Bhuvi Ghosh
60009210191

In [1]:
```
import matplotlib.pyplot as plt
import numpy as np

from pomegranate import State, HiddenMarkovModel, DiscreteDistribution
```

In [2]:
```
model = HiddenMarkovModel(name="Climate Model")
```

In [3]:
```
rainy_emissions = DiscreteDistribution({"Happy":0.9,"Sad":0.1})
rainy_state = State(rainy_emissions,name = "rainy")

cloudy_emissions = DiscreteDistribution({"Happy":0.6,"Sad":0.4})
cloudy_state = State(cloudy_emissions,name = "cloudy")

sunny_emissions = DiscreteDistribution({"Happy":0.2,"Sad":0.8})
sunny_state = State(cloudy_emissions,name = "sunny")
```

In [4]:
```
model.add_states(rainy_state,cloudy_state,sunny_state)
```

In [6]:
```
model.add_transition(model.start, rainy_state, 0.33)
model.add_transition(model.start, cloudy_state ,0.33)
model.add_transition(model.start, sunny_state ,0.34)
#Adding Transitions for Rainy State
model.add_transition(rainy_state, rainy_state , 0.5)
model.add_transition(rainy_state, cloudy_state,0.3)
model.add_transition(rainy_state, rainy_state,0.2)
#Adding Transitions for Cloudy State
model.add_transition(cloudy_state,rainy_state,0.4)
model.add_transition(cloudy_state,cloudy_state, 0.2)
model.add_transition(cloudy_state,sunny_state,0.4)
#Adding Transitions for Sunny State
model.add_transition(sunny_state,rainy_state , 0.0)
model.add_transition(sunny_state,cloudy_state , 0.3)
model.add_transition(sunny_state,sunny_state , 0.7)
```

In [7]:
```
model.bake()
```

In [8]:
```
model.edge_count()
```

Out[8]:
```
11
```

In [9]:
```
model.node_count()
```

Out[9]:
```
5
```

In [13]:
```
column_order = ["Climate Model-start", "rainy", "cloudy", "sunny", "Climate Model
column_names = [s.name for s in model.states]
order_index = [column_names.index(c) for c in column_order]
transitions = model.dense_transition_matrix()[:, order_index][order_index, :]
print("The state transition matrix, P(Xt|Xt-1):\n")
print(transitions)
print("\nThe transition probability from Cloudy to Rainy is {:.0f}%".format(100 *
```

```
The state transition matrix, P(Xt|Xt-1):

[[0.   0.33 0.33 0.34 0.  ]
 [0.   0.4  0.6  0.   0.  ]
 [0.   0.4  0.2  0.4  0.  ]
 [0.   0.   0.3  0.7  0.  ]
 [0.   0.   0.   0.   0.  ]]

The transition probability from Cloudy to Rainy is 40%
```

Calculate Sequence Likelihood

In [16]:
```
observations = ['Happy', 'Sad', 'Happy']
assert len(observations) > 0
forward_matrix = np.exp(model.forward(observations))
probability_percentage = np.exp(model.log_probability(observations))
print("          " + "".join(s.name.center(len(s.name)+6) for s in model.states))
for i in range(len(observations) + 1):
    print("   " if i==0 else observations[i - 1].center(9), end="")
    print("".join("{:.0f}%".format(100 * forward_matrix[i, j]).center(len(s.name)
                  for j, s in enumerate(model.states)))

print("\nThe likelihood over all possible paths " + \
      "of this model producing the sequence {} is {:.2f}%\n\n"
      .format(observations, 100 * probability_percentage))
```

```
            cloudy      rainy      sunny      Climate Model-start      Climat
e Model-end
       0%          0%         0%                100%                     0%
  Happy      20%         30%        20%                 0%
0%
   Sad       11%          2%         9%                 0%
0%
  Happy       4%          5%         6%                 0%
0%

The likelihood over all possible paths of this model producing the sequence
['Happy', 'Sad', 'Happy'] is 14.79%
```

Decoding the Most Likely Hidden State

Sequence

In [17]:
```
observations = ['Happy', 'Sad', 'Happy']
viterbi_likelihood, viterbi_path = model.viterbi(observations)
print("The most likely weather sequence to have generated " + \
      "these observations is {} at {:.2f}%."
      .format([s[1].name for s in viterbi_path[1:]], np.exp(viterbi_likelihood)*1
)
```

```
The most likely weather sequence to have generated these observations is ['rain
y', 'cloudy', 'rainy'] at 2.57%.
```

Forward likelihood vs Viterbi likelihood

In [24]:
```
from itertools import product

observations = ['Happy', 'Happy', 'Sad']

p = {'Rainy' : {'Rainy': np.log(.5),'Cloudy': np.log(.3),'Sunny': np.log(.2)},'Cl
e = {'Rainy' : {'Sad': np.log(.9),'Happy': np.log(.1)},'Cloudy': {'Sad': np.log(.
```

In [26]:

```
o = observations
k = []
vprob = np.exp(model.viterbi(o)[0])
print("The likelihood of observing {} if the weather sequence is...".format(o))
for s in product(*[['Rainy', 'Cloudy','Sunny']]*3):
    k.append(np.exp(np.log(.5)+e[s[0]][o[0]] + p[s[0]][s[1]] + e[s[1]][o[1]] + p[
    print("\t{} is {:.2f}% {}".format(s, 100 * k[-1], " <-- Viterbi path" if k[-1
print("\nThe total likelihood of observing {} over all possible paths is {:.2f}%'
```

```
0.025660799999999999
The likelihood of observing ['Happy', 'Happy', 'Sad'] if the weather sequence i
s...
        ('Rainy', 'Rainy', 'Rainy') is 0.11%
        ('Rainy', 'Rainy', 'Cloudy') is 0.05%
        ('Rainy', 'Rainy', 'Sunny') is 0.01%
        ('Rainy', 'Cloudy', 'Rainy') is 0.22%
        ('Rainy', 'Cloudy', 'Cloudy') is 0.07%
        ('Rainy', 'Cloudy', 'Sunny') is 0.05%
        ('Rainy', 'Sunny', 'Rainy') is 0.00%
        ('Rainy', 'Sunny', 'Cloudy') is 0.14%
        ('Rainy', 'Sunny', 'Sunny') is 0.11%
        ('Cloudy', 'Rainy', 'Rainy') is 0.36%
        ('Cloudy', 'Rainy', 'Cloudy') is 0.14%
        ('Cloudy', 'Rainy', 'Sunny') is 0.03%
        ('Cloudy', 'Cloudy', 'Rainy') is 0.58%
        ('Cloudy', 'Cloudy', 'Cloudy') is 0.19%
        ('Cloudy', 'Cloudy', 'Sunny') is 0.13%
        ('Cloudy', 'Sunny', 'Rainy') is 0.00%
        ('Cloudy', 'Sunny', 'Cloudy') is 1.15%
        ('Cloudy', 'Sunny', 'Sunny') is 0.90%
        ('Sunny', 'Rainy', 'Rainy') is 0.00%
        ('Sunny', 'Rainy', 'Cloudy') is 0.00%
        ('Sunny', 'Rainy', 'Sunny') is 0.00%
        ('Sunny', 'Cloudy', 'Rainy') is 1.73%
        ('Sunny', 'Cloudy', 'Cloudy') is 0.58%
        ('Sunny', 'Cloudy', 'Sunny') is 0.38%
        ('Sunny', 'Sunny', 'Rainy') is 0.00%
        ('Sunny', 'Sunny', 'Cloudy') is 4.03%
        ('Sunny', 'Sunny', 'Sunny') is 3.14%

The total likelihood of observing ['Happy', 'Happy', 'Sad'] over all possible p
aths is 14.10%
```

In [ ]: