Tarea 4 Inteligencia Artifical.

MORENO LOPEZ JOSE RODRIGO

October 14, 2024

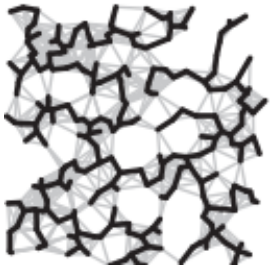

Figure 1: Grafo con búsqueda tipo DFS para un grafo enorme, imagen obtenida del libro recomendado para el curso.

Contents

1 Introducción y como correr el código entregado.

Este trabajo resuelve el siguiente checklist:

1. Desarrollo de Modelo de redes bayesianas.
2. Desarrollo del Algoritmo de D-separación.
3. Desarrollo del algoritmo de inferencia entre: Exhaustiva.
4. Problema de la alarma y contestar P(Burglary — JohnCalls=true, MaryCalls=true) y otras 2 consultas.
5. Un problema mediano de repositorio en internet y hacer 3 consultas.
   (a) Para este ejercicio se resolvió solamente 1. 2. de este listado y se intento ejecutar una búsqueda exhaustiva pero no se logro por un problema de optimización de memoria. Es lo único que faltó.
6. Construcción de red bayesiana desde datos, esto se hizo para poder resolver el problema 5, para ello se desarrolló un constructor de redes bayesianas con base en un input de tipo *.bif.
7. Se resolvieron los 5 ejercicios del libro.

Para poder ejecutar el codigo solamente se tiene que navegar por el archivo .zip al home y ejecutar el archivo tests.py, con el comando python, la version recomendada de python es: 3.10.5.

Part I
Desarrollo.

2 Modelo de redes bayesianas.

El lector puede revisar la clase: **BayesianNetwork** en la cual se implementa un DAG que modela una red bayesiana, donde hay variables, dependencias y probabilidades. Se implementaron solamente unos metodos como es el agregado de nodos para cada lista de tipos de variables que despues se utilizan por el constructor de redes bayesianas, clase: **ConstructorDeDAG** con la cual se construyen RedesBayesianas usando datos. De esta manera, el modelo se ejemplifica cuando se ejecuta el archivo de tests:

```
    A continuacion la red modelada y tambien una revision de d-separacion

Variables:
Burglar: ['True', 'False']
Earthquake: ['True', 'False']
Alarm: ['True', 'False']
JohnCalls: ['True', 'False']
MaryCalls: ['True', 'False']

Dependencias:
Burglar: []
Earthquake: []
Alarm: ['Burglar', 'Earthquake']
JohnCalls: ['Alarm']
MaryCalls: ['Alarm']

Probabilidades:
Burglar: {(): [0.001, 0.999]}
Earthquake: {(): [0.002, 0.998]}
Alarm: {('True', 'True'): [0.95, 0.05], ('True', 'False'): [0.94, 0.06],
   ('False', 'True'): [0.29, 0.71], ('False', 'False'): [0.001, 0.999]}
JohnCalls: {('True',): [0.9, 0.1], ('False',): [0.05, 0.95]}
MaryCalls: {('True',): [0.7, 0.3], ('False',): [0.01, 0.95]}
```

Esto representa nuestra red bayesiana, notese que la otra red bayesiana es muy grande y por practicidad no se pone aqui pero el lector puede ejecutar el codigo y revisar la red bayesiana del problema de Barley que fue el elegido.

3 Algoritmo D-Separacion.

Para el algoritmo de D-Separacion se utiliza el mismo algoritmo del libro para el curso de modo que el algoritmo de búsqueda utiliza una búsqueda de profundidad con el objetivo de encontrar los caminos bloqueadores dadas las observaciones, de modo que, el algoritmo implementado se puede encontrar en el siguiente metodo: **isdseparated**, dicho metodo contiene toda la logica para calcular si dos nodos son dependencialmente separados o no. Este da como output un booleano, se rectifica el siguiente resultado:

```
    False
```

Si el input es:

```
    result = redBayesiana.is_d_separated("Alarm", "Burglar", [])
```

Y si el input es:

```
     result = redBayesiana.is_d_separated("nedbarea", "komm", [])
```

en ambos casos es falso pues no estan separados ambos, el lector puede testear sencillamente con mas nodos del DAG.

4 Inferencia exhaustiva.

Finalmente la implementación del algoritmo de inferencia exhaustiva se encuentra en el método **inference** el cual básicamente hace una búsqueda exhaustiva como el nombre indica sobre todas las posibles combinaciones y es por eso que el algoritmo es ineficiente, sin embargo para redes relativamente pequeñas como el caso de las redes bayesianas del caso del robo, el algoritmo encuentra soluciones consistentes como se puede observar en el siguiente listado:

```
    {'Burglar': {'True': 0.9395272127542605, 'False': 0.060472787245739415}}
{'Burglar': {'True': 0.9395272127542605, 'False': 0.06047278724573941}}
{'Alarm': {'True': 0.9678284182305631, 'False': 0.03217158176943699}}
```

Donde se hacen las siguientes búsquedas:

```
    # Realizar una consulta: P(Burglar | JohnCalls=true, MaryCalls=true)

        print('\n A continuacion se hacen los siguiente queries \n')

        query = "Burglar"
        evidence = {"JohnCalls": "True", "MaryCalls": "True"}

        # Obtener inferencia
        result = redBayesiana.inference(query, evidence)
        print(result)

        query = "Burglar"
        evidence = {"JohnCalls": "True"}

        # Obtener inferencia
        result = redBayesiana.inference(query, evidence)
        print(result)

        query = "Alarm"
        evidence = {"MaryCalls": "True"}

        # Obtener inferencia
        result = redBayesiana.inference(query, evidence)
        print(result)
```

El lector podría dejar ejecutando la query de inferencia exhaustiva en el problema mediano del repositorio pero no se recomienda pues no es óptimo en memoria.

Part II
Implementación y probar.

5 Construcción de red bayesiana desde datos.

Este algoritmo básicamente se encuentra implementado en la clase: **ConstructorDeDAG** donde básicamente se explican en el código en comentarios el paso a paso de como construir una red bayesiana usando solamente un archivo .bis. Se construye y se verifica en el código implementado su uso.
Todo fue implementado de cero y se omitió el usar otras paqueterías como pgmpy.

Part III
Ejercicios de libro.

6 Problemas.

6.1 Problema 14.4.

a. Efectivamente. Se puede calcular numéricamente que P(B, E) = P(B)P(E). Topológicamente, B y E están d-separados por A.
b. Comprobamos si P(B, E|a) = P(B|a)P(E|a). Primero, calculamos P(B, E|a):

$$\mathrm{P}(\mathrm{B},\mathrm{E}|\mathrm{a}) = \alpha \mathrm{P}(\mathrm{a}|\mathrm{B},\mathrm{E})\mathrm{P}(\mathrm{B},\mathrm{E})$$

$$= \alpha \begin{cases} 0.95 \times 0.001 \times 0.002 & \text{si B} = \text{b y E} = \text{e} \\ 0.94 \times 0.001 \times 0.998 & \text{si B} = \text{b y E} = \neg\text{e} \\ 0.29 \times 0.999 \times 0.002 & \text{si B} = \neg\text{b y E} = \text{e} \\ 0.001 \times 0.999 \times 0.998 & \text{si B} = \neg\text{b y E} = \neg\text{e} \end{cases}$$

$$= \alpha \begin{cases} 0.0008 & \text{si B} = \text{b y E} = \text{e} \\ 0.3728 & \text{si B} = \text{b y E} = \neg\text{e} \\ 0.2303 & \text{si B} = \neg\text{b y E} = \text{e} \\ 0.3962 & \text{si B} = \neg\text{b y E} = \neg\text{e} \end{cases}$$

donde α es una constante de normalización. Al verificar si P(b, e|a) = P(b|a)P(e|a), encontramos que:

$$\mathrm{P}(\mathrm{b},\mathrm{e}|\mathrm{a}) = 0.0008 \neq 0.0863 = 0.3736 \times 0.2311 = \mathrm{P}(\mathrm{b}|\mathrm{a})\mathrm{P}(\mathrm{e}|\mathrm{a})$$

lo que demuestra que B y E no son independientes condicionalmente dado A.

6.2 Problema 14.8.

Agregar variables a una red existente puede realizarse de dos maneras. Hablando formalmente, se deben insertar las variables en el orden de variables y volver a ejecutar el proceso de construcción de la red desde el punto en que aparece la primera nueva variable. Hablando informalmente, nunca se construye una red siguiendo un orden estricto. En su lugar, se pregunta qué variables son causas o influencias directas sobre otras, y se construyen gráficos locales de padres/hijos de esa manera. Generalmente, es fácil identificar dónde encaja la nueva variable en tal estructura, pero es fundamental verificar posibles dependencias inducidas hacia abajo.
a. IcyWeather no es causada por ninguna de las variables relacionadas con el automóvil, por lo que no necesita padres. Afecta directamente a la batería y al motor de arranque. StarterMotor es una condición previa adicional para Starts. La nueva red se muestra en la Figura S14.1.
b. Las probabilidades razonables pueden variar significativamente dependiendo del tipo de automóvil y quizás de la experiencia personal del evaluador. Los siguientes valores indican el orden de magnitud general y los valores relativos que tienen sentido:

- Una probabilidad previa razonable para IcyWeather podría ser 0.05 (quizás dependiendo de la ubicación y la temporada).

- P(Battery|IcyWeather) = 0.95, P(Battery|¬IcyWeather) = 0.997.
- P(StarterMotor|IcyWeather) = 0.98, P(Battery|¬IcyWeather) = 0.999.
- P(Radio|Battery) = 0.9999, P(Radio|¬Battery) = 0.05.
- P(Ignition|Battery) = 0.998, P(Ignition|¬Battery) = 0.01.
- P(Gas) = 0.995.
- P(Starts|Ignition, StarterMotor, Gas) = 0.9999, otras entradas 0.0.
- P(Moves|Starts) = 0.998.

6.3 Problema 14.14.

La red afirma (ii) y (iii). (Para (iii), considera la manta de Markov de M.)
b. $P(b, i, \neg m, g, j) = P(b)P(\neg m)P(i|b, \neg m)P(g|b, i, \neg m)P(j|g) = 0.9 \times 0.9 \times 0.5 \times 0.8 \times 0.9 = 0.2916$
c. Dado que B, I, M están fijos como verdaderos en la evidencia, podemos tratar a G como si tuviera una probabilidad previa de 0.9 y simplemente observar el submodelo con G y J:

$$P(J|b, i, m) = \alpha \sum_{g} P(J, g) = \alpha \left[P(J, g) + P(J, \neg g)\right]$$

$$= \alpha \left[\langle P(j, g), P(\neg j, g)\rangle + \langle P(j, \neg g), P(\neg j, \neg g)\rangle\right]$$

$$= \alpha \left[\langle 0.81, 0.09\rangle + \langle 0, 0.1\rangle\right] = \langle 0.81, 0.19\rangle$$

Es decir, la probabilidad de ir a la cárcel es 0.81.
d. Intuitivamente, una persona no puede ser declarada culpable si no es acusada, sin importar si quebrantó la ley y sin importar el fiscal. Esto es lo que indica la tabla de probabilidad condicional (CPT) para G; por lo tanto, G es independientemente contextual de B y M dado que I = false.
e. Un indulto es innecesario si la persona no es acusada o no es declarada culpable; así que I y G son padres de P. También se podrían agregar B y M como padres de P, ya que es más probable que se conceda un indulto si la persona es realmente inocente y si el fiscal tiene motivos políticos. (Existen otras causas de indulto, como LargeDonationToPresidentsParty, pero tales variables no están actualmente en el modelo). El indulto (presumiblemente) es una carta de salida libre de la cárcel, por lo que P es un padre de J.

6.4 Problema 14.16.

1. Considere una fórmula 3-CNF $C_1 \wedge \ldots \wedge C_n$ con n cláusulas, donde cada cláusula es una disyunción $C_i = (\ell_{i1} \vee \ell_{i2} \vee \ell_{i3})$ de literales, es decir, cada ℓ_{ij} es P_k o $\neg P_k$ para alguna proposición atómica $P_1, \ldots, P_m$.
    - Construya una red bayesiana con una variable (booleano) S para toda la fórmula, C_i para cada cláusula y P_k para cada proposición atómica. Definiremos padres y tablas de probabilidad condicional (CPT) de tal manera que, para cualquier asignación a las proposiciones atómicas, S es verdadero si y solo si la fórmula 3-CNF es verdadera.

- Las proposiciones atómicas no tienen padres y son verdaderas con una probabilidad de 0.5. Cada cláusula C_i tiene como padres las proposiciones atómicas correspondientes a los literales ℓ_{i1}, ℓ_{i2} y ℓ_{i3}. La variable de cláusula es verdadera si y solo si uno de sus literales es verdadero. Note que esto es una CPT determinista. Finalmente, S tiene todas las variables de cláusulas C_i como sus padres y es verdadera si y solo si todas las variables de cláusulas son verdaderas.
- Observe que $P(S = \text{True}) > 0$ si y solo si la fórmula es satisfacible, y la inferencia exacta responderá a esta pregunta.

2. Utilizando la misma red que en la parte (a), note que $P(S = \text{True}) = s^{2-m}$ donde s es el número de asignaciones satisfacibles a las proposiciones atómicas $P_1, \ldots, P_m$.

6.5 Problema 14.21.

1. Las clases son *Equipo*, con instancias A, B y C, y *Partido*, con instancias AB, BC y CA. Cada equipo tiene una calidad Q y cada partido tiene Equipo1, Equipo2 y un *Resultado*. Los nombres de los equipos para cada partido están, por supuesto, fijos de antemano. La distribución previa sobre la calidad podría ser uniforme, y la probabilidad de ganar para el equipo 1 debería aumentar conforme a Q(Equipo1) – Q(Equipo2).

2. Las variables aleatorias son A.Q, B.Q, C.Q, AB.Resultado, BC.Resultado, y CA.Resultado. La red se muestra en la Figura S14.3.

3. El resultado exacto dependerá de las probabilidades utilizadas en el modelo. Con cualquier distribución previa sobre la calidad que sea la misma para todos los equipos, esperamos que la posterior sobre BC.Resultado muestre que C es más probable que gane que B.

4. El costo de inferencia en tal modelo será $O(2^n)$ porque todas las calidades de los equipos se vuelven interdependientes.

5. El método de Monte Carlo por cadenas de Markov (MCMC) parece funcionar bien en este problema, siempre que las probabilidades no estén demasiado sesgadas. Nuestros resultados muestran un comportamiento de escalado que es aproximadamente lineal en el número de equipos, aunque no investigamos valores de n muy grandes.