4 Inventory Control

Lesson objectives:

- Use bootstrapping/simulation methods to estimate the probability of an event.

Background

The Port of Oakland operates on almost 500 hectares of space and is one of the largest ports on the west coast of the United States. Almost all the business at the port is done via containerization methods: 40 and 20 foot containers are put directly from ships onto trucks and rail. Despite the efficiency of this transportation method, some freight is unpackaged and repackaged onto other transportation methods at the port.

This type of loose freight is difficult to deal with as the port operators need to decide when and how to ship it. The port operators are most efficient when trucks are operating at capacity, but with different size loose materials coming in to different destinations maintaining efficiency is difficult. Table A.1 shows a partial day manifest for Tuesday, December 8th.

Name	Destination	Length (feet)
Chu Household Good	San Francisco, CA	28
S.R. Batteries	New York, NY	12
Desmo Parts	Santa Monica, CA	3.5
MIA BSA Tents	Miami, FL	8
Bat bamboo	Louisville, KY	12
Chalk	San Francisco, CA	22
D. Handbags	Oklahoma City, Oklahoma	8.2

Table A.1: Loose freight Manifest, Tuesday, December 8th, 2015

Table A.1 demonstrates a few interesting features regarding loose freight. First, while the items don't fill a full 20- or 40-foot container, the contents have already been measured and packaged in order to easily determine how they fit into the standard shipping containers. While the company doing the shipping would prefer to have all trucks 100% full of materials, they pay a holding cost for keeping the material in the port's factories. They also incur significant consumer wrath if the freight does not ship in a reasonable amount of time. The port's goal is to send out all shipments in less than seven days, but some packages lay in wait for over two weeks.

The port currently uses software, written in the early 1980's, that attempts to maximize revenue using algorithms that were, at the time, cutting edge. The way that this system currently works is by leveraging the company's historical shipping record to try to anticipate the likelihood of a truck being full. As the likelihood of freight decreases the system becomes more likely to send out the truck; though there are edge cases that are handled outside of this framework. In rare cases, the freight will be handled by either a small truck to go to a location in a one-off fashion or the shipment will be repackaged for non-truck shipment. For example, during winter, trucks sent to Canadian cities have to be verified as having accessible roads.

Ignoring these edge case issues, the problem of maximizing the revenue of this process is called a "random knapsack" problem. The standard knapsack problem involves attempting to fill a knapsack that can only hold weight ω with a set of objects with weights $w_1, w_2, ...w_n$ while minimizing the amount of dead space. For example, if a knapsack can hold 10 lbs and there are objects of weights 7, 4, 2 and 2 then choosing 7 and 2 has dead space of 1 lb., while choosing objects 4,2 and 2 has dead space of 2 lbs. The reason that this is a "random" knapsack problem is because future weights are unknown.

A new, silicon-valley startup is attempting to disrupt this industry, it's called XXX[2]. Ignoring the efficacy of the knapsack solution that the XXX's software uses, there are a number of desirable features such as sending TXT notifications, an internet enabled interface and a mobile app. The company has also offered to let the port use the software on a trial basis for a few days before committing. As a data analyst at the port, it is up to you and your team to decide if using this software is worth it.

Fill Level (Feet)	Percent of trucks (%)
< 10	8
10-15	27
15-20	10
20-25	11
25-30	15
30-35	20
35-37	7
37+	2

Table A.2: Distribution of fills, in 40 feet containers, last 6 months

To evaluate this software, you will study the impact of the new software on the proportion of trucks that have between 35-37 feet filled and those greater than 37 feet filled. Because of the nature of the business, moving those numbers even a small amount has a large effect

[2] If someone can think of a clever name, +1 bonus point

on the number of trucks that the company sends out. Currently, as table A.1 demonstrates, only 2% of trucks are more than 37 feet full and only 7% of trucks are between 35 and 37 feet filled. This new software company claims that it can move these numbers in a significant fashion.

At the main port locations, around 5,000 trucks are sent out each day with loose fill. Because of the cost of testing the new system, the port administrators have decided to only run the test for a single day. Since the new software works on a TXT message based system instead of the current system which uses walkie-talkies, the port was required to hire a number of temp works to convert the walkie-talkie messages to TXT messages in order to send them to the drivers.

Fill Level (Feet)	Number of trucks
< 10	452
10-15	1,212
15-20	625
20-25	653
25-30	713
30-35	858
35-37	368
37+	108
Total	4,982

Table A.3: XXX Test Fill Distribution

The test ran on an otherwise average Friday, with 4,982 trucks leaving the port. Luck was on your side the day of the test and, despite having to send out over 15K TXT messages, all communication went though without issue. Table A.3 shows the distribution of containers on the test day. On the day after the test, Table A.3 was tabulated, showing the results of the test.

Jamie, the CEO of XXX declared the test a success:

> Look at that! 476 trucks left with more than 35 feet full, which is almost 10%. This is a great demonstration of the power of our software and the efficacy of its new algorithm. Feel free to go over the data, but I'm sure that you'll come to the same conclusion that we have: this software is the best.

You and your team dug through the numbers and found nothing factually incorrect with them, but you have your doubts on the value of their software. Talking it over with the port director, it turns out that doing a full conversion to this new technology would not only be quite costly, but would also lock them into using this system. If, as Jamie says, this is really increasing the efficiency of the loose freight process, then everyone agrees that the fixed transfer cost would be tiny in comparison to even the value of small bump in freight efficiency.

In order to evaluate that statistical significance of Jamie's the test you've decided to run some Monte-Carlo simulations and do some bootstrapping. To simplify this problem, make the assumption that the fill percentage of a truck is independent of the fill percentage of other trucks. In other words, if you tell me that the truck leaving the station now is at 22.5 feet, this doesn't tell me anything about the next truck's level.

1. Write a function to simulate trucks, trucksim, which takes as input the following:
   - A number, n, of trucks to simulate
   - A dictionary, d, of value-probability pairs which represent the distribution of fill rates. For the case above it would look like { "<10" : .08, "10-15" : .27, ... }

   The output of this function should be a list n different truck levels, randomly simulated. For example, if $n = 1$ and $d = \{\text{“} < 10\text{”} : .5, \text{“} > 10\text{”} : .5\}$, then the result will either be $[\text{“} < 10\text{”}]$ or $[\text{“} > 10\text{”}]$.

2. Using the function above, write a wrapper function to bootstrap confidence intervals. In particular, write a function called truck1CI, that takes the following arguments:
   - A string ("keyname"), the name of a single truck level ("10-15", "15-20", ..)
   - A simulated list (simlist)
   - the number of times to bootstrap (n)
   - α, which is equal to 1 - the confidence interval level. So, if we want a 90% Confidence interval, α will be equal to .1.

   The function should create n samples (or bootstraps) which have the same length as simlist and are created by sampling, with replacement from simlist. In other words, the function create n new samples from the original simlist. The function should calculate the percentage of each of these n lists that are equal to "keyname". Finally, truck1C1 should return the mean, $1 - \frac{\alpha}{2}$, $\frac{\alpha}{2}$ percentiles of this percentage and the standard deviation of the bootstrapped means.

   The last item returned by the function, the standard deviation of the bootstrapped mean, represents the standard error on our estimator (in this case, the sample mean).

   Please answer the following questions, using a single truck simulation of length 1,000 as input and using the information from table A.2:

   (a) Fill out the following table:

Confid. Level	Truck Fill Level	Number of Bootstraps	Est. Value	CI Lower Bound	CI Upper Bound	Std. Error
90%	30-35	100				
90%	30-35	1,000				
90%	30-35	2,500				
90%	30-35	5,000				
90%	< 10	100				
90%	< 10	1,000				
90%	< 10	2,500				
90%	< 10	5,000				

(b) Did this behave as expected? In particular, comment on the range of the confidence interval and the size of the standard error as the number of bootstraps increases.

3. Using the same process, use the original data (A.2) to build confidence intervals around the percentage of trucks for each type you would expect if 4,982 trucks went out.

Confid. Level	Truck Fill Level	Number of Bootstraps	Est. Value	CI Lower Bound	CI Upper Bound	Std. Error
95%	35-37	200				
95%	37+	200				
90%	35-37	200				
90%	37+	200				

After completing the table above, what results do you draw about XXX's algorithm?

4. Let's now compute the likelihood that XXX's algorithm is better than than the current algorithm by using a Monte-Carlo algorithm. To do this repeat the following 1,000 times:

- Using trucksim, create a simulated list of 4,982 trucks under the null hypothesis that there is no difference.
- Calculate the number of trucks that that have between 35-37 feet filled. If this is greater than or equal to 368, count it as a "1", otherwise as a "0". Sum up the list of 1's and 0's and divide it by 1,000. This represents the likelihood of getting data as or more extreme than what was observed under the null hypothesis (sound familiar?).
- What is your estimated probability?
- Repeat the analysis on the trucks with length 37+.

5. Repeat the same procedure, but this time doing a joint test of both hypothesis (having more extreme values for BOTH 37+ and 35-37). What is the relationship between the tests?

6. What are the strengths and weaknesses of this method? What do you conclude about XXX's algorithm?

7. To turn in:

   - Electronic submission of your code. The code should be in text file format using standard libraries (numpy, scipy, etc.). It will be run from the command line. If it fails to run or errors out your grade will be negatively effected.

   - A write up, less than two pages, answering the questions above, including filled-in tables. It should also present a recommendation as to accepting or rejecting XXX's product.