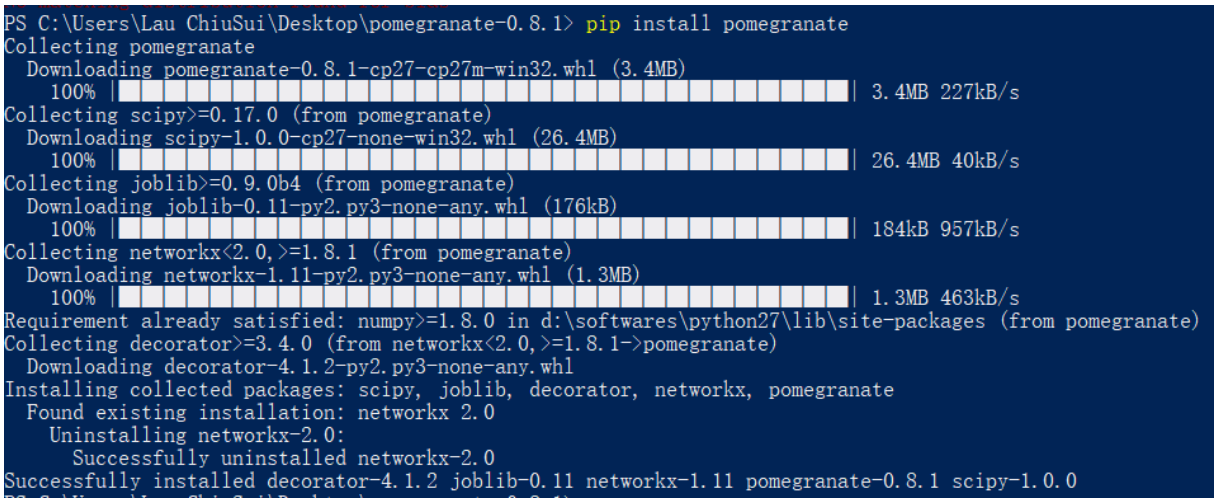




Under Windows

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.

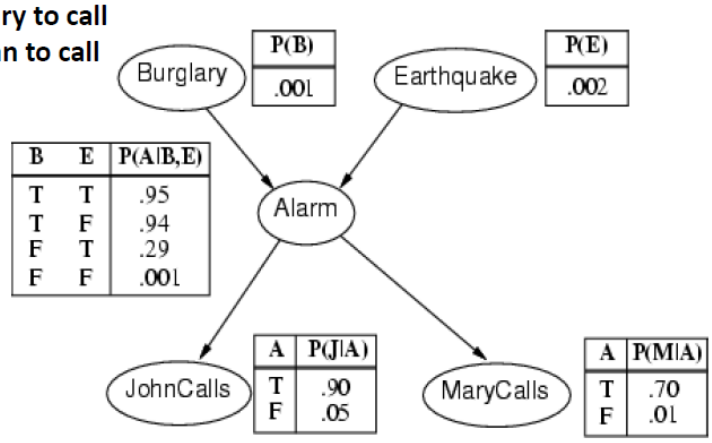


2 Building Bayesian Network

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

- **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



3 Tasks

3.1 Burglary

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='...' ∧ 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

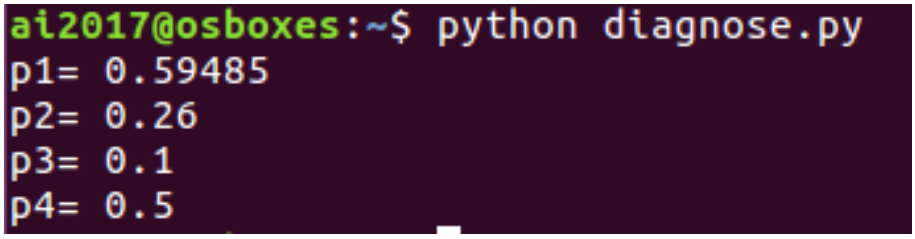
Please code to calculate the following probability value:

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')

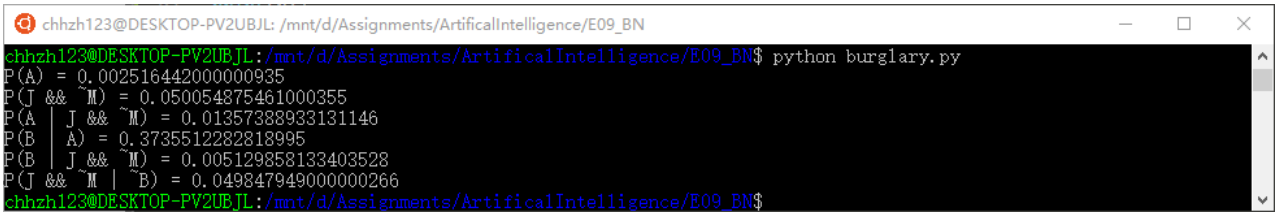
p4 = P(Anticoagulants='Not used' | PatientAge='0-30')



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

4 Codes and Results

The codes use Python 3!



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

s1 = State(burglary, name="burglary")
s2 = State(earthquake, name="earthquake")
s3 = State(alarm, name="alarm")
s4 = State(johncalls, name="johncalls")
s5 = State(marycalls, name="marycalls")
```

```
model = BayesianNetwork("Burglary")

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

model.add_transition(s1,s3)
model.add_transition(s2,s3)
model.add_transition(s3,s4)
model.add_transition(s3,s5)
model.bake()

marginals = model.predict_proba({})
# P(A)
print("P(A) = {}".format(marginals[2].parameters[0]["T"]))
# P(J&&~M) = P(J|~M)P(~M)
j_nm = model.predict_proba({'marycalls':'F'})[3].parameters[0]["T"] * marginals[4].
    ↪ parameters[0]["F"]
print("P(J && ~M) = {}".format(j_nm))
# P(A|J&&~M)
print("P(A | J && ~M) = {}".format(model.predict_proba({'johncalls':'T','marycalls':'F'})
    ↪ [2].parameters[0]["T"]))
# P(B|A)
print("P(B | A) = {}".format(model.predict_proba({'alarm':'T'})[0].parameters[0]["T"]))
# P(B|J&&~M)
b_c_j_nm = model.predict_proba({'johncalls':'T','marycalls':'F'})[0].parameters[0]["T"]
print("P(B | J && ~M) = {}".format(b_c_j_nm))
# P(J&&~M|~B) = P(~B && J && ~M) / P(~B)
#             = P(~B | J && ~M) P(J && ~M) / P(~B)
#             = (1- P(B | J && ~M)) P(J && ~M) / P(~B)
print("P(J && ~M | ~B) = {}".format((1-b_c_j_nm) * j_nm / marginals[0].parameters[0]["F"]))
```

```
chhzh123@DESKTOP-PV2UBJL: /mnt/d/Assignments/ArtificalIntelligence/E09_BN
chhzh123@DESKTOP-PV2UBJL:/mnt/d/Assignments/ArtificalIntelligence/E09_BN$ python diagnosing.py
0.5948499999999999
0.2600000000000001
0.10000000000000044
0.5
chhzh123@DESKTOP-PV2UBJL:/mnt/d/Assignments/ArtificalIntelligence/E09_BN$
```

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])

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

model = BayesianNetwork("Diagnosing")
```

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

model.add_transition(s2,s4)
model.add_transition(s3,s4)

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

model.add_transition(s1,s7)
model.add_transition(s4,s7)

model.bake()

marginals = model.predict_proba({})

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