```
pip install pgmpy pandas numpy scikit-learn
```

Show hidden output

```
import pandas as pd
from pgmpy.models import BayesianNetwork, DiscreteBayesianNetwork
from pgmpy.estimators import MaximumLikelihoodEstimator
from pgmpy.inference import VariableElimination
from sklearn.model_selection import train_test_split

# Load the Heart Disease dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data"
columns = [
    "age", "sex", "cp", "trestbps", "chol", "fbs", "restecg",
    "thalach", "exang", "oldpeak", "slope", "ca", "thal", "target"
]
data = pd.read_csv(url, names=columns, na_values='?')

# Drop rows with missing values
data.dropna(inplace=True)

# Convert target to binary: 0 (no heart disease), 1 (heart disease)
data['target'] = data['target'].apply(lambda x: 1 if x > 0 else 0)

# Discretize continuous variables
data['age'] = pd.qcut(data['age'], q=3, labels=[0, 1, 2])  # 3 bins
data['chol'] = pd.qcut(data['chol'], q=3, labels=[0, 1, 2])
data['trestbps'] = pd.qcut(data['trestbps'], q=3, labels=[0, 1, 2])

# Convert categorical columns to numeric (example: cp, thal, slope)
data['cp'] = data['cp'].astype(int)
data['thal'] = data['thal'].astype(int)
data['slope'] = data['slope'].astype(int)

# Split into train/test sets
train_data, test_data = train_test_split(data, test_size=0.2, random_state=42)
print("Data shape:", train_data.shape, test_data.shape)
```

```
Data shape: (237, 14) (60, 14)
```

```
data
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	2	1.0	1	2	1	1.0	2.0	150.0	0.0	2.3	3	0.0	6	0
1	2	1.0	4	2	2	0.0	2.0	108.0	1.0	1.5	2	3.0	3	1
2	2	1.0	4	0	1	0.0	2.0	129.0	1.0	2.6	2	2.0	7	1
3	0	1.0	3	1	1	0.0	0.0	187.0	0.0	3.5	3	0.0	3	0
4	0	0.0	2	1	0	0.0	2.0	172.0	0.0	1.4	1	0.0	3	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
297	1	0.0	4	2	1	0.0	0.0	123.0	1.0	0.2	2	0.0	7	1
298	0	1.0	1	0	1	0.0	0.0	132.0	0.0	1.2	2	0.0	7	1
299	2	1.0	4	2	0	1.0	0.0	141.0	0.0	3.4	2	2.0	7	1
300	1	1.0	4	1	0	0.0	0.0	115.0	1.0	1.2	2	1.0	7	1
301	1	0.0	2	1	1	0.0	2.0	174.0	0.0	0.0	2	1.0	3	1

297 rows × 14 columns

```
model = DiscreteBayesianNetwork([
    # Edges based on medical intuition
    ('age', 'chol'),          # Age affects cholesterol
    ('sex', 'trestbps'),      # Sex affects blood pressure
    ('cp', 'target'),         # Chest pain type affects heart disease
    ('chol', 'target'),       # Cholesterol affects heart disease
    ('trestbps', 'target'),   # Blood pressure affects heart disease
    ('thal', 'target'),       # Thalassemia affects heart disease
    ('slope', 'target'),      # Slope of ST segment affects heart disease
    ('fbs', 'target'),        # Fasting blood sugar affects heart disease
    ('restecg', 'target'),    # Resting electrocardiographic results affect heart disease
    ('thalach', 'target'),    # Maximum heart rate achieved affects heart disease
    ('exang', 'target'),
    ('oldpeak', 'target'),
    ('fbs', 'target'),
```

```
    ('restecg', 'target'),
    ('ca', 'target')
])


# Fit the model on training data
model.fit(train_data, estimator=MaximumLikelihoodEstimator)

# # Print CPDs (Conditional Probability Distributions)
# for cpd in model.get_cpds():
#     print("CPD for", cpd.variable)
#     print(cpd, "\n")
```

```
<pgmpy.models.DiscreteBayesianNetwork.DiscreteBayesianNetwork at 0x7fa09b8fcf50>
```

```
model.get_cpds('age')
model.get_cpds('sex')
model.get_cpds('cp')
model.get_cpds('thal')
```

```
<TabularCPD representing P(thal:3) at 0x7fa09bd87490>
```

```
# Check the state names for the 'thal' variable
print(model.get_cpds('thal'))
```

```
+---------+-----------+
| thal(3) | 0.540084  |
+---------+-----------+
| thal(6) | 0.0548523 |
+---------+-----------+
| thal(7) | 0.405063  |
+---------+-----------+
```

```
# Initialize inference engine
inference = VariableElimination(model)

# Example: Predict heart disease for a patient with:
# age=1, sex=1, cp=3, trestbps=1, chol=1, thal=3, slope=2
query = inference.query(
    variables=['target'],
    evidence={
        'age': 1,
        'sex': 1,
        'cp': 3,
        'trestbps': 1,
        'chol': 1,
        'thal': 7,
        'slope': 2
    }
)
print("Probability of heart disease:", query.values[1])
```

```
Probability of heart disease: 0.5000229253446513
```

```
Start coding or generate with AI.
```