# Untitled.txt

```
from pgmpy.models import BayesianModel
from pgmpy.factors.discrete import TabularCPD
from pgmpy.inference import VariableElimination
import numpy as np

# Define the structure of the Bayesian network
model = BayesianModel([('C', 'S'), ('D', 'S')])

# Define the conditional probability distributions (CPDs)
cpd_c = TabularCPD('C', 2, [[0.5], [0.5]])
cpd_d = TabularCPD('D', 2, [[0.5], [0.5]])
cpd_s = TabularCPD('S', 2,
[[0.8, 0.6, 0.6, 0.2],
[0.2, 0.4, 0.4, 0.8]],
evidence=['C', 'D'],
evidence_card=[2, 2])

# Add the CPDs to the model
model.add_cpds(cpd_c, cpd_d, cpd_s)

# Check if the model is valid
assert model.check_model()

# Generate synthetic data with 3 columns: C, D, S
# We'll generate the data manually to match the model
data = []

for _ in range(5000):
c = np.random.choice([0, 1])
d = np.random.choice([0, 1])
prob_s = [0.8, 0.2] if (c, d) == (0, 0) else \
[0.6, 0.4] if (c, d) == (0, 1) else \
[0.6, 0.4] if (c, d) == (1, 0) else \
[0.2, 0.8]
s = np.random.choice([0, 1], p=prob_s)
data.append([c, d, s])

import pandas as pd
df = pd.DataFrame(data, columns=['C', 'D', 'S'])

# Fit the model using Maximum Likelihood Estimator
from pgmpy.estimators import MaximumLikelihoodEstimator
mle = MaximumLikelihoodEstimator(model, df)
model.fit(df)

# Perform inference
infer = VariableElimination(model)
query = infer.query(['S'], evidence={'C': 1})
```

```
print(query)
```