In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pyGM as gm
```

In [2]:
```
data = np.genfromtxt('179-hw5-riskdata.csv', delimiter=",",names=True)
data_int = np.array([list(xj) for xj in data], dtype=int)-1
nTrain = int(.75*len(data_int))
train = data_int[:nTrain]
valid = data_int[nTrain:]
print(data)
```

```
[(6., 2., 2., 4., 2., 2., 1., 2., 2., 3.)
 (2., 2., 1., 3., 1., 2., 1., 2., 2., 3.)
 (6., 1., 2., 3., 1., 2., 3., 2., 2., 3.) ...
 (2., 2., 2., 3., 1., 2., 3., 2., 2., 3.)
 (1., 2., 2., 4., 1., 2., 1., 2., 2., 3.)
 (1., 2., 2., 2., 1., 2., 3., 2., 2., 3.)]
```

In [3]:
```
income = gm.Var(0,8)
smoke = gm.Var(1,2)
cholesterol = gm.Var(2,2)
bmi = gm.Var(3,4)
exercise = gm.Var(4,2)
attack = gm.Var(5,2)
bp = gm.Var(6,4)
angina = gm.Var(7,2)
stroke = gm.Var(8,2)
diabetes = gm.Var(9,4)

X = [income,smoke,cholesterol,bmi,exercise,attack,bp,angina,stroke,diabetes]

pI = gm.Factor([income], 0.0)
pEgI = gm.Factor([income,exercise], 0.0)
pSgI = gm.Factor([income,smoke], 0.0)
pBgIE = gm.Factor([income,exercise,bmi], 0.0)
pCgIES = gm.Factor([income,smoke,cholesterol,exercise], 0.0)
pBPgIES = gm.Factor([income,smoke,exercise,bp], 0.0)
pDgB = gm.Factor([bmi,diabetes], 0.0)
pSTgBBPC = gm.Factor([cholesterol,bmi,bp,stroke], 0.0)
pAgBBPC = gm.Factor([cholesterol,bmi,attack,bp], 0.0)
pANgBBPC = gm.Factor([cholesterol,bmi,bp,angina], 0.0)
#numel()

for case in train:
    pI[case[0]] += 1
    pEgI.setValueMap({income:case[0],exercise:case[4]},pEgI.valueMap({income:case[0],exercise:case[4]})+1)
    pSgI.setValueMap({income:case[0],smoke:case[1]},pSgI.valueMap({income:case[0],smoke:case[1]})+1)
    pBgIE.setValueMap({income:case[0],bmi:case[3],exercise:case[4]},pBgIE.valueMap({income:case[0],bmi:case[3],exercise:case[4]})+1)
    pCgIES.setValueMap({income:case[0],smoke:case[1],exercise:case[4],cholesterol:case[2]},pCgIES.valueMap({income:case[0],smoke:case[1],exe
    pBPgIES.setValueMap({income:case[0],smoke:case[1],exercise:case[4],bp:case[6]},pBPgIES.valueMap({income:case[0],smoke:case[1],exercise:c
    pDgB.setValueMap({bmi:case[3],diabetes:case[9]},pDgB.valueMap({bmi:case[3],diabetes:case[9]})+1)
    pSTgBBPC.setValueMap({bmi:case[3],bp:case[6],stroke:case[8],cholesterol:case[2]},pSTgBBPC.valueMap({bmi:case[3],bp:case[6],stroke:case[8
    pAgBBPC.setValueMap({bmi:case[3],bp:case[6],attack:case[5],cholesterol:case[2]},pAgBBPC.valueMap({bmi:case[3],bp:case[6],attack:case[5],
    pANgBBPC.setValueMap({bmi:case[3],bp:case[6],angina:case[7],cholesterol:case[2]},pANgBBPC.valueMap({bmi:case[3],bp:case[6],angina:case[7
factors = [pI,pEgI,pSgI,pBgIE,pCgIES,pBPgIES,pDgB,pSTgBBPC,pAgBBPC,pANgBBPC]
for x in range(10):
    factors[x] /= nTrain
```

In [4]:
```
print(pI.table)
print(pEgI.table)
```

```
[0.0486259  0.0589277  0.07324057 0.09263729 0.1156428  0.15099311
 0.16441857 0.29551406]
[[0.03036506 0.01826084]
 [0.0359143  0.02301339]
 [0.04640936 0.02683121]
 [0.06149132 0.03114598]
 [0.08123117 0.03441163]
 [0.11310284 0.03789026]
 [0.13016865 0.03424993]
 [0.25168016 0.04383391]]
```

In [5]:
```
total = 0
for x in range(10):
    temp = 1
    for num in factors[x].dims():
        temp *= num
    total += temp
print("Total probabilities for newtwork:",total)

joint = pI*pEgI*pSgI*pBgIE*pCgIES*pBPgIES*pDgB*pSTgBBPC*pAgBBPC*pANgBBPC
total = 1
for num in joint.dims():
        total *= num
print("Total probabilities for joint distribution:",total)
```

```
Total probabilities for newtwork: 504
Total probabilities for joint distribution: 32768
```

In [6]:
```
aLL = 0
for case in train:
    aLL += np.log(pI[case[0]])
    aLL += np.log(pEgI.valueMap({income:case[0],exercise:case[4]}))
    aLL += np.log(pSgI.valueMap({income:case[0],smoke:case[1]}))
    aLL += np.log(pBgIE.valueMap({income:case[0],bmi:case[3],exercise:case[4]}))
    aLL += np.log(pCgIES.valueMap({income:case[0],smoke:case[1],exercise:case[4],cholesterol:case[2]}))
    aLL += np.log(pBPgIES.valueMap({income:case[0],smoke:case[1],exercise:case[4],bp:case[6]}))
    aLL += np.log(pDgB.valueMap({bmi:case[3],diabetes:case[9]}))
    aLL += np.log(pSTgBBPC.valueMap({bmi:case[3],bp:case[6],stroke:case[8],cholesterol:case[2]}))
    aLL += np.log(pAgBBPC.valueMap({bmi:case[3],bp:case[6],attack:case[5],cholesterol:case[2]}))
    aLL += np.log(pANgBBPC.valueMap({bmi:case[3],bp:case[6],angina:case[7],cholesterol:case[2]}))
aLL /= len(train)*10 #times 10 to account for the fact that there are 10 values to add together per the number of cases
print("average log-likelihood (training data):",aLL)
```

```
average log-likelihood (training data): -2.8165109877041465
```

In [7]:

```
aLL = 0
for case in valid:
    aLL += np.log(pI[case[0]])
    aLL += np.log(pEgI.valueMap({income:case[0],exercise:case[4]}))
    aLL += np.log(pSgI.valueMap({income:case[0],smoke:case[1]}))
    aLL += np.log(pBgIE.valueMap({income:case[0],bmi:case[3],exercise:case[4]}))
    aLL += np.log(pCgIES.valueMap({income:case[0],smoke:case[1],exercise:case[4],cholesterol:case[2]}))
    aLL += np.log(pBPgIES.valueMap({income:case[0],smoke:case[1],exercise:case[4],bp:case[6]}))
    aLL += np.log(pDgB.valueMap({bmi:case[3],diabetes:case[9]}))
    aLL += np.log(pSTgBBPC.valueMap({bmi:case[3],bp:case[6],stroke:case[8],cholesterol:case[2]}))
    aLL += np.log(pAgBBPC.valueMap({bmi:case[3],bp:case[6],attack:case[5],cholesterol:case[2]}))
    aLL += np.log(pANgBBPC.valueMap({bmi:case[3],bp:case[6],angina:case[7],cholesterol:case[2]}))
aLL /= len(valid)*10 #times 10 to account for the fact that there are 10 values to add together per the number of cases
print("average log-likelihood (validation data):",aLL)
#yes, there is a zero probability and the np module cannot go passed that
```

```
<ipython-input-7-ce4ecf8a81fe>:11: RuntimeWarning: divide by zero encountered in log
  aLL += np.log(pAgBBPC.valueMap({bmi:case[3],bp:case[6],attack:case[5],cholesterol:case[2]}))
average log-likelihood (validation data): -inf
```

In [8]:

```
#re-counting to normalize
#X = [income,smoke,cholesterol,bmi,exercise,attack,bp,angina,stroke,diabetes]
for case in train:
    pI[case[0]] += 1
    pEgI.setValueMap({income:case[0],exercise:case[4]},pEgI.valueMap({income:case[0],exercise:case[4]})+1)
    pSgI.setValueMap({income:case[0],smoke:case[1]},pSgI.valueMap({income:case[0],smoke:case[1]})+1)
    pBgIE.setValueMap({income:case[0],bmi:case[3],exercise:case[4]},pBgIE.valueMap({income:case[0],bmi:case[3],exercise:case[4]})+1)
    pCgIES.setValueMap({income:case[0],smoke:case[1],exercise:case[4],cholesterol:case[2]},pCgIES.valueMap({income:case[0],smoke:case[1],exe
    pBPgIES.setValueMap({income:case[0],smoke:case[1],exercise:case[4],bp:case[6]},pBPgIES.valueMap({income:case[0],smoke:case[1],exercise:c
    pDgB.setValueMap({bmi:case[3],diabetes:case[9]},pDgB.valueMap({bmi:case[3],diabetes:case[9]})+1)
    pSTgBBPC.setValueMap({bmi:case[3],bp:case[6],stroke:case[8],cholesterol:case[2]},pSTgBBPC.valueMap({bmi:case[3],bp:case[6],stroke:case[8
    pAgBBPC.setValueMap({bmi:case[3],bp:case[6],attack:case[5],cholesterol:case[2]},pAgBBPC.valueMap({bmi:case[3],bp:case[6],attack:case[5],
    pANgBBPC.setValueMap({bmi:case[3],bp:case[6],angina:case[7],cholesterol:case[2]},pANgBBPC.valueMap({bmi:case[3],bp:case[6],angina:case[7
#adding one, as if i observed everything at least once
pI += 1
pEgI += 1
pSgI += 1
pBgIE += 1
pCgIES += 1
pBPgIES += 1
pDgB += 1
pSTgBBPC += 1
pAgBBPC += 1
pANgBBPC += 1
#and finally re-normalizing with the additional observation
for x in range(10):
    factors[x] /= (nTrain+1)
```

In [9]:

```
aLLT = 0
for case in train:
    aLLT += np.log(pI[case[0]])
    aLLT += np.log(pEgI.valueMap({income:case[0],exercise:case[4]}))
    aLLT += np.log(pSgI.valueMap({income:case[0],smoke:case[1]}))
    aLLT += np.log(pBgIE.valueMap({income:case[0],bmi:case[3],exercise:case[4]}))
    aLLT += np.log(pCgIES.valueMap({income:case[0],smoke:case[1],exercise:case[4],cholesterol:case[2]}))
    aLLT += np.log(pBPgIES.valueMap({income:case[0],smoke:case[1],exercise:case[4],bp:case[6]}))
    aLLT += np.log(pDgB.valueMap({bmi:case[3],diabetes:case[9]}))
    aLLT += np.log(pSTgBBPC.valueMap({bmi:case[3],bp:case[6],stroke:case[8],cholesterol:case[2]}))
    aLLT += np.log(pAgBBPC.valueMap({bmi:case[3],bp:case[6],attack:case[5],cholesterol:case[2]}))
    aLLT += np.log(pANgBBPC.valueMap({bmi:case[3],bp:case[6],angina:case[7],cholesterol:case[2]}))
aLLT /= len(train)*10 #times 10 to account for the fact that there are 10 values to add together per the number of cases
print("average log-likelihood (training data):",aLLT)
```

```
average log-likelihood (training data): -2.8163173395493755
```

In [10]:

```
aLLV = 0
for case in valid:
    aLLV += np.log(pI[case[0]])
    aLLV += np.log(pEgI[case[0],case[4]])
    aLLV += np.log(pSgI[case[0],case[1]])
    aLLV += np.log(pBgIE[case[0],case[3],case[4]])
    aLLV += np.log(pCgIES[case[0],case[1],case[2],case[4]])
    aLLV += np.log(pBPgIES[case[0],case[1],case[4],case[6]])
    aLLV += np.log(pDgB[case[3],case[9]])
    aLLV += np.log(pSTgBBPC[case[2],case[3],case[6],case[8]])
    aLLV += np.log(pAgBBPC[case[2],case[3],case[5],case[6]])
    aLLV += np.log(pANgBBPC[case[2],case[3],case[6],case[7]])
aLLV /= len(valid)*10 #times 10 to account for the fact that there are 10 values to add together per the number of cases
print("average log-likelihood (validation data):",aLLV)
```

```
average log-likelihood (validation data): -2.837726984908465
```

In [11]:
```
#factors = [pI,pEgI,pSgI,pBgIE,pCgIES,pBPgIES,pDgB,pSTgBBPC,pAgBBPC,pANgBBPC]
#X = [income,smoke,cholesterol,bmi,exercise,attack,bp,angina,stroke,diabetes]
Mi = pI
Msmo = pSgI.sum([X[0]])
Mcho = pCgIES.sum([X[0],X[4],X[1]])
Mbmi = pBgIE.sum([X[4],X[0]])
Mexe = pEgI.sum([X[0]])
Matt = pAgBBPC.sum([X[2],X[3],X[6]])
Mbp = pBPgIES.sum([X[0],X[4],X[1]])
Mang = pANgBBPC.sum([X[2],X[3],X[6]])
Mstr = pSTgBBPC.sum([X[2],X[3],X[6]])
Mdia = pDgB.sum([X[3]])
marginals = [Mi,Msmo,Mcho,Mbmi,Mexe,Matt,Mbp,Mang,Mstr,Mdia]
for n in range(10):
    print("Marginal Probability for",X[n],marginals[n].table)
    print()
```

```
Marginal Probability for 0 [0.04862985 0.05893164 0.07324451 0.09264124 0.11564674 0.15099705
 0.16442252 0.29551801]

Marginal Probability for 1 [0.47570874 0.52435436]

Marginal Probability for 2 [0.43350766 0.56674476]

Marginal Probability for 3 [0.01440753 0.32442772 0.36916087 0.2922563 ]

Marginal Probability for 4 [0.7503944 0.2496687]

Marginal Probability for 5 [0.06296638 0.93728604]

Marginal Probability for 6 [0.42374619 0.00619212 0.55708584 0.01348068]

Marginal Probability for 7 [0.06701295 0.93323947]

Marginal Probability for 8 [0.04160159 0.95865083]

Marginal Probability for 9 [0.13255478 0.00917775 0.84142648 0.0169041 ]
```

In [12]:
```
pair_lists = [] #each entry will be a list of all paired probabilities corresponding to the index of the "pair_lists" variable
#ex) pair_lists[0] shows all of the paired probabilities for variable 0 (income) and each other variable 1-9
#X = [income,smoke,cholesterol,bmi,exercise,attack,bp,angina,stroke,diabetes]
no_repeats = []
for p in range(10):
    probs = []
    for n in range(10):
        if (n,p) not in no_repeats and (p,n) not in no_repeats:
            if n != p:
                probs.append(marginals[p]*marginals[n])
                no_repeats.append((n,p))
                no_repeats.append((p,n))
    pair_lists.append(probs)
```

In [13]:
```
print(pair_lists[0]) #a quick example for clarity
print(pair_lists[0][0].table)
```

```
[Factor({0,1},[0x55a03c5afd20]), Factor({0,2},[0x55a03be2cb40]), Factor({0,3},[0x55a03c45ff40]), Factor({0,4},[0x55a03be2f7c0]), Factor
({0,5},[0x55a03c517530]), Factor({0,6},[0x55a03bce9990]), Factor({0,7},[0x55a03c740260]), Factor({0,8},[0x55a03c7407c0]), Factor({0,9},[0
x55a03bc1eb80])]
[[0.02313364 0.02549927]
 [0.0280343  0.03090106]
 [0.03484306 0.03840608]
 [0.04407025 0.04857684]
 [0.05501417 0.06063987]
 [0.07183062 0.07917596]
 [0.07821723 0.08621566]
 [0.1405805  0.15495616]]
```

In [14]:
```
pwmp = []
for pair in pair_lists:
    for prob in pair:
        helpers = prob.vars
        pwmp.append((prob+(np.log(prob/(marginals[helpers[0]]*marginals[helpers[1]])))))
        print("Pairwise marginal probability of",prob.vars,"=","\n",(prob+(np.log(prob/(marginals[helpers[0]]*marginals[helpers[1]])))).tabl
```

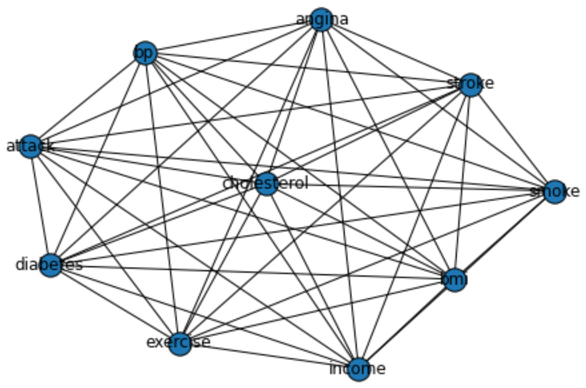
```
Pairwise marginal probability of {0,1} =
 [[0.02313364 0.02549927]
 [0.0280343  0.03090106]
 [0.03484306 0.03840608]
 [0.04407025 0.04857684]
 [0.05501417 0.06063987]
 [0.07183062 0.07917596]
 [0.07821723 0.08621566]
 [0.1405805  0.15495616]]
Pairwise marginal probability of {0,2} =
 [[0.02108141 0.02756071]
 [0.02554732 0.0333992 ]
 [0.03175206 0.04151094]
 [0.04016069 0.05250393]
 [0.05013375 0.06554218]
 [0.06545838 0.08557679]
 [0.07127842 0.0931856 ]
 [0.12810932 0.16748328]]
Pairwise marginal probability of {0,3} =
 [[0.00070064 0.01577687 0.01795224 0.01421238]
 [0.00084906 0.01911906 0.02175526 0.01722314]
 [0.00105527 0.02376255 0.02703901 0.02140617]
 [0.00133473 0.03005538 0.03419952 0.02707498]
 [0.00166618 0.03751901 0.04269225 0.03379849]
 [0.00217549 0.04898763 0.0557422  0.04412984]
 [0.00236892 0.05334322 0.06069836 0.04805352]
 [0.00425768 0.09587423 0.10909368 0.086367  ]]
Pairwise marginal probability of {0,4} =
 [[0.03649156 0.01214135]
 [0.04422197 0.01471339]
 [0.05496227 0.01828686]
 [0.06951746 0.02312962]
 [0.08678067 0.02887337]
 [0.11330734 0.03769924]
 [0.12338174 0.04105116]
 [0.22175506 0.0737816 ]]
Pairwise marginal probability of {0,5} =
 [[0.00306205 0.04558007]
 [0.00371071 0.05523581]
 [0.00461194 0.06865106]
 [0.00583328 0.08683134]
 [0.00728186 0.10839408]
 [0.00950774 0.14152743]
 [0.01035309 0.15411093]
 [0.0186077  0.2769849 ]]
Pairwise marginal probability of {0,6} =
 [[0.02060671 0.00030112 0.027091   0.00065556]
 [0.02497206 0.00036491 0.03282998 0.00079444]
 [0.03103708 0.00045354 0.04080348 0.00098739]
 [0.03925637 0.00057365 0.05160912 0.00124887]
 [0.04900487 0.0007161  0.06442516 0.001559  ]
 [0.06398443 0.00093499 0.08411832 0.00203554]
 [0.06967342 0.00101812 0.09159745 0.00221653]
 [0.12522463 0.00182988 0.1646289  0.00398378]]
Pairwise marginal probability of {0,7} =
 [[0.00325883 0.04538329]
 [0.00394918 0.05499733]
 [0.00490833 0.06835467]
 [0.00620816 0.08645646]
 [0.00774983 0.1079261 ]
 [0.01011876 0.14091641]
 [0.01101844 0.15344558]
 [0.01980353 0.27578907]]
Pairwise marginal probability of {0,8} =
 [[0.00202308 0.04661904]
 [0.00245165 0.05649487]
 [0.00304709 0.07021591]
 [0.00385402 0.0888106 ]
 [0.00481109 0.11086484]
 [0.00628172 0.14475345]
 [0.00684024 0.15762378]
 [0.01229402 0.28329858]]
Pairwise marginal probability of {0,9} =
 [[0.00644612 0.00044631 0.04091844 0.00082204]
 [0.00781167 0.00054086 0.04958664 0.00099619]
 [0.00970891 0.00067222 0.06162987 0.00123813]
 [0.01228004 0.00085024 0.07795079 0.00156602]
 [0.01532953 0.00106138 0.09730823 0.0019549 ]
 [0.02001538 0.00138581 0.12705292 0.00255247]
 [0.02179499 0.00150903 0.13834946 0.00277941]
 [0.03917233 0.00271219 0.24865668 0.00499547]]
Pairwise marginal probability of {1,2} =
 [[0.20622338 0.26960544]
 [0.22731163 0.29717509]]
Pairwise marginal probability of {1,3} =
 [[0.00685379 0.1543331  0.17561305 0.13902888]
 [0.00755465 0.17011509 0.19357111 0.15324587]]
Pairwise marginal probability of {1,4} =
 [[0.35696918 0.11876958]
 [0.39347258 0.13091487]]
Pairwise marginal probability of {1,5} =
 [[0.02995366 0.44587516]
 [0.0330167  0.49147002]]
Pairwise marginal probability of {1,6} =
 [[0.20157977 0.00294565 0.2650106  0.00641288]
 [0.22219317 0.00324687 0.29211039 0.00706865]]
Pairwise marginal probability of {1,7} =
 [[0.03187865 0.44395017]
 [0.03513853 0.48934818]]
Pairwise marginal probability of {1,8} =
 [[0.01979024 0.45603858]
 [0.02181398 0.50267274]]
Pairwise marginal probability of {1,9} =
 [[0.06305747 0.00436594 0.40027393 0.00804143]
 [0.06950568 0.00481239 0.44120564 0.00886374]]
Pairwise marginal probability of {2,3} =
 [[0.00624577 0.1406419  0.16003406 0.12669534]
 [0.00816539 0.18386771 0.20921999 0.16563473]]
Pairwise marginal probability of {2,4} =
 [[0.32530172 0.10823329]
 [0.42528209 0.14149843]]
Pairwise marginal probability of {2,5} =
 [[0.02729641 0.40632068]
 [0.03568587 0.53120195]]
Pairwise marginal probability of {2,6} =
 [[0.18369722 0.00268433 0.24150098 0.00584398]
 [0.24015593 0.00350935 0.31572548 0.00764011]]
Pairwise marginal probability of {2,7} =
 [[0.02905063 0.40456646]
 [0.03797924 0.52890857]]
Pairwise marginal probability of {2,8} =
 [[0.01803461 0.41558248]
 [0.02357748 0.54331033]]
Pairwise marginal probability of {2,9} =
 [[0.05746351 0.00397862 0.36476482 0.00732806]
 [0.07512473 0.00520144 0.47687404 0.00958031]]
Pairwise marginal probability of {3,4} =
 [[0.01081133 0.00359711]
 [0.24344875 0.08099945]
 [0.27701625 0.09216791]
 [0.21930749 0.07296725]]
Pairwise marginal probability of {3,5} =
 [[0.00090719 0.01350397]
 [0.02042804 0.30408157]
 [0.02324472 0.34600933]
 [0.01840232 0.27392775]]
Pairwise marginal probability of {3,6} =
 [[6.10513519e-03 8.92131561e-05 8.02622986e-03 1.94223300e-04]
 [1.37475012e-01 2.00889568e-03 1.80734089e-01 4.37350684e-03]
 [1.56430513e-01 2.28588874e-03 2.05654292e-01 4.97654015e-03]
 [1.23842494e-01 1.80968634e-03 1.62811845e-01 3.93981412e-03]]
Pairwise marginal probability of {3,7} =
 [[0.00096549 0.01344567]
 [0.02174086 0.30276875]
 [0.02473856 0.34451549]
 [0.01958496 0.27274511]]
Pairwise marginal probability of {3,8} =
 [[0.00059938 0.01381179]
 [0.01349671 0.3110129 ]
 [0.01535768 0.35389637]
 [0.01215833 0.28017174]]
Pairwise marginal probability of {3,9} =
 [[1.90978675e-03 1.32228681e-04 1.21228756e-02 2.43546253e-04]
 [4.30044461e-02 2.97751630e-03 2.72982074e-01 5.48415767e-03]
 [4.89340386e-02 3.38806591e-03 3.10621729e-01 6.24033113e-03]
 [3.87399701e-02 2.68225504e-03 2.45912187e-01 4.94032881e-03]]
Pairwise marginal probability of {4,5} =
 [[0.04724962 0.7033342 ]
 [0.01572073 0.23401099]]
Pairwise marginal probability of {4,6} =
 [[0.31797677 0.00464653 0.41803409 0.01011583]
 [0.10579616 0.00154598 0.1390869  0.0033657 ]]
Pairwise marginal probability of {4,7} =
 [[0.05028614 0.70029767]
 [0.01673104 0.23300069]]
Pairwise marginal probability of {4,8} =
 [[0.0312176  0.71936621]
 [0.01038661 0.23934511]]
Pairwise marginal probability of {4,9} =
 [[0.09946837 0.00688693 0.63140172 0.01268474]
 [0.03309478 0.0022914  0.21007786 0.00422042]]
Pairwise marginal probability of {5,6} =
 [[2.66817641e-02 3.89895442e-04 3.50776789e-02 8.48829733e-04]
 [3.97171390e-01 5.80378847e-03 5.22148776e-01 1.26352547e-02]]
Pairwise marginal probability of {5,7} =
 [[0.00421956 0.05876271]
 [0.0628103  0.87471232]]
Pairwise marginal probability of {5,8} =
 [[0.0026195  0.06036277]
 [0.03899259 0.89853003]]
Pairwise marginal probability of {5,9} =
 [[8.34649489e-03 5.77889657e-04 5.29815797e-02 1.06438981e-03]
 [1.24241747e-01 8.60217628e-03 7.88657286e-01 1.58439741e-02]]
Pairwise marginal probability of {6,7} =
 [[2.83964832e-02 3.95456671e-01]
 [4.14952300e-04 5.77873161e-03]
 [3.73319663e-02 5.19894489e-01]
 [9.03380269e-04 1.25807041e-02]]
Pairwise marginal probability of {6,8} =
 [[1.76285153e-02 4.06224639e-01]
 [2.57602074e-04 5.93608184e-03]
 [2.31756564e-02 5.34050799e-01]
 [5.60817788e-04 1.29232666e-02]]
Pairwise marginal probability of {6,9} =
 [[5.61695845e-02 3.88903634e-03 3.56551266e-01 7.16304676e-03]
 [8.20795241e-04 5.68297335e-05 5.21021448e-03 1.04672213e-04]
 [7.38443920e-02 5.11279416e-03 4.68746773e-01 9.41703303e-03]
 [1.78692883e-03 1.23722317e-04 1.13430025e-02 2.27878750e-04]]
Pairwise marginal probability of {7,8} =
 [[0.00278785 0.06424202]
 [0.03882425 0.89465078]]
Pairwise marginal probability of {7,9} =
 [[8.88288723e-03 6.15028072e-04 5.63864717e-02 1.13279344e-03]
 [1.23705354e-01 8.56503786e-03 7.85252394e-01 1.57755704e-02]]
Pairwise marginal probability of {8,9} =
 [[5.51448968e-03 3.81808962e-04 3.50046790e-02 7.03237308e-04]
 [1.27073752e-01 8.79825697e-03 8.06634186e-01 1.62051266e-02]]
```

In [15]:

```
model = gm.GraphModel(pwmp)
```

In [16]:
```
gm.drawMarkovGraph(model,var_labels = {0:"income",1:"smoke",2:"cholesterol",3:"bmi",4:"exercise",5:"attack",6:"bp",7:"angina",8:"stroke",9:"
```

Out[16]:
```
<networkx.classes.graph.Graph at 0x7f28c0a59b80>
```



In [17]:
```
aLLTp = 0
for case in train:
    for x in range(10):
        #for each case from the dataset, calculate the log probability of every combination from the pairwise distributions
        for pair in pair_lists[x]:
            helper = pair.vars
            aLLTp += np.log(pair[case[helper[0]],case[helper[1]]])



aLLTp /= len(train)*10 #times 10 to account for the fact that there are 10 values to add together per the number of cases
print("average log-likelihood (training data):",aLLTp)
```

```
average log-likelihood (training data): -6.267275781652035
```

In [18]:
```
aLLVp = 0
for case in valid:
    for x in range(10):
        for pair in pair_lists[x]:
            helper = pair.vars
            aLLVp += np.log(pair[case[helper[0]],case[helper[1]]])



aLLVp /= len(valid)*10 #times 10 to account for the fact that there are 10 values to add together per the number of cases
print("average log-likelihood (validation data):",aLLVp)
```

```
average log-likelihood (validation data): -6.307552163535195
```

In [19]:
```
#finding free probabilities for the pairwise distributions
total = 0
for x in range(10):
    for z in pair_lists[x]:
        temp = 1
        for num in z.dims():
            temp *= num
        total += temp
print("Total probabilities for newtwork:",total)
```

```
Total probabilities for newtwork: 444
```

In [20]:
```
#504
#444
aLLT -= (504/2)*(np.log(len(train))/len(train))
print("BIC penalty for given tree:" ,aLLT)
aLLTp -= (444/2)*(np.log(len(train))/len(train))
print("BIC penalty for pairwise weighted tree:",aLLTp)
#the given tree from the homework is prefered
```

```
BIC penalty for given tree: -2.8286846771941874
BIC penalty for pairwise weighted tree: -6.2781708171962745
```