Tarea 2. Redes Bayesianas

Inteligencia Artificial

1 Introducción

El propósito de esta tarea es que desarrolle familiaridad con las redes bayesianas y entienda como calcular probabilidades en consultas de estas redes. Esta tarea tiene dos partes. Primero usará el paquete SAMIAM para diseñar una red bayesiana pequeña para calcular calidad de crédito. EN la segunda parte, la funcionalidad que implementa en el paquete SAMIAM la debe reproducir en el lenguaje de programación octave o matlab.

2 Creando una red para evaluar calidad de crédito

En la primera parte de la tarea se utilizará SAMIAM para crear una red bayesiana que evalúa la calidad de crédito de un cliente.

2.1 Utilizando SAMIAM

SAMIAM fue desarrollado por el grupo de razonamiento automatizado de universidad de California en Los Angeles UCLA para proveer un frente gráfico para manipular redes probabilísticas. Tiene una extensa funcionalidad para aprender y hacer inferencia en redes probabilisticas, sin embargo, nosotros solo lo usaremos para manipular una red y ver probabilidades marginales. El programa lo puede descargar del siguiente link:

`http://reasoning.cs.ucla.edu/samiam/index.php`

La red que utilizaremos en nuestra tarea es para evaluar la calidad de crédito de un cliente. La red ya trae los nodos necesarios y se encuentra en el archivo `Credit_net.net`.

Para construir su red, debe utilizar el *Edit Mode* para agregar aristas a su red. *Solo debe agregar aristas y poner los valores correspondientes para las probabilidades.* Las dems propiedades de los nodos (etiquetas y nombres), no las debe modificar.

2.2 Construyendo la Red

Un amigo suyo en el banco a sabiendas que esta llevando el curso de IA le solicita que le ayude a hacer una red bayesiana para evaluar la calidad de crédito de los clientes. Su amigo le dice que el banco esta en capacidad de observar el *ingreso* (income), la cantidad de *bienes* (assets), la *razón de deudas sobre ingresos* (ratio of debts to income), el *historial de pago* (payment history) así como la *edad* (age) de cada cliente. También su amigo considera que la *calidad de crédito* (credit worthiness) depende eventualmente en que tan *confiable* (reliable) es la persona, cuales son las expectativas de *ingreso futuro* (future income), así como la *razón de deudas a ingreso* (ratio of debts to income). De esta forma usted crea una red bayesiana con las 8 variables relevantes mencionadas anteriormente.

De su experiencia, su amigo le indica que:

1. Cuanto mejor es el historial de pago de una persona, mas probable es que esta sea confiable.

2. Cuanto mayor en edad sea una persona, mas probable es que sea confiable.

3. Las personas mayores tiene una probabilidad mas alta de tener un historial de pago excelente.

4. Personas con una razón alta de deudas sobre ingreso probablemente están en aprietos financieros y por lo tanto es menos probable que tengan un buen historial de pago.

5. Cuanto más alto es el ingreso de una persona, mas sube su probabilidad de tener bienes.

6. Cuanto más bienes e ingresos tiene una persona, mas probable es que tenga un ingreso futuro promisorio.

7. Manteniendo todo lo demás igual, personas confiables tienen una probabilidad mas alta de ser buenos sujetos de crédito. De la misma manera, personas con un futuro ingreso promisorio, o que tienen una razón baja de deudas a ingresos, también tienen una probabilidad mas alta de ser buenos sujetos de crédito.

Agregue en su red bayesiana las aristas necesarias para codificar estas relaciones, agregue también el valor de las probabilidades. Su res será revisada para ver si las aristas están correctas y para ver si *las probabilidades que pone producen marginales que son consistentes con el comportamiento específicado en los 7 puntos anteriores*. Por ejemplo, si C denota la variable aleatoria de confiabilidad y H es el historial de pago su red debería respetar la desigualdad:

$$\begin{aligned} P(C = Confiable \mid H = Excelente) &> P(C = Confiable \mid H = Aceptable) \\ P(C = Confiable \mid H = Aceptable) &> P(C = Confiable \mid H = Inaceptable) \end{aligned}$$

3 Calculando probabilidades en consultas a redes Bayesianas

Inspirado por el SAMIAM, ud. decide replicar algo de su funcionalidad para hacer consultas de probabilidades. En esta parte de la tarea ud. va escribir código para computar primero la distribución conjunta y luego las marginales.

3.1 Operaciones básicas sobre factores

Para calcular las probabilidades en la red bayesiana, primero se han de implementar tres funciones

- `FactorProduct.m` (10 puntos) - Esta función produce la multiplicación de dos factores.
- `FactorMarginalization.m` (10 puntos) - Esta función es la suma sobre algunas variables, y devolver el factor resultante.
- `ObserveEvidence.m` (10 puntos) - Esta funcin debe modificar un conjunto de factores dada la evidencia observada de algunas de las variables, así que los valores de las variables que no son consistentes con

los valores observados se deben poner en 0 (el factor no se debe renormalizar).

Vamos a utilizar Julia para implementar las estructuras de datos de los factores. Un tutorial de como utilizar los factores se encuentra en el archivo `FactorTutorial.jl`. Además, `FactorTutorial.jl` contiene un conjunto de pruebas con los valores esperados que debe producir su código.

3.2 Calculando la distribución conjunta

Con estas funciones ahora puede completar la función `ComputeJointDistribution.jl`, como recordatorio para calcular la distribución conjunta en una red$G$ sobre variables $X_1, \ldots, X_n$ y distribución $P(X_1, \ldots, X_n)$ se utiliza la fórmula:

$$P(X_1, \ldots, X_n) = \prod_i P(X_i \mid Padres_G(X_i))$$

- `ComputeJointDistribution.m` (10 puntos) - Esta funcin debe devolver el factor que representa la distribución conjunta de un conjunto de factores que define la red Bayesiana. puede suponer que sólo le darán factores válidos (no requiere hacer validación de la entrada).

3.3 Calculando Marginales

Después de calcular al distribución conjunta podemos calcular las marginales en un conjunto de variables en la red marginalizando las variables irrelevantes de la distribución conjunta. Sin embargo, este procedimiento le hace falta una etapa cuando observamos la evidencia: si tenemos evidencia, primero necesitamos reducir el factor representando la distribución conjunta por la evidencia antes de marginalizar las variables irrelevantes. Además, en este caso, el factor que obtenemos ahora no esta renormalizado, tenga el cuajado de normalizar despu es de llamar a la función `ObserveEvidence`. Complete `ComputeMarginal.jl` con su implementación.

- `ComputeMarginal.m` (20 puntos) - Esta función calcula la distribución marginal sobre un conjunto de variables, y evidencia opcional.