Escuela Técnica Superior de Ingeniería Informática

Inteligencia Artificial Avanzada

Práctica de Programación: Inferencia Condicional en Distribuciones Discretas Binarias

Objetivo

Implementar un programa que permita calcular **distribuciones condicionales** a partir de una **distribución conjunta discreta de variables binarias**, utilizando **marginalización de variables**.

El programa permitirá generar o cargar la distribución, seleccionar variables de interés y condicionadas de forma intuitiva, calcular automáticamente la distribución condicional y **analizar el tiempo de ejecución** en función del número de variables de interés y de condicionadas.

1. Carga de la distribución conjunta

1.1 Leer desde un CSV

Cada línea del CSV debe tener el formato:

`máscara_binaria, probabilidad`

- **máscara_binaria**: cadena de 0's y 1's que representa el estado completo de todas las variables binarias.
  - La longitud de la máscara es igual al número de variables N.
  - Cada bit indica el valor de una variable (cero o uno):
    - Bit 0 (derecha) $\rightarrow X_1$
    - Bit 1 $\rightarrow X_2$
    - ...
    - Bit $N-1$ (izquierda) $\rightarrow X_N$
- **probabilidad**: valor de $P(X_1,\ldots,X_N)$ correspondiente a esa configuración.
- La suma de todas las probabilidades debe ser 1.

Ejemplo con $N = 3$:

```
000,0.10
001,0.05
010,0.15
011,0.10
100,0.20
101,0.10
110,0.20
111,0.10
```

Interpretación:

X3	X2	X1	Probabilidad
0	0	0	0.10
0	0	1	0.05
0	1	0	0.15
0	1	1	0.10
1	0	0	0.20
1	0	1	0.10
1	1	0	0.20
1	1	1	0.10

1.2 Generación aleatoria

Generar valores positivos y normalizarlos (dividirlos por su suma) para que la suma de probabilidades sea 1.

2. Selección de variables

El usuario podrá seleccionar:

- **Variables condicionadas** X_C y sus valores x_C, introduciendo los índices de las variables (1 a N) y los valores (0 o 1).
- **Variables de interés** X_I, introduciendo los índices de las variables cuya distribución condicional se desea calcular.

Las variables que no estén en X_C ni en X_I se consideran **variables a marginalizar**.

3. Carga de las probabilidades en un array de probabilidades

La distribución de probabiildad se almacenará en un array unidimensional `p[k]`. Cada índice k codifica en binario la configuración de todas las variables:

- Bit 0 $\rightarrow X_1$
- Bit 1 $\rightarrow X_2$
- ...
- Bit $N-1 \rightarrow X_N$

Ejemplo con $N = 3$:

Máscara "011" → X3=0, X2=1, X1=1 Índice k = 3 p[3] = 0.10

4. Uso de máscaras en el cálculo de la probabilidad condicional

- `maskC` → variables condicionadas
- `valC` → valores de las variables condicionadas
- `maskI` → variables de interés
- Las demás variables se consideran a marginalizar

Ejemplo:

- N = 3 (X1, X2, X3)
- Condición: X2=1 → maskC = 010, valC = 010
- Variable de interés: X1 → maskI = 001
- Variable a marginalizar: X3 → maskM = 100

X1	X2	X3	indices	p[k]
0	1	0	010	p(2)
0	1	1	110	p(6)
1	1	0	011	p(3)
1	1	1	111	p(7)

$P(X_1, X_2=1)$

X1	indices	p[k]
0	0	p(2)+p(6)
1	1	p(3)+p(7)

Se normaliza para obtener $P(X_1|X_2 = 1)$.

5. Función principal de cálculo

```
double* prob_cond_bin(double *p, int N, int maskC, int valC, int maskI);
```

Devuelve un array `out` con la distribución condicional normalizada.

6. Salida del programa

- Muestra la distribución generada o leída del CSV.
- Muestra las probabilidades condicionadas $P(X_I|X_C)$ para todas las combinaciones de X_I.

7. Estudio del tiempo de ejecución

Además de calcular las probabilidades condicionales, se debe medir el tiempo de ejecución de la función `prob_cond_bin` y estudiar cómo varía en función de:

1. Número de variables de interés $|X_I|$
2. Número de variables condicionadas $|X_C|$

- Generar varias configuraciones con el mismo N y distintos conjuntos de variables de interés y condicionadas.
- Presentar tabla o gráfico mostrando el crecimiento del tiempo de ejecución.

8. Requisitos de implementación

1. Debe funcionar (salvo problemas de memoria) para cualquier número de variables binarias (hasta 64) y cualquier combinación de variables condicionadas/interés.
2. Código correctamente comentado
3. Lenguaje de programación libre
4. Grupos de dos alumnos

9. Entregables

- Primera sesión

  Código fuente de 1,2,3

- Segunda sesión

Informe con:

1. Código fuente (enlace a repositorio)
2. Ejemplo de ejecución mostrando distribución condicional, variables condicionadas/interés y resultados.
3. Archivo CSV de ejemplo en formato `máscara_binaria,probabilidad`.
4. Informe breve con análisis del tiempo de ejecución, incluyendo tablas o gráficos.
5. Porcentaje de participación de cada alumno en la ejecución del trabajo.

10. Criterios de evaluación

Criterio	Puntos
Primera sesión	
Correcta generación o lectura de la distribución conjunta	20
Entrada intuitiva de variables condicionadas e interés	15
Segunda sesión	25
Correcta implementación de marginalización y normalización	
Resultados correctos de la distribución condicional	20
Estudio y análisis del tiempo de ejecución	10
Calidad del código (comentarios, legibilidad, manejo de memoria)	10

NOTA: En todas las prácticas se pasan detectores antiplagio.