The Monty Hall problem

Bruna Wundervald

October, 2018

Original code here: https://github.com/pgmpy/pgmpy

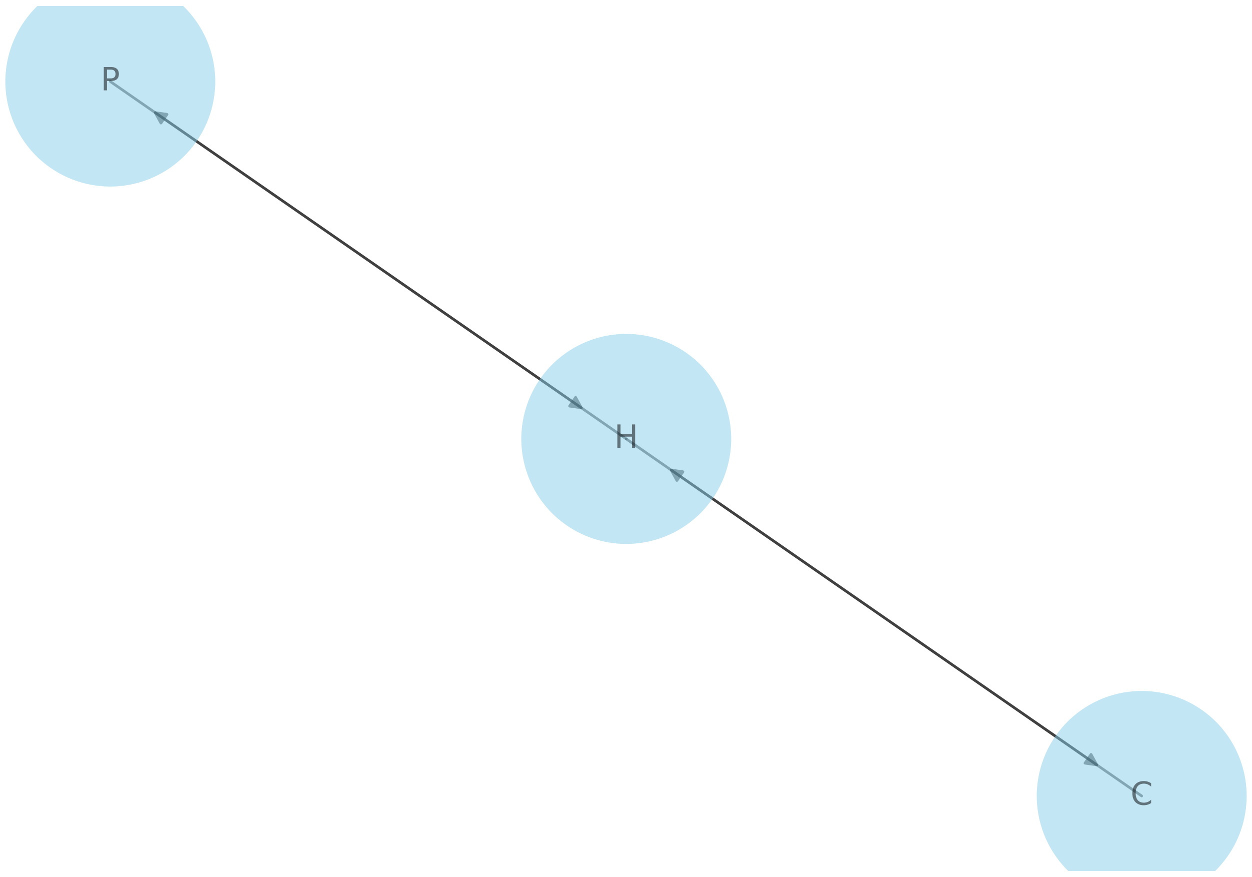
```
# Packages and models import
# Starting with defining the network structure
from pgmpy.models import BayesianModel
from pgmpy.factors.discrete import TabularCPD

# Infering the posterior probability
from pgmpy.inference import VariableElimination

# Graph packages
import pandas as pd
import numpy as np
import networkx as nx
import matplotlib.pyplot as plt
```

```
# Build a dataframe with the connections
df = pd.DataFrame({ 'from':['C', 'P'], 'to':['H', 'H']})
df

# Build your graph
```

```
##   from to
## 0    C  H
## 1    P  H
```

```
G=nx.from_pandas_edgelist(df, 'from', 'to')
G = nx.DiGraph(G)

# Graph with Custom nodes:
plt.clf()
nx.draw(G, with_labels=True, node_size=1500, node_color="skyblue", node_shape="o", alpha=0.5, linewidths
plt.show()
```

```
## 'monty-hall_files/figure-latex/unnamed-chunk-2-1.pdf'
```



```
# Defining the network structure
model = BayesianModel([('C', 'H'), ('P', 'H')])



# Defining the CPDs:

# Choosen door: is has the same probability of being
# any of the 3 doors
cpd_c = TabularCPD('C', 3, [[0.33, 0.33, 0.33]])

# Prize: first, is has the same probability of being
# in any of the 3 doors
cpd_p = TabularCPD('P', 3, [[0.33, 0.33, 0.33]])

# P(H | P, C): probability of the host opening a door,
# given that the constestant chose one and where the prize
# really is
cpd_h = TabularCPD('H', 3, [[0, 0, 0, 0, 0.5, 1, 0, 1, 0.5],
                            [0.5, 0, 1, 0, 0, 0, 1, 0, 0.5],
                            [0.5, 1, 0, 1, 0.5, 0, 0, 0, 0]],
                   evidence=['C', 'P'], evidence_card=[3, 3])

# Associating the CPDs with the network structure.
model.add_cpds(cpd_c, cpd_p, cpd_h)
```

```
# check_model check for the model structure and the associated CPD and returns True if everything is co
model.check_model()

# Variable elimination
```

```
## True
```

```
infer = VariableElimination(model)


# Finding posterior probability:
# What is the probability of each door given
# that the constestant choose 0 and the host opened 2?

posterior_p = infer.query(['P'], evidence={'C': 0, 'H': 2})
# print(posterior_p['P'])
#  | P   |   phi(P) |
# | P_0 |   0.3333 |
# | P_1 |   0.6667 |
# | P_2 |   0.0000|

posterior_p = infer.query(['P'], evidence={'C': 2, 'H': 1})
# print(posterior_p['P'])
#  | P   |   phi(P) |
# | P_0 |   0.6667 |
# | P_1 |   0.0000 |
# | P_2 |   0.3333 |

# It's worth it to change the door!
```