Decision Trees

Sample Question

Riya is in her second semester in the USC business analytics program. She has been applying to summer internships and has received an offer from a data analyst position at Disney. However, she has to respond to this offer in one week, while she has an interview at Google in a month, which will be after her Disney deadline. While Google is her dream company, she may not get an offer from them, and Disney is a sure place where she can sharpen her data analytics skills for the summer. She has to decide whether or not to take the Disney offer.

If she rejects the Disney internship, then she estimates there's a 50% chance she will succeed in the Google interview and get an internship, and a 50% chance she will not, at which point she may have to settle for a second-tier company to do an internship. She estimates that her utility (i.e. level of happiness) for the summer internship at Google is 10, her utility for the internship at Disney is 8, and her utility for second-tier companies is normalized to 0.

However, she does not only care about her utility for the summer, but also for after graduation. In other words, **her total utility (on which she is basing the current decision) is the sum of her summer utility plus her post-graduation utility.** Currently Google is her dream company for full time job, and she estimates that there's a 80% chance she is a good fit for the company, for which her post-graduation utility would be 100 if she works there. If she is not a good fit, then her post-graduation utility is -50 for working there (as she would have to forego other full-time opportunities). One benefit of doing a Google internship this summer is to figure out whether she is a good fit. She estimates that if she did an internship at Google this summer, she would know for certain whether or not she is a good fit there, so is able to better make the post-graduation application decision.

Her dilemma is as follows. If she takes the internship at Disney, then she would get a summer utility of 8, plus she still plans to apply for a full-time position at Google later. Her experience at Disney would give her a 60% chance of getting the full time offer at Google (but she doesn't know whether she would be a good fit so would have to bear the above risk for her post-graduation utility). If she applies for a Google full time position and does not get in, then her post-graduation utility would be 0.

If she rejects the internship at Disney, then if she gets into Google, she will obtain the summer utility of 10, plus the knowledge of whether she is a good fit for the company. If she is a good fit, then she is sure she would get the full time position, and so obtain a post-graduation utility of 100. If she is not a good fit, then she has time to prepare for other companies, and get a post-graduation utility of 20. If she rejects Disney and doesn't get the Google internship, then she gets a summer utility of zero, after which she will reapply for Google. She estimates the chance of getting a Google full-time offer after a second-tier summer internship would be 40%. If she gets in, she bears the aforementioned risk on her post-graduation utility (based on whether she is a good fit for Google). If she does not get in, she obtains a post-graduation utility of zero.

a) Draw a decision tree to capture the uncertainties and sequential decisions described above. (Use rectangles for decision nodes, ovals for event nodes, and blank for outcome nodes.) (10 points)

b) Write the values of each outcomes and probabilities for each event node on the tree. (10 points)

c) Solve the decision tree and write down her expected utility for accepting and rejecting the Disney internship respectively. (5 points)

Other Sample Questions (from DMD book)

Problems from the textbook of similar level of complexity as the decision tree problem in the exam.

- DMD Exercise 1.1 b) (Mary's optimal strategy with forecast)
- DMD Exercise 1.4 (from Homework 2)
- DMD Exercise 1.6 (from Homework 2)

Coding (Simulations)

Sample Question 1

Part 1. (10 points) Predict the output of the following code, which is similar to the RM2 policy in Lab 2.

```
[ ]: data=np.array([100,70,50,30,200])
     order=np.array([4,0,1,2,3])
     price=60
     left=3
     tot=0
     for element in data[order.argsort()]:
         if element>=price:
             tot+=price
             left-=1
         if left==2:
             price=100
         elif left==1:
             price=150
         elif left==0:
             break
     print(tot)
```

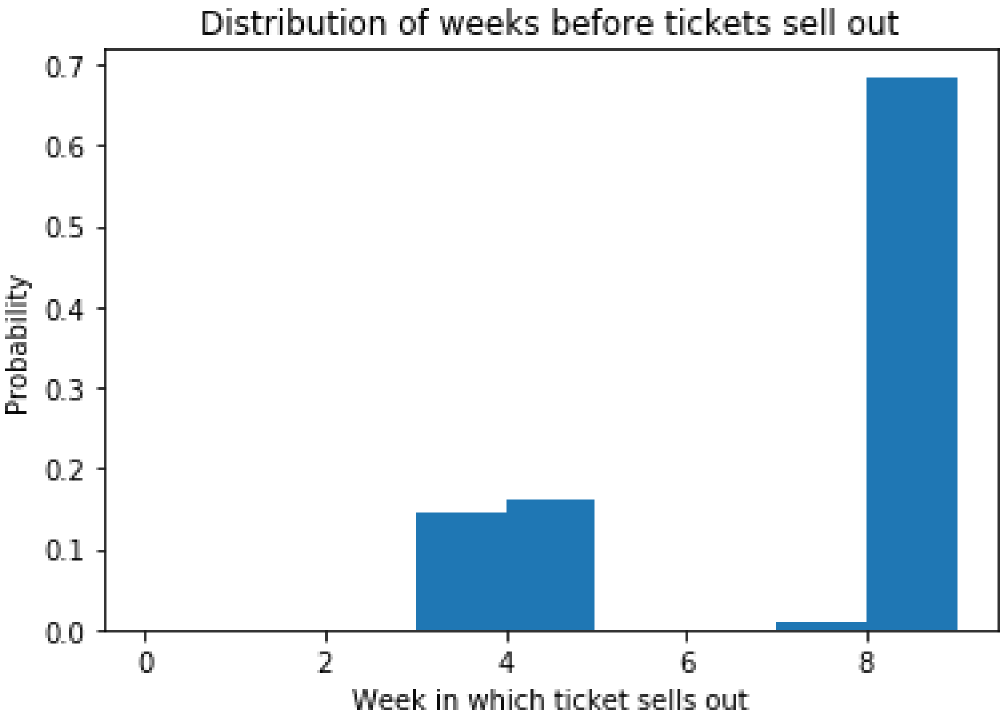
Part 2. (15 points) In this question, you will write code to complete an analysis of the distribution of times before the tickets for a certain event sells out. USC is selling 400 tickets to a special event. The selling starts 8 weeks before the event takes place. Demand for any particular event is uncertain:

- With probability 0.7, the event is of average popularity. In this case, the marketing team predicts that the weekly demand will distributed as a Poisson random variable with mean 45, with each week independent of the next. (However, all 8 weeks will have this mean.)
- With probability 0.3, the event is very popular. In this case, the mean of the Poisson random variable will be 100 each week (for the duration of 8 weeks).

Complete the functions `generateScenario()` and `simulateScenario()` to complete the following analysis on the distribution of the number X of weeks before the ticket sells out, starting the index from 0. (If the ticket sells out in the first week, then $X = 0$. If it sells out in the second week, then $X = 1$. If it does not sell out in all 8 weeks, then for convenience we say that $X = 8$.) Read the docstrings for what the input and output should be. The output from running 1000 simulations is attached.

```
[ ]: from scipy.stats import bernoulli,poisson
     import numpy as np

     # Write your code only in the following block
     # BEGINNING OF CODE EDIT BLOCK---------------
     def generateScenario():
         ''' Function that returns a numpy array of size 8, with the
         ith element corresponding to the realized demand in each week'''



     def simulateScenario(data,inventory):
         ''' Function with two inputs:
                data: the numpy array of 8 elements that you generate above.
                inventory: an integer corresponding to how many tickets.
                           In this problem it should be 400.
             Returns: the index of the week when tickets sell out. (Start index from 0)
                      For example, if inventory is 400, and the demand in week 0 already
                      is at least 400, then the return value should be 0.
                      If the demand in week 0 is 200 and in week 1 is 300, then the return
                      value should be 1. (smallest return value possible)
                      If the sum of demand in all 8 weeks is less than 400, then the return
                      value should be 8.'''



     # END OF BLOCK--------------

     inventory=400
     dataset=[generateScenario() for i in range(1000)]
     values=[simulateScenario(data,inventory) for data in dataset]
     import matplotlib.pyplot as plt
     plt.hist(values,bins=range(10),density=True)
     plt.title('Distribution of weeks before tickets sell out')
     plt.ylabel('Probability')
     plt.xlabel('Week in which ticket sells out')
     plt.show()
```

```
[59]: # Expected output when complete.
```



Sample Question 2

Part 1. (10 points) Predict the output of the following code

```
[ ]: import numpy as np

     def f(x):
         x[1:]=x[:-1]
         x[0]=0
         s=np.sum(x)
         x[0]=max(0,15-s)
         return x

     print(f(np.array([3,8,2]))[0])
```

Part 2. (15 points) This question asks you to simulate the operations of a hospital ward. You are to write a function simulate(data,beds,stay)

The input parameter data is a list containing the number of requests for beds for each day. For example, data=[6,10,3,8,4,6] denotes that 6 patients requested beds on first day, 10 on the second day, and 3 on the third day, and so on. The parameter beds is the total number of beds in this ward. In the example below, there are 15 beds in total. stay is the number of days each patient stays in the ward. In the example below, stay=3, in which case you can assume that every patient stays for exactly 3 days and leave in the early morning on the 4th day since arrival.

Initially the beds are all empty. As patients come in, beds are assigned from a first come first serve basis. When a patient requests a bed and none is available, then that patient is rejected from the ward. The return value of the function is the total number of rejected patients.

In the numerical example above, the number of beds available is first 15. Then after day 1, it becomes 15-6=9. After day 2, all beds are taken and 10-9=1 patient is rejected. On day 3, all 3 patients are rejected because the patients from day 1 haven't left yet. On day 4, all 6 admitted patients from day 1 leave in the early morning and 6 beds are emptied up. Hence, 6 out of the 8 patients who come on day 4 get beds, and 2 are rejected. On day 5, the 9 patients from day 2 all leave, and all 4 patients who come in get beds, and there are now 5 empty beds. On day 6, no patient leaves because on day 3 none were admitted, and there are 5 empty beds, so 1 patient gets rejected.

Complete the function below.

Hint:

- You only need to write code in the region bounded by # BEGIN ------ and # END -----. Although you are free to insert code anywhere else.
- You can use the variable pipeline to store the number of admitted patients in the last 3 days. (if stay=3.) See part 1 for sample code of how to update pipeline.
- You only have to calculate the number of patients admitted in each period, as well as update the pipeline variable in the loop.

```
[ ]: import numpy as np

     def simulate(data,beds,stay):
         pipeline=np.zeros(stay,dtype=int)
         totRejected=0
         day=1

         for demand in data:
             # BEGIN ------------------
```

```
            # END ---------------------------
            rejected=demand-admitted
            print('On day {0}, {1} incoming patient(s) is rejected.'.format(day,rejected))
            totRejected+=rejected
            day+=1
        return totRejected

    data=[6,10,3,8,4,6]
    stay=3
    beds=15
    ans=simulate(data,beds,stay)
    print('Total number of rejected patients is', ans)
```

```
[13]: # Expected output after code is completed.
```

```
On day 1, 0 incoming patient(s) is rejected.
On day 2, 1 incoming patient(s) is rejected.
On day 3, 3 incoming patient(s) is rejected.
On day 4, 2 incoming patient(s) is rejected.
On day 5, 0 incoming patient(s) is rejected.
On day 6, 1 incoming patient(s) is rejected.
Total number of rejected patients is 7
```

Sample Question 3

Part 1: (10 points) Predict the output of the following code.

```
[ ]: def f(x):
         return 3*x

     x=np.array([3,6,2,10])
     y=np.array([5,4,2,9])
     a=np.sum(x-y)
     b=np.sum(x>y)
     c=np.sum(x<y)
     d=len(x)
     e=np.sum(np.maximum(x-y,0))
     l=[f(x) for x in [a,b,c,d,e]]
     print(l)
```

Part 2: (15 points) This problem asks you to write a function to compare the sequence of daily profits from two products, say A and B. (This code can be used to supplement the analysis from Homework 5 comparing the profit from selling at Rockport and Glouceter for example.) The inputs to the function `simulateScenario` are

- `profitA`: a 1-D numpy array of daily profits from product A.
- `profitB`: a 1-D numpy array of corresponding daily profits from product B.

There are four outputs you are to calculate:

- `prob`: the proportion of days for which the profit of product A is strictly bigger than that of B.
- `avgDiff`: the average difference in profit between A and B. (Average profit of A minus the average profit of B).
- `stdDiff`: the standard deviation of the daily difference in profits between A and B.
- `avgGain`: on the days when the profit for product A is larger than B, the average amount it is larger by. (If the daily profits for A is always less than or equal to B, then you should return `np.nan`.)

You only need to write the code within the `# BEGIN-----------` and `# END -----------` block. (Hint: the majority of commands you need are given in part 1.) You will need to define the four variables within the block and assign them the correct value.

```
[ ]: import numpy as np

     def simulateScenario(profitA,profitB):

         # BEGIN ------------------------



         # END ----------------------------

         return prob,avgDiff,stdDiff,avgGain

     profitA=np.array([10,5,8,4,6])
     profitB=np.array([11,8,3,7,6])
     prob,avgDiff,stdDiff,avgGainCondBigger=simulateScenario(profitA,profitB)

     print('The profit for A is strictly larger than B with probability {0:.0%}'.format(prob)
     print('The expected profit for A minus the expected profit for B is',avgDiff)
     print('The standard deviation in the difference in expected profit is {0:.2f}'.format(st
     print('On days when profit for A is larger, it is larger on average by',avgGainCondBigge
```

```
[58]: # Expected output when complete
```

```
The profit for A is strictly larger than B with probability 20%
The expected profit for A minus the expected profit for B is -0.4
The standard deviation in the difference in expected profit is 2.94
On days when profit for A is larger, it is larger on average by 5.0
```