**Student’s name: Nguyen Van Ty (M5231130)**

**Report of Machine Learning Course #5**

**BAYESIAN NETWORK**

**Q1. For the example given in P. 23, write the equation for finding the joint probability based on Eq. (2)**

**Q2. Based on the equation, write a program to build a “joint probability distribution table” (JPDT) based on your equation**

**Code:**

|  |
| --- |
| import numpy as np  P\_x1 = [0.2, 0.8]  P\_x2 = [[0.8, 0.2], [0.2, 0.8]]  P\_x3 = [[0.8, 0.2], [0.2, 0.8]]  P\_x4 = [[[0.8, 0.2], [0.4, 0.6]], [[0.4, 0.6], [0.2, 0.8]]]  P\_x5 = [[0.8, 0.2], [0.2, 0.8]]  # Build Table  print('Joint Probability Distribution Table (JPDT)')  table = dict()  for a in [0, 1]:  for b in [0, 1]:  for c in [0, 1]:  for d in [0, 1]:  for e in [0, 1]:  table[(a, b, c, d, e)] = P\_x1[a] \* P\_x2[a][b] \* P\_x3[a][c] \* P\_x4[b][c][d] \* P\_x5[c][e]  print('P(x1, x2, x3, x4, x5) = P(x1).P(x2|x1).P(x3|x1).P(x4|x2, x3).P(x5|x3)')  for k, v in table.items(): print(f'P{k} = {v:>6.2%}') |

**Output:**

|  |  |
| --- | --- |
| **Joint Probability Distribution Table (JPDT)**  P(x1, x2, x3, x4, x5) = P(x1).P(x2|x1).P(x3|x1).P(x4|x2, x3).P(x5|x3)  P(0, 0, 0, 0, 0) = 8.19%  P(0, 0, 0, 0, 1) = 2.05%  P(0, 0, 0, 1, 0) = 2.05%  P(0, 0, 0, 1, 1) = 0.51%  P(0, 0, 1, 0, 0) = 0.26%  P(0, 0, 1, 0, 1) = 1.02%  P(0, 0, 1, 1, 0) = 0.38%  P(0, 0, 1, 1, 1) = 1.54%  P(0, 1, 0, 0, 0) = 1.02%  P(0, 1, 0, 0, 1) = 0.26%  P(0, 1, 0, 1, 0) = 1.54%  P(0, 1, 0, 1, 1) = 0.38%  P(0, 1, 1, 0, 0) = 0.03%  P(0, 1, 1, 0, 1) = 0.13%  P(0, 1, 1, 1, 0) = 0.13% | P(0, 1, 1, 1, 1) = 0.51%  P(1, 0, 0, 0, 0) = 2.05%  P(1, 0, 0, 0, 1) = 0.51%  P(1, 0, 0, 1, 0) = 0.51%  P(1, 0, 0, 1, 1) = 0.13%  P(1, 0, 1, 0, 0) = 1.02%  P(1, 0, 1, 0, 1) = 4.10%  P(1, 0, 1, 1, 0) = 1.54%  P(1, 0, 1, 1, 1) = 6.14%  P(1, 1, 0, 0, 0) = 4.10%  P(1, 1, 0, 0, 1) = 1.02%  P(1, 1, 0, 1, 0) = 6.14%  P(1, 1, 0, 1, 1) = 1.54%  P(1, 1, 1, 0, 0) = 2.05%  P(1, 1, 1, 0, 1) = 8.19%  P(1, 1, 1, 1, 0) = 8.19%  P(1, 1, 1, 1, 1) = 32.77% |

**Q3. Based on the JPDT, write a program to generate 100 data**

**Code:**

|  |
| --- |
| # Random Generation  print('\nData Generation:')  n = 100  ks = [k for k in table.keys()]  vs = [v for v in table.values()]  ns = np.random.choice(len(ks), n, p=vs)  for n in ns: print(\*ks[n]) |

**Output:**

Data Generation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 1 1 0 1  1 0 0 0 1  1 0 1 1 1  0 1 0 1 0  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  1 1 0 0 0  1 1 1 1 0  1 1 1 1 1  1 1 0 0 0  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  0 0 0 0 0  1 1 1 0 1  0 0 0 0 0  1 1 0 0 0  1 1 1 1 1  1 0 1 1 1 | 1 1 1 1 0  0 0 0 0 0  0 0 0 0 0  1 0 0 0 0  1 1 1 1 1  0 0 0 0 0  0 0 0 0 0  1 1 1 1 1  1 1 1 1 0  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  0 0 0 0 0  1 0 0 0 0  1 1 1 1 1  0 0 1 0 1  0 0 0 0 0  1 1 1 0 1  1 1 1 1 1 | 1 1 1 1 1  1 1 1 1 1  1 1 0 0 0  1 1 0 1 0  0 0 0 1 0  1 1 0 0 0  0 0 0 0 1  1 0 0 0 0  0 0 1 1 1  1 1 1 0 1  0 1 0 1 0  0 0 0 0 0  0 1 0 1 0  1 1 1 1 1  1 1 0 0 1  1 1 1 1 1  1 1 0 0 1  1 0 0 1 1  1 0 1 1 0  1 1 1 1 1 | 1 1 1 1 0  1 1 0 1 1  1 1 0 0 0  1 0 1 1 1  0 0 0 0 0  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  1 1 0 1 0  1 1 1 0 0  1 0 1 0 1  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  1 1 1 1 0  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  1 1 1 0 1 | 1 0 0 0 0  0 0 0 1 0  1 1 1 1 0  1 1 1 1 0  1 1 1 1 1  1 1 1 0 1  1 1 1 1 1  1 1 1 1 1  1 1 1 1 1  1 0 0 0 1  1 1 1 1 1  1 1 1 0 1  1 1 1 0 1  1 1 0 1 0  1 1 1 1 1  1 1 1 1 0  1 1 1 1 1  1 1 1 1 0  1 1 0 0 0  0 1 0 1 0 |