Loan Approval Decision Support System with Bayesian Network

1. Introduction

The paper is organized as follows. In Section 2, a conceptual design, containing the task, objectives, constraints, potential users, knowledge, and improvements, is provided. In Section 3, background knowledge is presented as a Bayesian Network; specifically, a description of the background knowledge, nodes, links, and a description of the associated probabilities are given. In Section 4 paper conclusions are drawn. Finally, in the Appendix the Bayesian Network is implemented in Python and three questions for the decision support system are answered.

2. Conceptual Design

Task: The task of this paper is to provide a loan approval decision support system capable of analysing the ability of a client to pay off a loan. It uses a Bayesian network to create the decision support system. The system takes into consideration some of the risk factors and their influence.

Objectives: This system has several objectives. The first objective is to integrate traditional loan granting systems by automatically predicting, without human intervention, if a client is going to extinguish his debt. The second objective is to represent the background knowledge as a Bayesian Network. Last, implement the Bayesian Network in Python and test it by answering three questions.

The questions that are going to be answered are:

1. What is the probability of getting a loan request approved?
2. What is the probability of getting a loan request approved if the client has a high value coefficient?
3. What is the probability for each credit score if a loan gets approved for a client with a low value coefficient?

Constraints: The main constraint of this system would be its simplicity. For the decision-making process to be accurate, it needs to have tens of variables (nodes) to represent the knowledge needed for the loan underwriting process. Another constraint, which is present in all automated systems is that they are less effective when they have to deal with non-standard loans, since the process of underwriting non-standard loans is personalized [1].

Potential Users: The loan approval decision support system aims at helping financial experts in analysing loan applications and deciding if a request is going to be approved or not based on the ability of the client to repay the loan. Moreover, it can be useful to clients who can check on their own if they are eligible for a loan application.

This system can be a **stand-alone** system. Anyway, it would be more effective if integrated into a combined system which takes into consideration human interaction too, especially in non-standard cases. Also, it can be integrated into the financial institution's website to allow clients to check online if they can get a loan based on the input provided.

Data and Knowledge: This system requires several background knowledge models that specify risk factors that influence the ability to repay a debt. Moreover, it would need estimates of approval depending on combinations of risk factors. These estimates can be calculated from previous financial institution's historical data or public institution reports. These risk factors serve as inputs for helping this system to produce a result.

Improvements: An extension of the data collection about loan payments or introducing other nodes could improve the current system. Furthermore, by integrating with other systems, the prediction could be further improved. Another important improvement would be creating a real time system that could offer a live prediction.

3. Bayesian Network

3.1 Textual description of the background knowledge

Automated underwriting systems are an alternative to the old manual underwriting process. They are time effective and more accurate for standard loans [2]. This application is a simplified version of an automated loan underwriting system based on Bayesian Network. It will take into consideration several variables such as: client's income, assets, credit score, and loan amount. In a second moment, income and assets will contribute, leading to a new node: value coefficient, which together with loan amount and credit score will affect the final result or the approve node. Income, assets, and loan amount nodes are divided into three categories: high, medium, and low. Credit score has two categories good and bad; meanwhile, approve node has yes and no values.

Analysing the income, our dataset is composed by 20% of people with high income, 50% with medium income and 30% with low income. If we consider the assets, 10% of the population possesses large assets, 30% medium assets and 60% has small or no assets.

In influencing the people's value coefficient variable, income plays a slightly major role compared to assets. Almost all people with high income and large assets are assigned a high value coefficient. People with high income and medium assets occasionally are assigned a medium value, while people with high income and low assets sometimes are assigned a medium or low value. On the other hand, people with medium income are usually assigned a medium value when they have medium assets. In cases when they have high or low assets, sometimes they might be considered as people with high value or low value, respectively. In the meantime, people with low income are usually assigned a low value and occasionally medium value, although only when they possess large assets. Quite always they have a low value coefficient when they own medium or small assets.

If we analyse the credit score node, we can say that from our population, 70% of the people have a good credit score while 30% have a bad credit score. On the meantime, 30% of the same population request a low amount of loan; 50% request a medium amount of loan, and 20% request a high amount of loan.

The approve node depends on 3 factors: value, loan amount, and credit score. We can see that value coefficient has a major impact on approvals followed by credit score and loan amount. People are likely to get a loan if they have a high value coefficient and a good credit score, although it gets more difficult to obtain a high amount of loan. Most of the people with medium value can get a small loan only if they have a good credit score. When they request a medium amount, just half of them receive it, of course, only if they have a good credit score. In other cases, not mentioned above, except from cases when people with low value coefficient and good credit score who request a small amount of loan occasionally obtain the loan, quite all applicants are not granted the loan, especially when they have low values.

3.2 Selection of the nodes and links

Before analysing the parents-child relationships between the variables, we need to specify that this network is composed of 6 nodes and 5 links. The value node is dependent on income and assets nodes which are considered as parents. Meanwhile, the approve child node depends on value, loan, and credit score parent nodes. Value node is linked with 3 nodes, same as approve node.

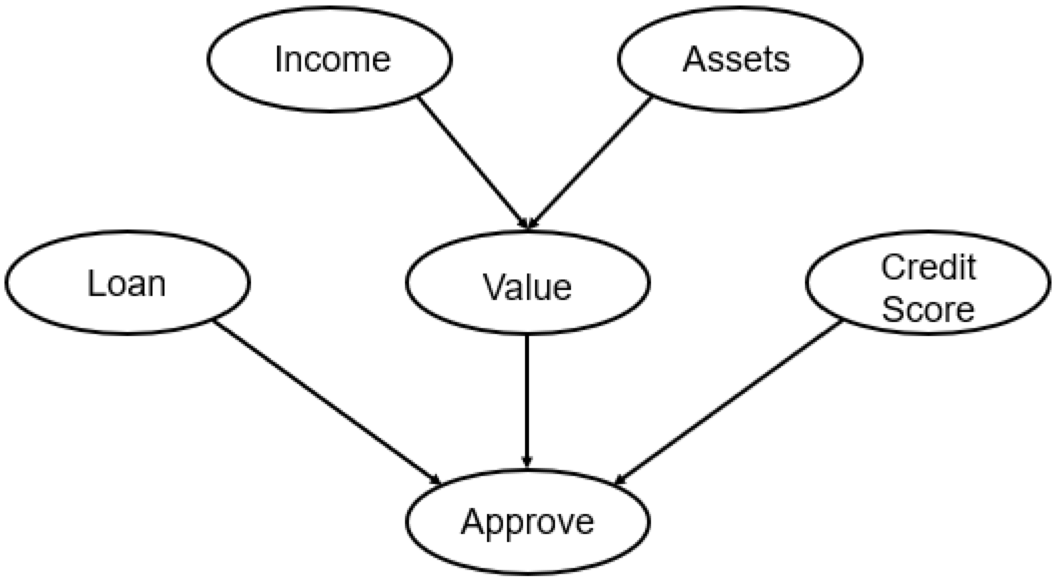

Fig 1. Graphical representation of the Bayesian Network

3.3 Associated probabilities

The conditional probability tables of any single node are assigned by relying on the subjective approach, meaning that the probabilities are assigned based on the degree of belief that the author holds in the occurrence of the event. [3] For the construction of the tables, it is not going to be used the full name of the variables or the values. Instead, they are going to be expressed as symbols and numbers:

- Income **(i)** – takes value 0 for high, 1 for medium and 2 for low
- Assets **(a)** – takes value 0 for large, 1 for medium and 2 for small-no assets
- Value **(v)** – takes value 0 for high, 1 for medium and 2 for small
- Loan **(l)** – takes value 0 for small, 1 for medium and 2 for large
- Credit Score **(c)** – takes value 0 for good, 1 for bad
- Approve **(ap)** – takes value 0 for yes and 1 for no.

The probability distributions for all the variables are given below:

Income

i0	i1	i2
0.2	0.5	0.3

Loan

l0	l1	l2
0.3	0.5	0.2

Assets

a0	a1	a2
0.1	0.3	0.6

Approve

	ap0	ap1
v0, l0, c0	0.99	0.01
v0, l0, c1	0.6	0.4
v0, l1, c0	0.9	0.1
v0, l1, c1	0.5	0.5
v0, l2, c0	0.8	0.2
v0, l2, c1	0.3	0.7
v1, l0, c0	0.7	0.3
v1, l0, c1	0.4	0.6
v1, l1, c0	0.5	0.5
v1, l1, c1	0.2	0.8
v1, l2, c0	0.2	0.8
v1, l2, c1	0.1	0.9
v2, l0, c0	0.3	0.7
v2, l0, c1	0.01	0.99
v2, l1, c0	0.1	0.9
v2, l1, c1	0.01	1
v2, l2, c0	0.05	0.95
v2, l2, c1	0	1

Credit Score

c0	c1
0.7	0.3

Value

	v0	v1	v2
i0, a0	0.9	0.1	0
i0, a1	0.7	0.3	0
i0, a1	0.5	0.4	0.1
i1, a0	0.4	0.5	0.1
i1, a1	0.4	0.7	0.2
i1, a2	0.05	0.5	0.45
i2, a0	0.05	0.35	0.6
i2, a1	0	0.2	0.8
i2, a2	0	0.1	0.9

Fig 2. Probability distributions for each variable

4. Conclusions

In this paper, an explanation of a loan approval decision support system was provided. Furthermore, its background knowledge was represented as a Bayesian Network and it was implemented in Python. This model was able to estimate several probabilities and answer the questions asked in Section 2.

For the first question, the probability of approved loans was P(ap) = 0.34, while the probability of getting a loan request approved for a client with a high value coefficient was P (ap | ho) = 0.78. To the question of what the probability for each credit score is if a loan gets approved for a client with a low value coefficient, the model provided the answer P (c | ap0, v2) = 0.98. These results show that for our population in general, it is difficult to get a loan, although it is easier for people with a high value coefficient. On the other hand, a good credit score proves its importance, especially in cases with a low value coefficient. This model provides an effective method for analysing loan requests and proves to be an interesting alternative to the old manual loan approval process.

Total number of words: 1535

References

[1] Connelly, Hugh W. 'Automated Risk Management for Small Business Lending', The RMA Journal, vol. 98/no. 3, (2015), pp. 36.

[2] Krovvidy, Srinivas, Robin Landsman, Steve Opdahl, et al. 'Custom DU--a Web-Based Business User-Driven Automated Underwriting System', The AI Magazine, vol. 29/no. 1, (2008), pp. 41.

[3] Personal.utdallas.edu. 2022. [online] Available at: <https://personal.utdallas.edu/~scniu/OPRE-6301/documents/Probability.pdf> [Accessed 5 January 2022]

Appendix

Loan Approval Decision Support System with Bayesian Network

1. Bayesian Network Implementation

In [1]:
```
#import libraries
import numpy as np
from pgmpy.factors.discrete import TabularCPD
from pgmpy.models import BayesianNetwork
```

First, we define the network structure

In [2]:
```
document_model = BayesianNetwork([('Income', 'Value'),
                                  ('Assets', 'Value'),
                                  ('Loan', 'Approve'),
                                  ('Credit_Score', 'Approve'),
                                  ('Value', 'Approve')])
```

Then we define the conditional probability distribution for each node.

In [3]:
```
income_cpd = TabularCPD(
    variable = 'Income',
    variable_card = 3, #high income,medium income, low income
    values = [[.2],[.5],[.3]])
```

In [4]:
```
assets_cpd = TabularCPD(
    variable = 'Assets',
    variable_card = 3, #Large, medium, small
    values = [[.1],[.3],[.6]])
```

In [5]:
```
value_cpd = TabularCPD(
    variable = 'Value',
    variable_card = 3, # high, medium, low
    values = [[.9,.7,.5,.4,.1,.05,.05,0,0],
              [.1,.3,.4,.5,.7,.5,.35,.2,.1],
             [0,0,.1,.1,.2,.45,.6,.8,.9]],
    evidence = ['Income','Assets'],
    evidence_card = [3,3])
```

In [6]:
```
loan_cpd = TabularCPD(
    variable = 'Loan',
    variable_card = 3, #low ammount, medium ammount, high ammount
    values = [[.3],[.5],[.2]])
```

In [7]:
```
cs_cpd = TabularCPD(
    variable = 'Credit_Score',
    variable_card = 2, #good, bad
    values = [[.7],[.3]])
```

In [8]:
```
approve_cpd = TabularCPD(
    variable = 'Approve',
    variable_card = 2, # yes,no
    values = [[.99,.6,.9,.5,.8,.3,.7,.4,.5,.2,.2,.1,.3,.01,.1,.005,.05,0],
             [.01,.4,.1,.5,.2,.7,.3,.6,.5,.8,.8,.9,.7,.99,.9,.995,.95,1]],
    evidence = ['Value','Loan','Credit_Score'],
    evidence_card = [3,3,2])
```

Then, we add all conditional probabilities to our model.

In [9]:
```
document_model.add_cpds(income_cpd,assets_cpd,value_cpd,loan_cpd,cs_cpd,approve_cpd)
```

In [10]:
```
document_model.get_cpds()
```

Out[10]:
```
[<TabularCPD representing P(Income:3) at 0x15a75f59520>,
 <TabularCPD representing P(Assets:3) at 0x15a75f440d0>,
 <TabularCPD representing P(Value:3 | Income:3, Assets:3) at 0x15a7605ce80>,
 <TabularCPD representing P(Loan:3) at 0x15a7605cf40>,
 <TabularCPD representing P(Credit_Score:2) at 0x15a7605cc70>,
 <TabularCPD representing P(Approve:2 | Value:3, Loan:3, Credit_Score:2) at 0x15a760780d0>]
```

In [11]:
```
document_model.get_independencies()
```

```
(Credit_Score ⊥ Income, Assets | Approve, Value)
(Credit_Score ⊥ Assets, Loan | Value, Income)
(Credit_Score ⊥ Income, Assets | Value, Loan)
(Credit_Score ⊥ Value, Assets | Income, Loan)
(Credit_Score ⊥ Income | Approve, Value, Assets)
(Credit_Score ⊥ Loan | Value, Income, Assets)
(Credit_Score ⊥ Income | Value, Assets, Loan)
(Credit_Score ⊥ Value | Income, Assets, Loan)
(Credit_Score ⊥ Assets | Approve, Value, Income)
(Credit_Score ⊥ Income, Assets | Approve, Value, Loan)
(Credit_Score ⊥ Assets | Value, Income, Loan)
(Credit_Score ⊥ Income | Approve, Value, Assets, Loan)
(Credit_Score ⊥ Assets | Approve, Value, Income, Loan)

(Assets ⊥ Credit_Score, Income, Loan)
(Assets ⊥ Income, Loan | Credit_Score)
(Assets ⊥ Approve, Credit_Score, Loan | Value)
(Assets ⊥ Credit_Score, Loan | Income)
(Assets ⊥ Credit_Score, Income | Loan)
(Assets ⊥ Approve, Loan | Value, Credit_Score)
(Assets ⊥ Loan | Credit_Score, Income)
```

2. Questions

```
In [12]: from pgmpy.inference import VariableElimination
```

```
In [13]: approve_infer=VariableElimination(document_model)
```

What is the probability of getting a loan request approved?

```
In [17]: # approved loans
prob_loan = approve_infer.query(variables=['Approve'],joint=False)
print(prob_loan['Approve'])
```

```
Finding Elimination Order: : 100%                                   5/5 [00:00<00:00, 62.65it/s]


Eliminating: Loan: 100%                                     5/5 [00:00<00:00, 63.36it/s]


+------------+----------------+
| Approve    |   phi(Approve) |
+============+================+
| Approve(0) |         0.3469 |
+------------+----------------+
| Approve(1) |         0.6531 |
+------------+----------------+
```

What is the probability of getting a loan request approved if the client has a high value coefficient?

```
In [15]: # probability to get a loan with a high value coeff
prob_loan_hvalue = approve_infer.query(variables = ['Approve'],joint = False,
                                      evidence = ({'Value' : 0}))
print(prob_loan_hvalue['Approve'])
```

```
Finding Elimination Order: : 100%                                   2/2 [00:00<00:00, 20.26it/s]

Eliminating: Loan: 100%                                     2/2 [00:00<00:00, 27.46it/s]


+------------+----------------+
| Approve    |   phi(Approve) |
+============+================+
| Approve(0) |         0.7819 |
+------------+----------------+
| Approve(1) |         0.2181 |
+------------+----------------+
```

What is the probability for each credit score if a loan gets approved for a client with a low value coefficient?

```
In [18]: # credit score of people who received a loan with a low value coeff
prob_cs_hvalue_loan = approve_infer.query(variables = ['Credit_Score'],joint = False,
                                      evidence = ({'Approve' : 0,'Value' : 2}))
print(prob_cs_hvalue_loan['Credit_Score'])
```

```
Finding Elimination Order: : 100%                                   1/1 [00:00<00:00, 14.65it/s]


Eliminating: Loan: 100%                                     1/1 [00:00<00:00, 15.83it/s]


+-----------------+---------------------+
| Credit_Score    |   phi(Credit_Score) |
+=================+=====================+
| Credit_Score(0) |              0.9845 |
+-----------------+---------------------+
| Credit_Score(1) |              0.0155 |
+-----------------+---------------------+
```