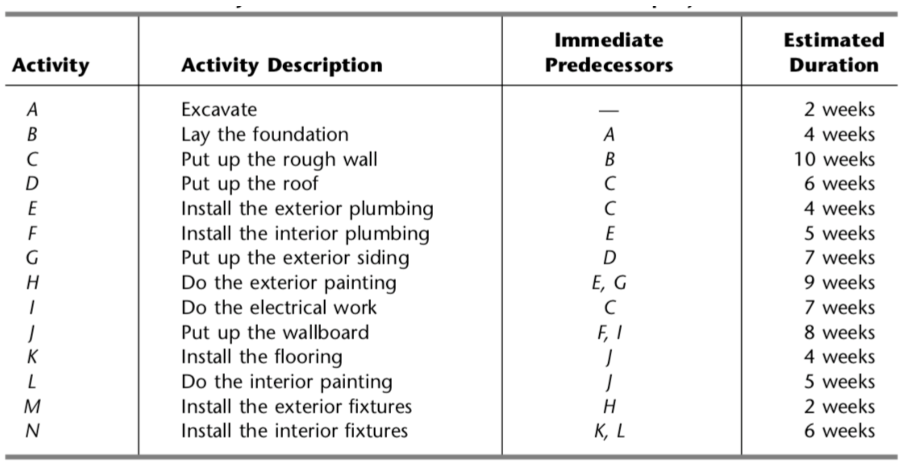
Kernel: Python 3 (system-wide)

Project 3 Report - Scheduling and Decision Analysis with Uncertainty

Deanna Schneider contributed the bulk of helped this project. THANK YOU Deanna. Don't blame her for the decision analysis though ... that was our idea.

For the final project, we're going to combine concepts from Lesson 7 (Constraint Programming), Lesson 8 (Simulation), and Lesson 9 (Decision Analysis). We'll do this by revisiting the scheduling problem from Lesson 7. But, we're going to make it a little more true-to-life by acknowledging some of the uncertainty in our estimates, and using simulation to help us come up with better estimates. We'll use our estimated profits to construct a payoff table and make a decision about how to proceed with the building project.

When we originally created the problem, we used the following estimates for time that each task would take:



Activity	Activity Description	Immediate Predecessors	Estimated Duration
A	Excavate	—	2 weeks
B	Lay the foundation	*A*	4 weeks
C	Put up the rough wall	*B*	10 weeks
D	Put up the roof	*C*	6 weeks
E	Install the exterior plumbing	*C*	4 weeks
F	Install the interior plumbing	*E*	5 weeks
G	Put up the exterior siding	*D*	7 weeks
H	Do the exterior painting	*E, G*	9 weeks
I	Do the electrical work	*C*	7 weeks
J	Put up the wallboard	*F, I*	8 weeks
K	Install the flooring	*J*	4 weeks
L	Do the interior painting	*J*	5 weeks
M	Install the exterior fixtures	*H*	2 weeks
N	Install the interior fixtures	*K, L*	6 weeks

But based on past experience, we know that these are just the most likely estimates of the time needed for each task. Here's our estimated ranges of values (in days instead of weeks) for each task:

	Optimistic Estimate	Most Likely Estimate	Pessimistic Estimate
A: Excavate	7	14	21
B: Lay Foundation	14	21	56
C: Put Up Rough Wall	42	63	126
D: Put Up Roof	28	35	70
E: Exterior Plumbing	7	28	35
F: Interior Plumbing	28	35	70
G: Exterior Siding	35	42	77
H: Exterior Painting	35	56	119
I: Electrical	21	49	63
J: Wallboard	21	63	63
K: Flooring	21	28	28
L: Interior Painting	7	35	49
M: Exterior Fixtures	7	14	21
N: Interior Fixtures	35	35	63

Further, we're going to consider the following factors:

- The base amount that Reliable will earn is $5.4 million.
- If Reliable completes the project in 280 days or less, they will get a bonus of $150,000.
- If Reliable misses the deadline of 329 days, there will be a $25,000 penalty for each day over 329.

P3.1 - Simulation

Create a simulation that uses a triangular distribution to estimate the duration for each of the activities. Use the Optimistic Estimate, Most Likely Estimate, and Pessimistic Estimate for the 3 parameters of your triangular distribution. Use CP-SAT to find the minimal schedule length in each iteration. Track the total days each simulation takes and the profit for the company.

Put your simulation code in the cell below. Use at least 1000 iterations. Check your simulation results to make sure the tasks are being executed in the correct order!

*** 8 points - answer in cell below *** (don't delete this cell)

In [15]:
```
from ortools.sat.python import cp_model
import numpy as np

task_params_dict = {
    'A': (7, 14, 21),
    'B': (14, 21, 56),
    'C': (42, 63, 126),
    'D': (28, 35, 70),
    'E': (7, 28, 35),
    'F': (28, 35, 70),
    'G': (35, 42, 77),
    'H': (35, 56, 119),
    'I': (21, 49, 63),
    'J': (21, 63, 63),
    'K': (21, 28, 28),
    'L': (7, 35, 49),
    'M': (7, 14, 21),
    'N': (35, 35, 63)
}

task_names = list(task_params_dict.keys())
num_tasks = len(task_names)



precedence_dict = {
    'A': ['B'],
    'B': ['C'],
    'C': ['D', 'E', 'I'],
    'D': ['G'],
    'E': ['F', 'H'],
    'F': ['J'],
    'G': ['H'],
    'H': ['M'],
    'I': ['J'],
    'J': ['K', 'L'],
    'K': ['N'],
    'L': ['N']
}

simSize = 1000
daysTaken = np.zeros(simSize)

for i in range(simSize):
    task_duration_dict = {
        t: round(
            np.random.triangular(task_params_dict[t][0],
task_params_dict[t][1], task_params_dict[t][2]))
        for t in task_names
    }
    durations  = list(task_duration_dict.values())
    task_name_to_number_dict = dict(zip(task_names, np.arange(0,
num_tasks)))

    horizon = int(sum(task_duration_dict.values()))

    model = cp_model.CpModel()

    start_vars = [
        model.NewIntVar(0, horizon, name=f'start_{t}') for t in
task_names
    ]

    end_vars = [model.NewIntVar(0, horizon, name=f' end_{t}') for t in
task_names]

    intervals = [
        model.NewIntervalVar(start_vars[j],
                             durations[j],
                             end_vars[j],
                             name=f'interval_{task_names[j]}')
        for j in range(num_tasks)
    ]
    for before in list(precedence_dict.keys()):
        for after in precedence_dict[before]:
            before_index = task_name_to_number_dict[before]
            after_index = task_name_to_number_dict[after]
            model.Add(end_vars[before_index] <=
start_vars[after_index])

    obj_var = model.NewIntVar(0, horizon, 'largest_end_time')
    model.AddMaxEquality(obj_var, end_vars)
    model.Minimize(obj_var)

    solver = cp_model.CpSolver()
    status = solver.Solve(model)

    daysTaken[i] = solver.ObjectiveValue()
```

What is the probability that Reliable Company will finish the bid in less than 280 days, between 280 and 329 days, and over 329 days? What is their average profit?

Include code to answer these questions with output below:

*** 2 points - answer in cell below *** (don't delete this cell)

In [51]:
```
p_low = np.sum(daysTaken <= 280)/ simSize
p_high = np.sum(daysTaken >= 329) / simSize
p_mid = 1 - (p_low + p_high)
print(f"The probability that construction would take less than 280
days is: {p_low:.3f}")
print(f"The probability that construction would take between 281 days
and 329 days is: {p_mid:.3f}")
print(f"The probability that construction would take more than 329
days is : {p_high:.3f}")

profit = np.zeros(simSize)
for i in range(len(daysTaken)):
    if daysTaken[i] <= 280:
        profit[i] = 5.55
    elif daysTaken[i] >= 329:
        oops = daysTaken[i] - 329
        profit[i] = 5.4 - (oops * 0.025)
    else:
        profit[i] = 5.4
meanProfit = np.mean(profit)
print(f"The average profit is: ${meanProfit:.2f} million")
```

Out[51]:
```
The probability that construction would take less than 280 days is:
0.053
The probability that construction would take between 281 days and 329
days is: 0.563
The probability that construction would take more than 329 days is :
0.384
The average profit is: $5.24 million
```

P3.2 - Add Random Cost

From past experience, we know that special artifacts are sometimes found in the area where Reliable Construction is planning this building project. When special artifacts are found, the excavation phase takes considerably longer and the entire project costs more - sometimes much more. They're never quite sure how much longer it will take, but it peaks around an extra 15 days, and takes at least an extra 7 days. They've seen some sites where relocating the special artifacts took as much as 365 extra days (yes - a whole year)!

In addition, there are usually unanticipated costs that include fines and other things. The accounting departments suggest that we model those costs with an exponential distribution with mean (scale) $100,000.

Run a second simulation with these new parameters and using at least 1000 iterations.

Put your simulation code in the cell below.

*** 8 points - answer in cell below *** (don't delete this cell)

In [10]:
```
from ortools.sat.python import cp_model
import numpy as np

task_params_dict = {
    'A': (14, 29, 386),
    'B': (14, 21, 56),
    'C': (42, 63, 126),
    'D': (28, 35, 70),
    'E': (7, 28, 35),
    'F': (28, 35, 70),
    'G': (35, 42, 77),
    'H': (35, 56, 119),
    'I': (21, 49, 63),
    'J': (21, 63, 63),
    'K': (21, 28, 28),
    'L': (7, 35, 49),
    'M': (7, 14, 21),
    'N': (35, 35, 63)
}

task_names = list(task_params_dict.keys())
num_tasks = len(task_names)



precedence_dict = {
    'A': ['B'],
    'B': ['C'],
    'C': ['D', 'E', 'I'],
    'D': ['G'],
    'E': ['F', 'H'],
    'F': ['J'],
    'G': ['H'],
    'H': ['M'],
    'I': ['J'],
    'J': ['K', 'L'],
    'K': ['N'],
    'L': ['N']
}

simSize = 1000
daysTaken = np.zeros(simSize)
artiCost = np.zeros(simSize)

for i in range(simSize):
    task_duration_dict = {
        t: round(
            np.random.triangular(task_params_dict[t][0],
task_params_dict[t][1], task_params_dict[t][2]))
        for t in task_names
    }
    artiCost[i] = np.random.exponential(scale = 100000, size=1)
    durations  = list(task_duration_dict.values())
    task_name_to_number_dict = dict(zip(task_names, np.arange(0,
num_tasks)))

    horizon = int(sum(task_duration_dict.values()))
    model = cp_model.CpModel()

    start_vars = [
        model.NewIntVar(0, horizon, name=f'start_{t}') for t in
task_names
    ]

    end_vars = [model.NewIntVar(0, horizon, name=f' end_{t}') for t in
task_names]

    intervals = [
        model.NewIntervalVar(start_vars[j],
                             durations[j],
                             end_vars[j],
                             name=f'interval_{task_names[j]}')
        for j in range(num_tasks)
    ]
    for before in list(precedence_dict.keys()):
        for after in precedence_dict[before]:
            before_index = task_name_to_number_dict[before]
            after_index = task_name_to_number_dict[after]
            model.Add(end_vars[before_index] <=
start_vars[after_index])

    obj_var = model.NewIntVar(0, horizon, 'largest_end_time')
    model.AddMaxEquality(obj_var, end_vars)
    model.Minimize(obj_var)

    solver = cp_model.CpSolver()
    status = solver.Solve(model)

    daysTaken[i] = solver.ObjectiveValue()
```

What is the probability of meeting the Under 280, 280-329 or over 329 cutoff points now? What's the average profit now?

Include code to answer these questions with output below:

*** 2 points - answer in cell below *** (don't delete this cell)

In [11]:
```
p_low = np.sum(daysTaken <= 280)/ simSize
p_high = np.sum(daysTaken >= 329) / simSize
p_mid = 1 - (p_low + p_high)
print(f"The probability that construction would take less than 280
days is: {p_low:.3f}")
print(f"The probability that construction would take between 281 days
and 329 days is: {p_mid:.3f}")
print(f"The probability that construction would take more than 329
days is : {p_high:.3f}")

profit = np.zeros(simSize)
for i in range(len(daysTaken)):
    artiCost[i] = artiCost[i]*0.000001
    if daysTaken[i] <= 280:
        profit[i] = 5.55 - artiCost[i]
    elif daysTaken[i] >= 329:
        oops = daysTaken[i] - 329
        profit[i] = 5.4 - (oops * 0.025) - artiCost[i]
    else:
        profit[i] = 5.4 - artiCost[i]
meanProfit = np.mean(profit)
print(f"The average profit is: ${meanProfit:.2f} million")
```

Out[11]:

```
The probability that construction would take less than 280 days is:
0.000
The probability that construction would take between 281 days and 329
days is: 0.045
The probability that construction would take more than 329 days is :
0.955
The average profit is: $2.19 million
```

P3.3 - Make Decision about Insurance

Clearly dealing with artifacts can be very costly for Reliable Construction. It is known from past experience that about 30% of building sites in this area contain special artifacts. Fortunately, they can purchase an insurance policy - a quite expensive insurance policy. The insurance policy costs $500000, but it covers all fines and penalities for delays in the event that special artifacts are found that require remediation. Effectively, this means that Reliable could expect the same profit they would get if no artifacts were found (minus the cost of the policy).

Given the estimated profit without artifacts, the estimated profit with artifacts, the cost of insurance, the 30% likelihood of finding artifacts, create a payoff table and use Baye's Decision Rule to determine what decision Reliable should make. You should round the simulated profits to the nearest $100,000 and use units of millions of dollars so that, for example, $8,675,309 is 8.7 million dollars.

Provide appropriate evidence for the best decision such as a payoff table or picture of a suitable (small) decision tree.

*** 6 points - answer in cell below *** (don't delete this cell)

	State of Nature	
Alternative	Artifacts	No Artifacts
Insurance	4.74	4.74
No Insurance	2.19	5.24
Prior Prob.	0.3	0.7

$$4.74(0.3) + 4.74(0.7) = \$4.74 million$$

$2.19(0.3) + 5.24(0.7) = $4.33 million$

Describe, in words, the best decision and the reason for that decision:

*** 2 points - answer in cell below *** (don't delete this cell)

Bayes rule shows that buying the insurance is a better idea for Reliable than not. Even though it costs $500000 for the policy, the amount of profit lost to the discovery of artifacts is more than enough to make the initial loss payoff even if artifacts aren't discovered.

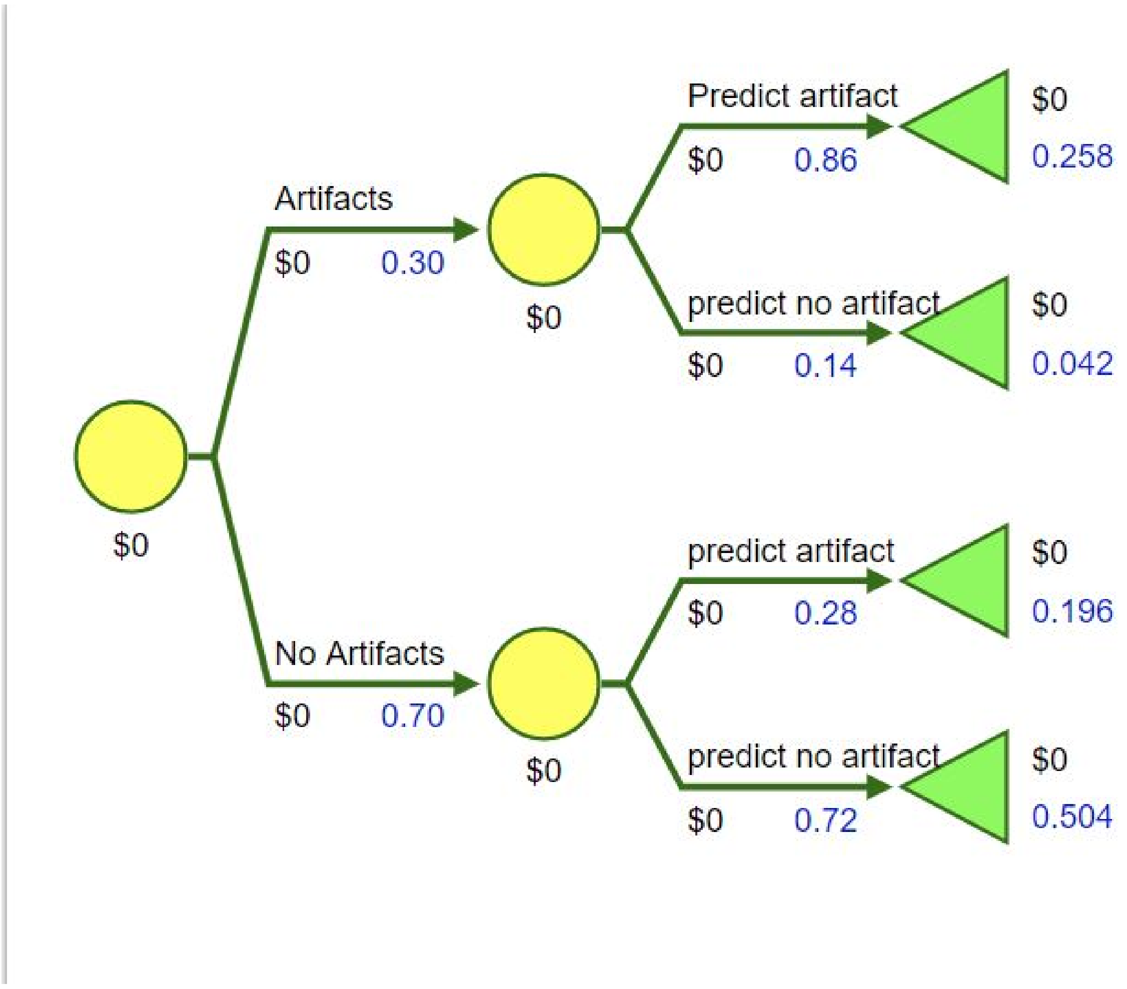
P3.4 - Posterior Probabilities

Reliable has been contacted by an archeological consulting firm. They assess sites and predict whether special artifacts are present. They have a pretty solid track record of being right when there are artifacts present - they get it right about 86% of the time. Their track record is less great when there are no artifacts - they're right about 72% of the time.

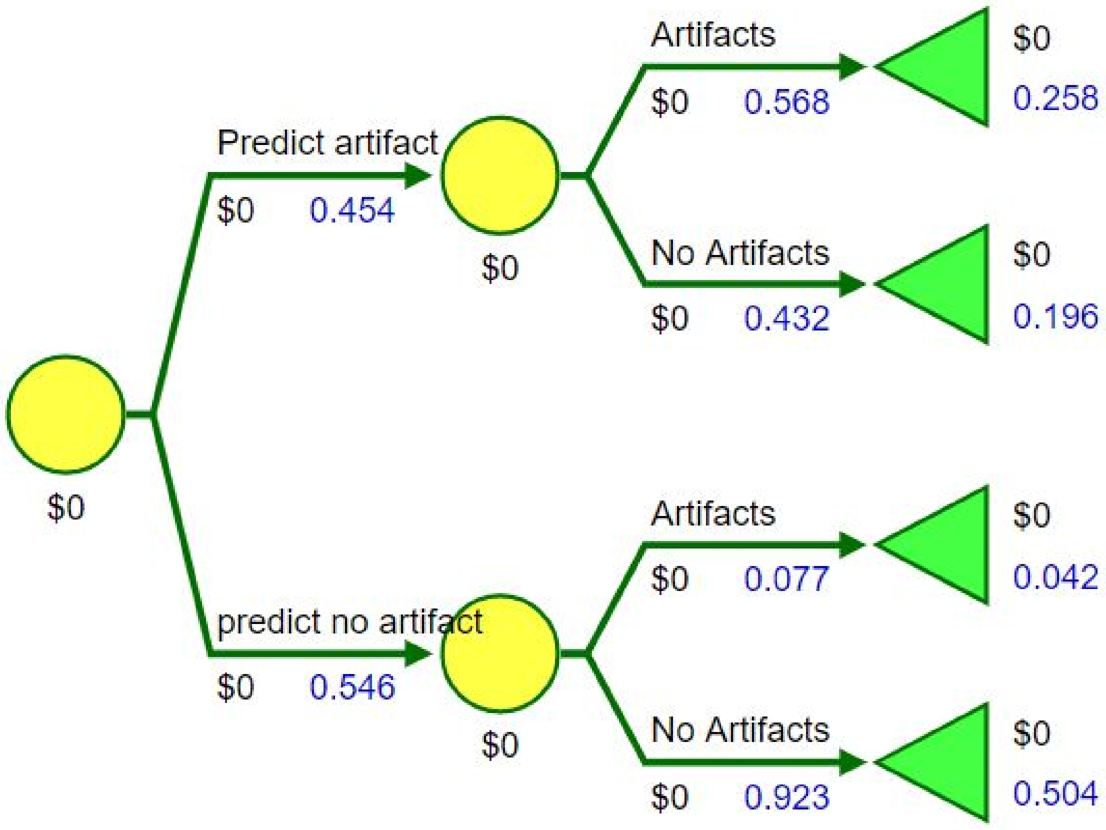
First find the posterior probabilities and provide evidence for how you got them (Silver Decisions screenshot or ?).

*** 6 points - answer in cell below *** (don't delete this cell)

Tree constructed using prior probabilities



Tree flipped to find posterior probabilities

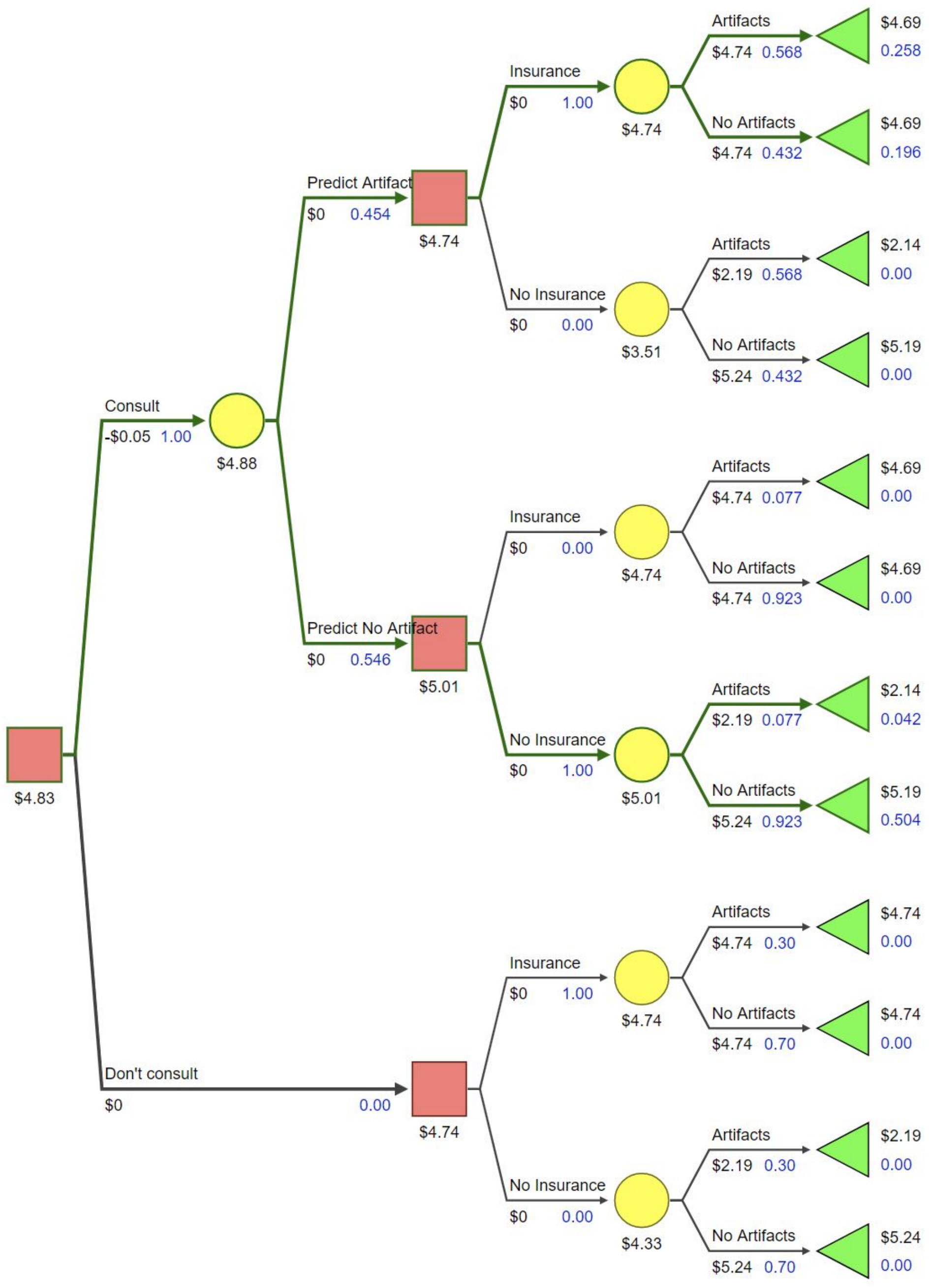


The consulting fee for the site in question is $50,000.

Construct a decision tree to help Reliable decide if they should hire the consulting firm or not and if they should buy insurance or not. Again, you should round the simulated profits to the nearest $100,000 and use units of millions of dollars (e.g. 3.8 million dollars) in your decision tree.

Include a picture of the tree exported from Silver Decisions.

*** 10 points - answer in cell below *** (don't delete this cell)



Summarize the optimal policy in words here:

*** 2 points - answer in cell below *** (don't delete this cell)

It looks like according to the tree that we want to get insurance if the consulting firm predicts if there are artifacts and not get insurance if the consulting firm predicts there will be not artifacts. I'm not sure if that's how insurance works but that's what the tree says to do

P3.5 - Final Steps

How confident do you feel about the results of your decision analysis? If you were being paid to complete this analysis, what further steps might you take to increase your confidence in your results?

*** 4 points - answer in cell below *** (don't delete this cell)

I feel confident that this decision analysis will do a pretty good job of making Reliable some money. I think I would improve this by getting a second (or even third) consultation. When we're talking money on this scale, $50,000 is that much to spend and the profits would still be higher getting the insurance and another consulatation than the risk in pay we would have if we were to find artifacts without insurance.

```
In [0]:
```