Georgia Tech College of Engineering
H. Milton Stewart School of
Industrial and Systems Engineering

ISyE 6202 Supply Chain Facilities

Casework 1.1

FruitSoul Demand and Capacity Modeling Under Full Clairvoyance

Guillaume Eymery - Louise Lacroix - Vaishnavi Vaidya

H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

Contents

List of Figures

List of Tables

TASK 1

Task 1.1 to task 1.4

For Task 1.1 to task 1.4, we are asked to provide key insights and visual representation for earliest, latest, preferred delivery date as well as the demand request history. To automate this process for each product and volume, we will be executing a Python Script that will filter the data and compute the pivot table for any product. It will then provide a graph, showing the desired data.

Here is an extract of the code with explanations for some lines:

```
def get_data(table_main, mix, volume):
    # Definition of the function, choosing the product family as well as the volume
    df = table_main[['Quantity', 'Earliest Delivery Date', 'Preferred Delivery Date',
                     'Latest Delivery Date', 'Demand Request Date', 'Mix',
                     'Jar Size (vol. ounces)']]

    # Filtering the desired Data
    filtered_df = df[df['Mix'].str.contains(mix)]
    filtered_df = filtered_df[filtered_df['Jar Size (vol. ounces)'].astype(str).str.contains(str(volume))]

    # Transforming the dates in a format that can be read by Python
    filtered_df['Earliest Delivery Date'] = pd.to_datetime(filtered_df['Earliest Delivery Date'])
    filtered_df['Latest Delivery Date'] = pd.to_datetime(filtered_df['Latest Delivery Date'])
    filtered_df['Preferred Delivery Date'] = pd.to_datetime(filtered_df['Preferred Delivery Date'])
    filtered_df['Demand Request Date'] = pd.to_datetime(filtered_df['Demand Request Date'])

    # Creating the pivot tables
```

```
    grouped = filtered_df.groupby('Latest Delivery
        Date')['Quantity'].sum() # Summing all the quantities
    grouped1 = filtered_df.groupby('Earliest Delivery
        Date')['Quantity'].sum()
    grouped2 = filtered_df.groupby('Preferred Delivery
        Date')['Quantity'].sum()
    grouped4 = filtered_df.groupby('Demand Request
        Date')['Quantity'].sum()
```

We can than use the Python Library Plotly, that has great features in data manipulation and visualization. The results are shown in the Browser in a way that can be easily understandable. To make the code more user-friendly, we incorporated a Tkinter user interface. Before running the code, the reader must make sure the previous python Libraries are already installed, if not, please type the following code in the console:

```
pip install Tkinter
pip install Plotly
pip install Pandas # usually already installed by default
```

The results should appear as such:

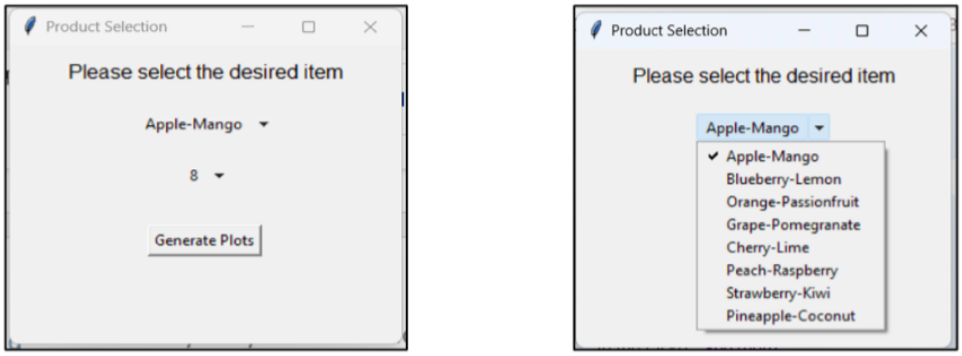


Figure 1: Tkinter User-friendly interface

For example, in the case of the Apple Mango, 8 ounces:

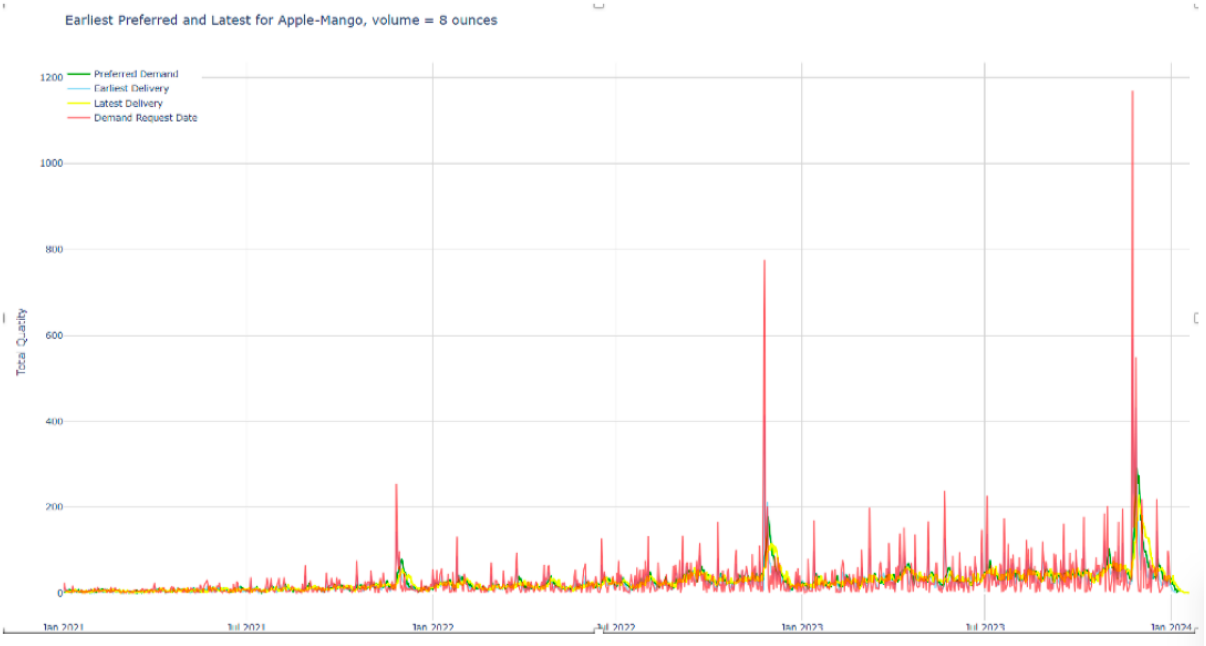


Figure 2: Apple Mango, Demand Request Date, Earliest, Latest and Preferred Delivery Date

In the case where the reader wants to hide a curve, he/she may click on the label on the upper left of the figure. Tools are available if he/she wants to zoom in the graph.

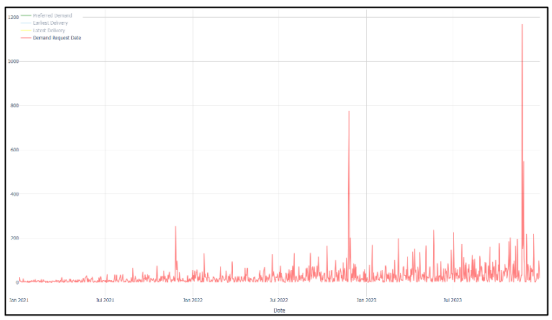


Figure 3: Demand Request Date, Apple Mango 8 ounces

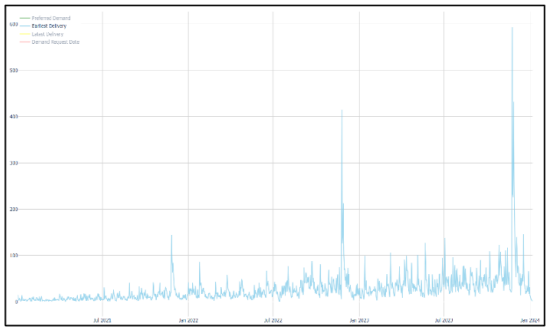


Figure 4: Earliest Delivery Date, Apple Mango 8 ounces

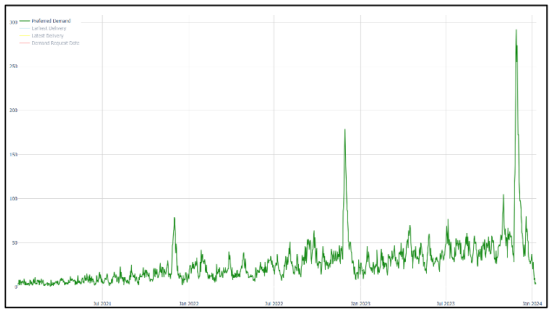


Figure 5: Preferred Delivery Date, Apple Mango 8 ounces

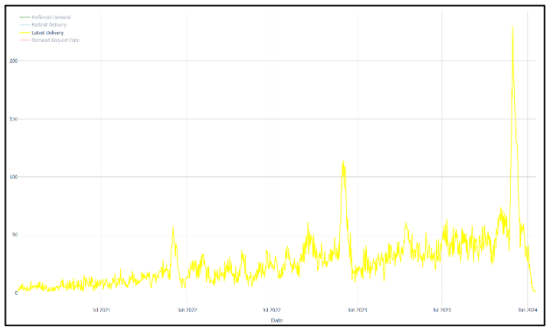


Figure 6: Latest Delivery Date, Apple Mango 8 ounces

The first element that strikes the eye are the three peaks, appearing in every graph, each year, for each product. These peaks happen every year during the thanksgiving period. We can also see that the Demand Request Date has a high volatility compared to the other graphs, this is because customers order on very specific days. On the opposite, the Latest Delivery Date graph (in yellow) seems to be smoother than the others, this characteristic will later be taken in consideration for the smoothing algorithm (task 1.5, task 6, task 7...)

Task 1.5

For this part, we are asked to smooth the demand so that every customer gets the product he/she ordered at the right time, while diminishing the volatility of the demand for the supplier.

In this illustrative example, we will focus on the Apple-Mango Mix (all sizes merged). However, a Python Code relative to this specific task is accessible in the folder. With this code, and using the Tkinter interface provided, the reader can access the results for every other product (for all sizes).

The main idea for the smoothing is to try and allocate the same number of deliveries for each day, will satisfying the Earliest and Latest Delivery Date for the customer.

As first approach, let us imagine that the orders are either all delivered during the earliest or the latest possible delivery date. This should give a first overview of the smoothing objective.

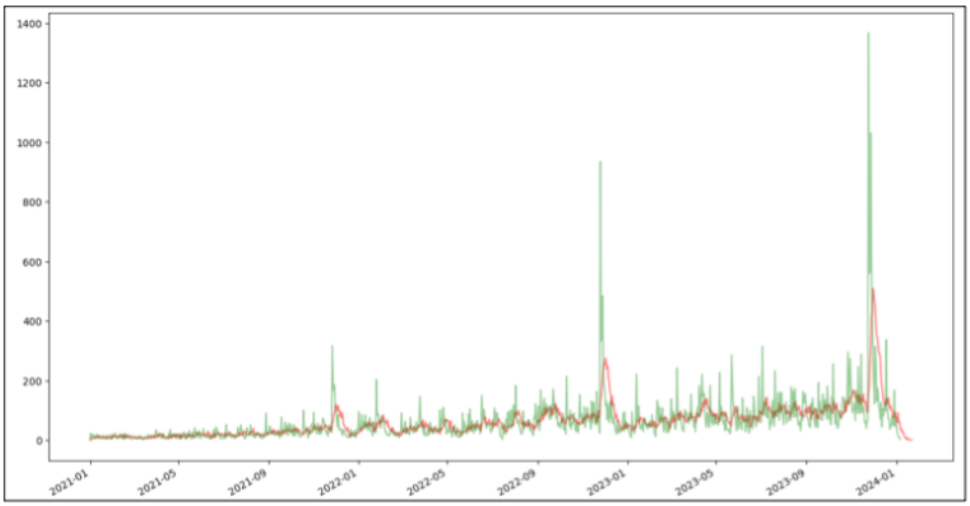


Figure 7: Evolution of the Demand Earliest (green) vs Latest (red) delivery date - Apple Mango

We can see that the Latest Delivery Date (red) is much smoother than the Earliest Delivery Date (green). The objective is now to smoothen even more this curve, it can be achieved by spreading uniformly the demand over the days.

To do so, we will use the following strategy, that we will later on compute in a pro-

gram for all the different orders.

Let us consider the following order:

Order Number	Quantity	Earliest Delivery Date	Latest Delivery Date	Mix
#XXXXXX	1	2021-01-06	2021-01-10	Apple-Mango

Table 1: Example Order for Task 1.5

We will try to deliver this order between the Earliest and Latest acceptable Delivery Date.

To do so, we may have a look in the delivery log.

Day	Quantity
2021-01-06	4
2021-01-07	8
2021-01-08	4
2021-01-09	1
2021-01-10	9

Table 2: Example of Delivery Log

We can see that the less busy day for the moment is 2021-01-09. Therefore, we will assign this day for the delivery date for this order and move on to the next order.
We can generalize this idea:
Let DD represent the delivery date for an order 'i', EDD and LDD respectively the Earliest and Latest delivery Dates:

$$DD_i = \min(Quantity[EDD_i : LDD_i])$$

This method assigns an order to the least busy delivery day and should enable a better smoothening.

Under the circumstance that the minimum is attained in two distinct days, we will choose the latest of both. This is based under the intuition that the LDD demand curve is smoother than the EDD curve. Indeed, it helps buffer if the following orders have more

urgent requirements.

Now that the strategy is correctly implemented, we can code the smoothening function. To make it more user friendly, we incorporated an interface using Python library Tkinter. We plot the results in a browser window using Plotly, an advanced Matplotlib Library that enables the user to navigate through the plot.

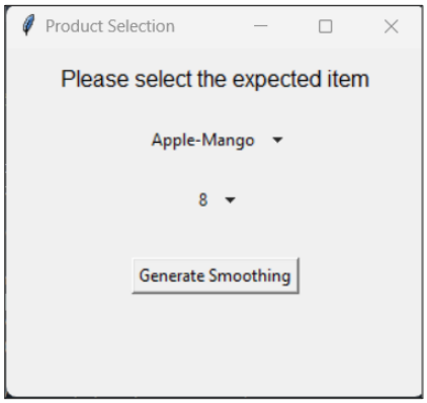


Figure 8: Drop down interface for product selection

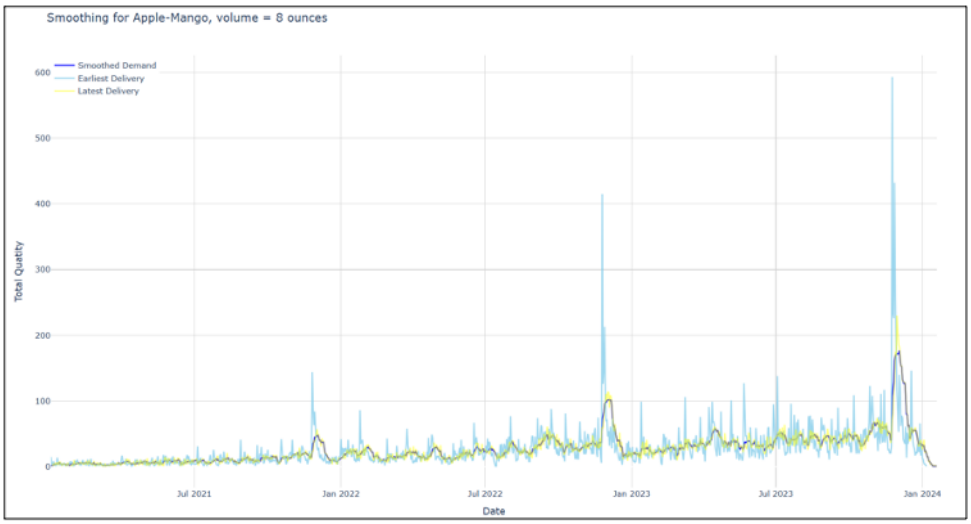


Figure 9: Smoothing for Apple-Mango 8 ounces

The results of the smoothening are shown in figure 9. The curve is way smoother than the EDD curve and is like the LDD curve. However, when we zoom in the graph, we can see that the smoothed demand curve is better than the LDD, especially during high demand periods.

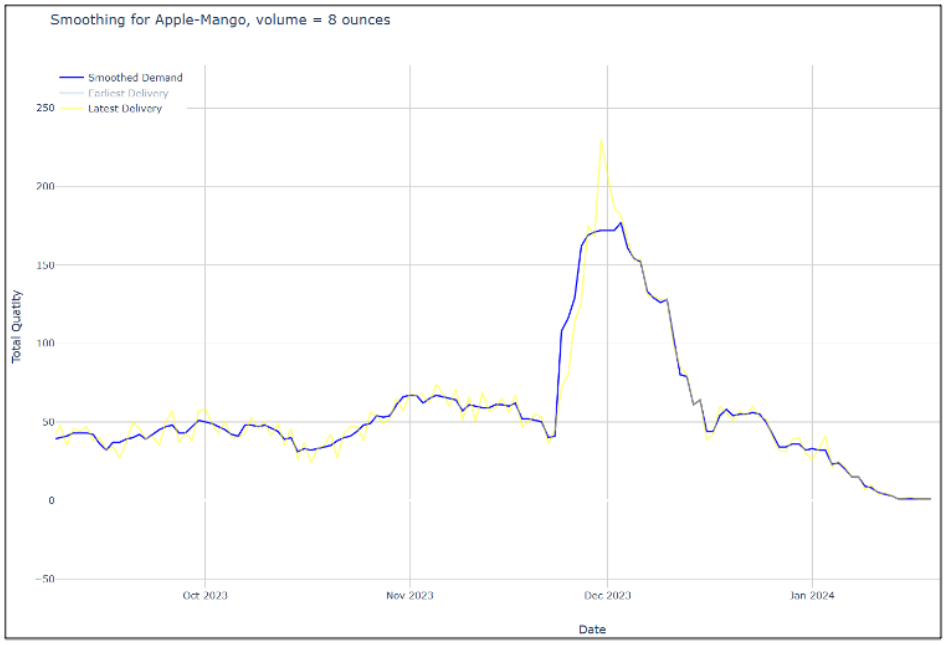


Figure 10: LDD demand VS Smoothed demand

Even during low demands periods, the smoothed demand curve still performs better:

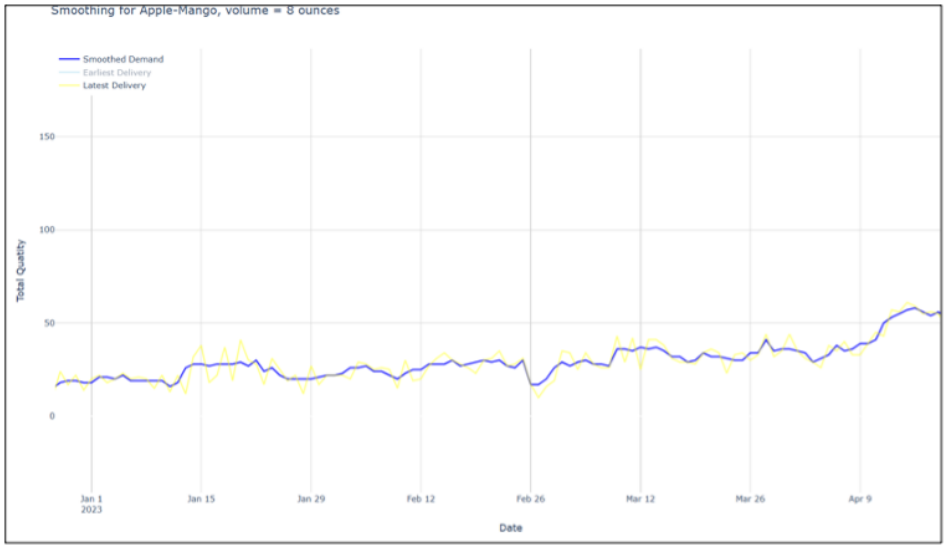


Figure 11: Zoom on smoothing for Apple mango 8 ounces

To conclude, we can say that the smoothing works effectively, even though it may be not the most optimal solution. Another downside for this method is that it does not consider the client's preferred delivery date, which may have a negative impact on customer service on the long term.

TASK 2

For this task, we shall investigate whether there is at the macroscopic level indication of demand growth over the horizon, day-of-week seasonality, weekly or 4-weekly seasonality, and promotional impact. To do so we will compute the data on Python. We draw a graph pointing out the demand of each day. Please find it in the figure below.

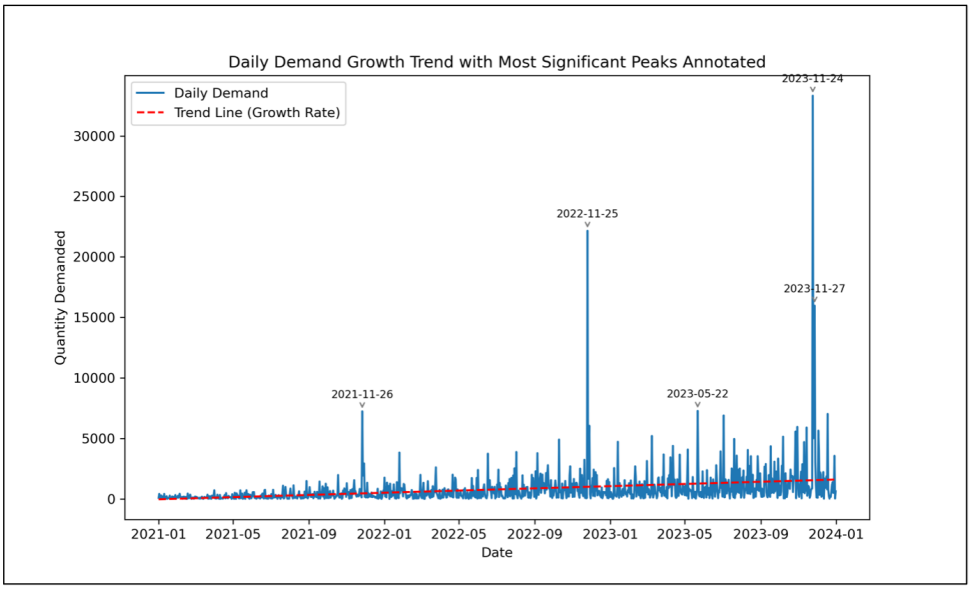


Figure 12: Daily demand growth trend with most significant peaks annotated

On this first graph, we can already see that there is a seasonality especially in november for Thanksgiving. We can also observe that there is a positive growth trend in demand over time.

We are going to go further in the analysis by looking at day of week seasonality. We define the seasonality as: $\frac{\text{Average demand on a specific day of the week}}{\text{Overall average Demand}}$

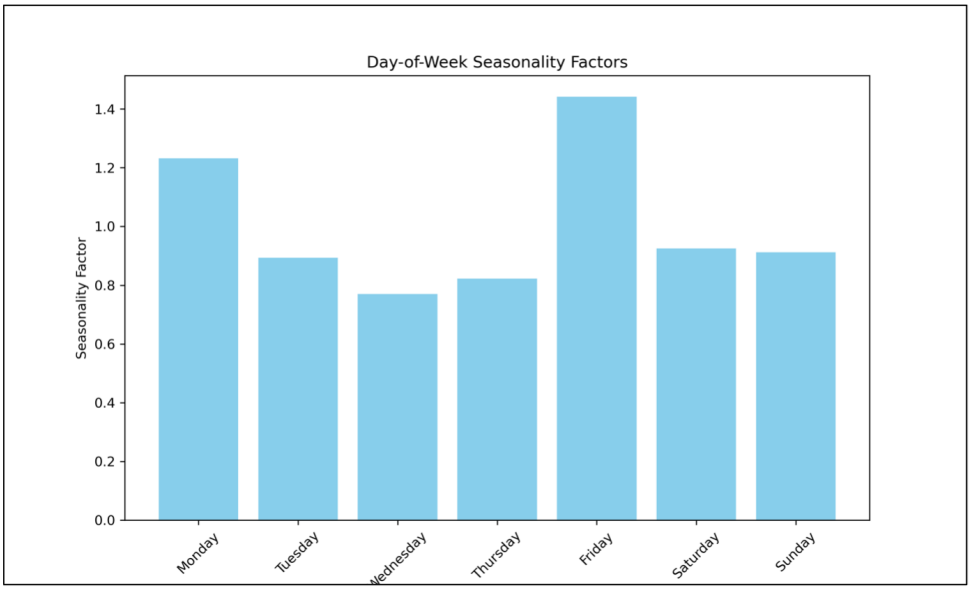


Figure 13: Day of week seasonality factors

A seasonality factor greater than 1 means that the demand on that day is above the average demand across all days. Indeed, a factor of 1.44 for Fridays means that Fridays typically have 44% higher demand than the average day.

A seasonality factor less than 1 means that the demand on that day is below the average. Indeed, a factor of 0.77 for Wednesdays means that Wednesdays typically have 23% lower demand than the average day.

We can also look at the seasonality over the weeks, as shown in the next graph. We took the data of the first year and looked how the demand fluctuates over the weeks. The demand is slightly going up through the year with a huge peak on Thanksgiving and then decreases a lot by the end of the year. It is indeed the week after Christmas.

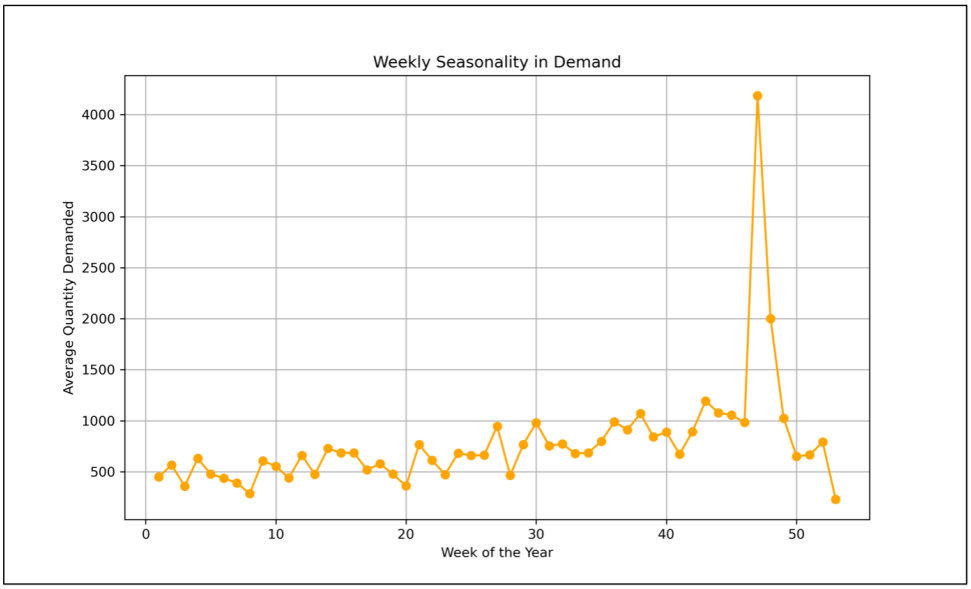


Figure 14: Weekly seasonality in demand

One final idea is to look over 4-weekly seasonality. The tendency remains as in the previous graph but one can notice that overall the growth is faster. Certain 4-week periods show peaks in demand, which could align with promotional cycles or other factors.

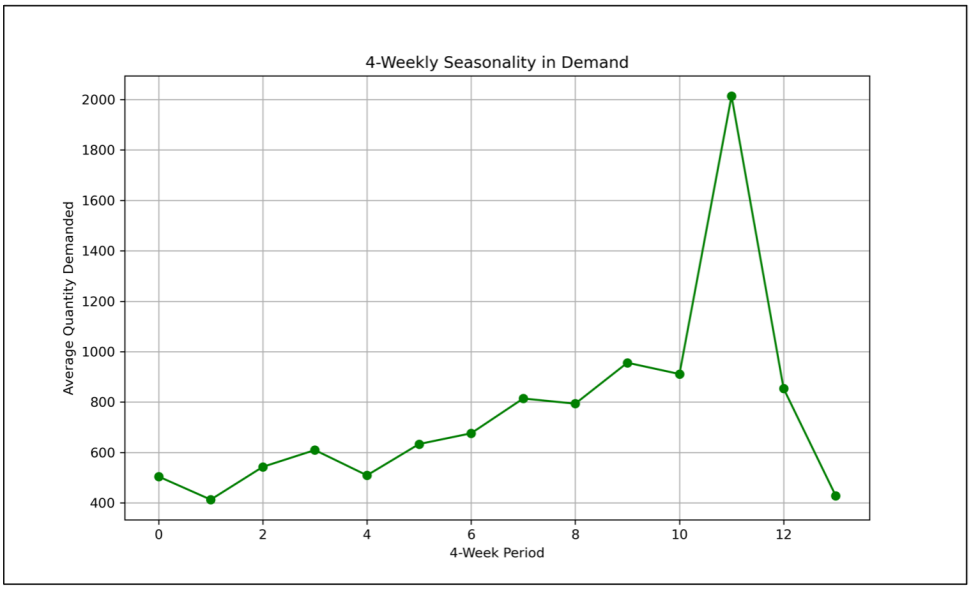


Figure 15: 4-Weekly seasonality in demand

TASK 3

For this task, we shall leverage the demand request history for all products, compute the demand share history of each product.

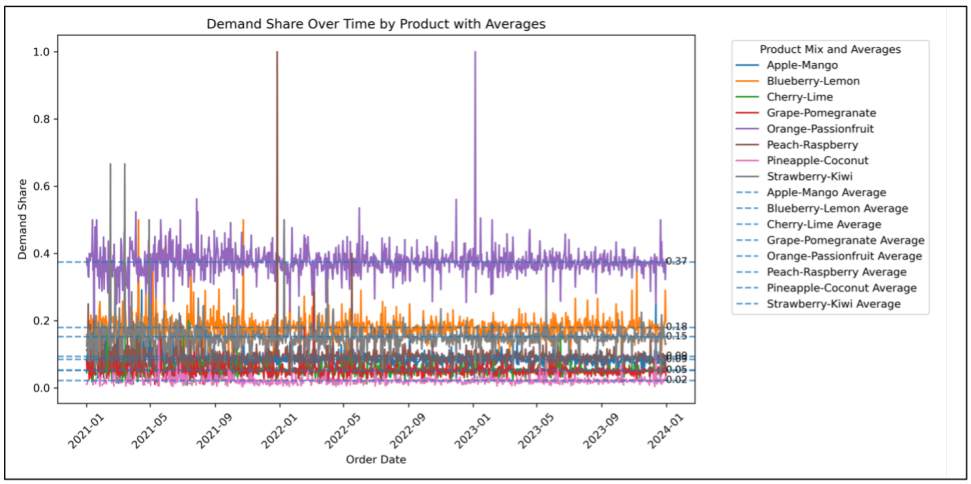


Figure 16: Demand share over time for each product with averages

The graph shows the demand share history for each product over time. We notice firstly that the trend doesn't change over time: Orange-Passionfruit remains the more ordered mix (37% share in average) and Pineapple coconut the least (2% in average). For all the other products, the trends remain the same and the order of preference does not really change over time.

Nevertheless, we notice that some irregularities appear by the end of the year. Peach raspberry has at one moment become the most popular mix while it wasn't before. This can be due to a special event or maybe just a promotion for this product. The same explanation can be brought forward for the other peaks we notice.

We can analyze the variability of the products by plotting a standard deviation graph as well:

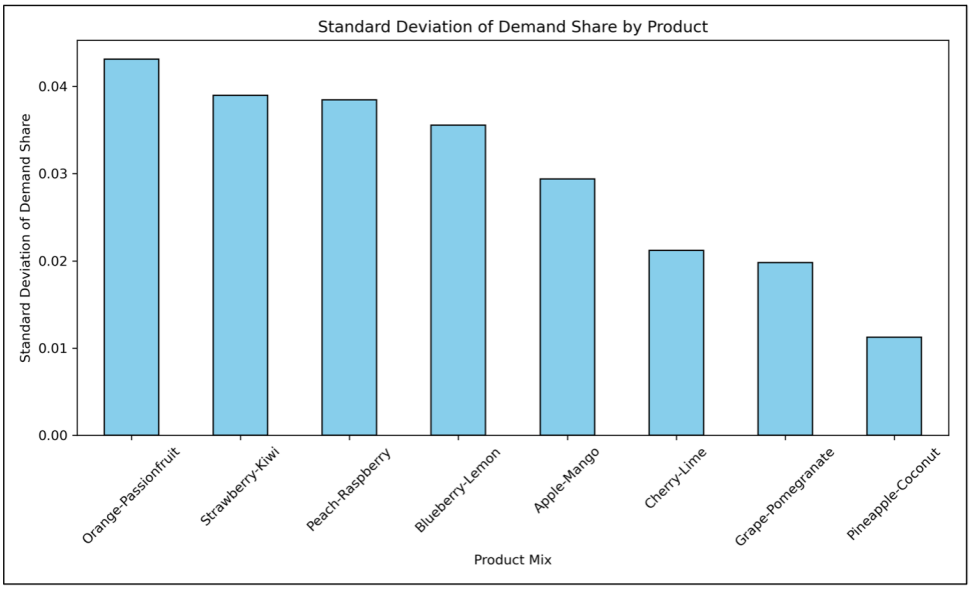


Figure 17: Standard deviation of demand share by product

We notice that the most demanded mix is also the mix with the greatest standard deviation. It is therefore one of the hardest products to predict how much stock we should keep of.

Here are the patterns we observe:

- High average demand share, high SD:
  - These products, like "Orange-Passionfruit," are popular but have unpredictable demand. This could be due to seasonality, promotions, or other market factors affecting their sales.
  - FruitSoul might want to ensure sufficient stock to handle potential spikes but should also be cautious about overstocking due to variability.
- Low average demand share, high SD:
  - These products have low overall demand but can experience occasional spikes. An example might be "Cherry-Lime," which might suggest niche popularity or appeal in specific periods (e.g., holidays or special events).
  - FruitSoul should be careful with inventory for these products—while not needing large quantities, they should still prepare for occasional high demand.
- Low average demand share, low SD:

– Products like "Pineapple-Coconut" with both low average demand share and low SD are consistently low in demand. Such products might be candidates for discontinuation, or they could be targeted for promotions to increase visibility and sales.

TASK 4

The graph shows the demand share history for each state over time. Some states exhibit relatively stable demand shares, while others show more fluctuation. For example, states like Alaska, Montana, and Rhode Island have more stable demand shares, while Texas shows significant variability.

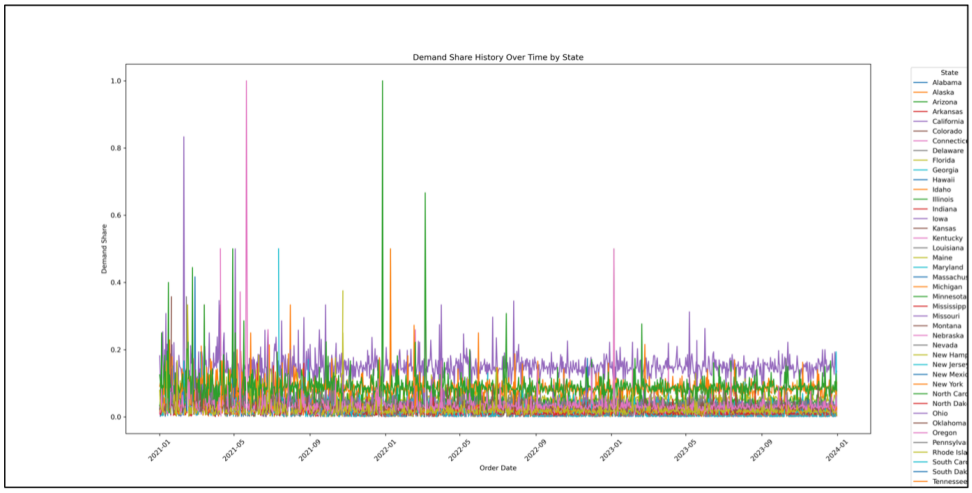


Figure 18: Demand share history over time by state

This graph is pretty hard to understand with 50 states being combined on the same graph. Nevertheless, we note that except for some specific peaks, the demand remains the same over time. We can calculate the average for each state and check the variability computing the standard deviation to get better insights.

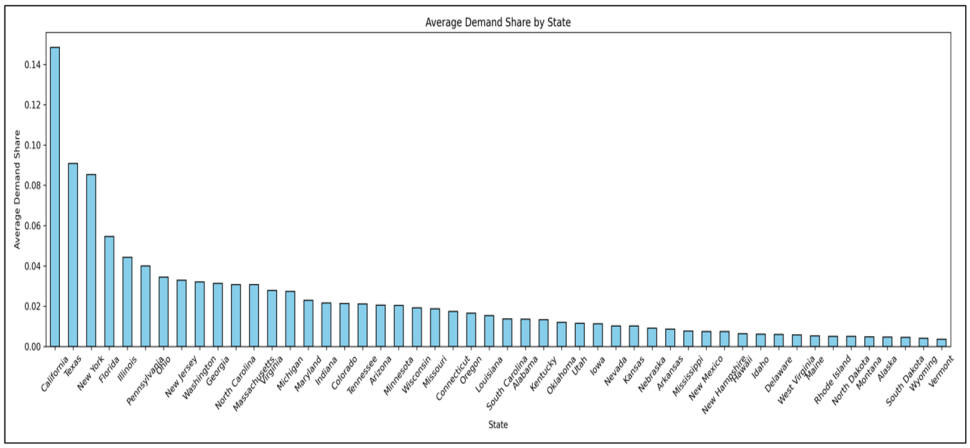


Figure 19: Average demand share by state

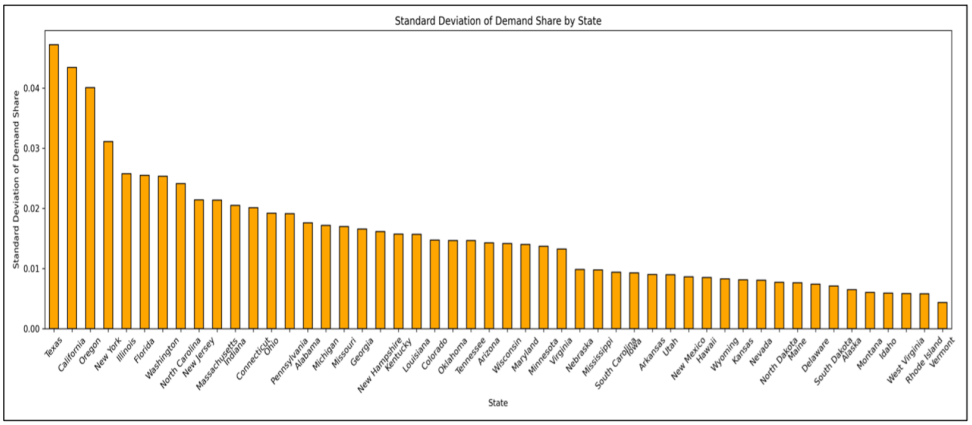


Figure 20: Standard deviation of demand share by state

Texas has the highest variability (0.0404), indicating a highly fluctuating demand share, likely due to promotions, market size, or changing demand patterns.

Alaska, Rhode Island, Montana, and Vermont have the lowest variability (around 0.003-0.004), suggesting very stable demand shares.

States with low variability in demand shares, such as Alaska, Montana, and Rhode Island, indicate consistent customer behavior or demand stability.

Texas and states like New York, Illinois, and California show higher variability, which could be influenced by local market dynamics, promotions, seasonality, or other factors affecting demand.

Let's compare the average demand and the variability for some specific cases:

- High average, high standard deviation states
  - Texas and California fall into this category. These states are major markets (high average) but also have highly variable demand (high SD). This suggests that while there is a strong customer base, the demand is not consistent, and strategies such as flexible inventory management, responsive supply chains, and targeted marketing may be required to handle the volatility.
- High average, low standard deviation states
  - These states (not significantly present in the graphs) would be ideal for businesses since they represent stable, high-demand markets.
- Low average, high standard deviation states
  - Oregon and some other states fall into this category. These states have lower average demand shares but higher variability. This might indicate niche markets

or areas where demand can spike due to specific events or factors. Businesses might consider targeted promotions or seasonal offerings in such states.

- Low average, low standard deviation states
  - States like Vermont and Rhode Island exhibit both low average demand shares and low standard deviations. These states have consistently low demand and might be considered for lower priority in terms of aggressive marketing or stocking strategies. However, they may represent stable, low-risk markets.

TASK 5

This task is similar to Task 1, but now we must take into consideration the shipping time to all the different States of America. This can be done directly on Excel by using the VLOOKUP function. We can also compute the time between the moment the order has been made and the Earliest, Latest and Preferred Acceptable Shipping date.

EASD: Earliest Acceptable Shipping Date

LASD: Latest Acceptable Shipping Date

PASD: Preferred Acceptable Shipping Date

i: order number *s:* state

$$EASD_{i,s} = EDD_i - Shipping\ Time_s$$
$$LASD_{i,s} = LDD_i - Shipping\ Time_s$$
$$PASD_{i,s} = PDD_i - Shipping\ Time_s$$

We can now compute the time between which the order is processed, and the shipping is made:

DRD: Demand Request Date

$$Minimal\ Order\ to\ Ship = EASD_{i,s} - DRD_i$$
$$Maximal\ Order\ to\ Ship = LASD_{i,s} - DRD_i$$
$$Preferred\ Order\ to\ Ship = PASD_{i,s} - DRD_i$$

If one of these values are negative, it means that the products cannot be shipped according to the clients' demand. If this is the case, the Earliest Possible Shipping date (EPSD) is the day the order is passed (providing the order is prepared and shipped the same day).

$$EPSD_i = \max(DRD_i, EASD_{i,s})$$

The Preferred Shipping Date (PSD) corresponds to the day the products have to be shipped from the warehouse/factory in order for them to arrive at the perfect time for the customer. Again, if the preferred order to ship time is negative, that means that the products will never arrive to the customer in time.

$$PSD_i = \max(DRD_i, PASD_{i,s})$$

Finally, the Latest Possible Shipping Date, compares the Latest Acceptable Shipping Date with the Demand Request Date.

$$LPSD_i = \max(DRD_i, LASD_{i,s})$$

If the Demand Request Date is earlier than the Latest Acceptable Shipping Date, than, the products may arrive in conformity with the client demand. However, if the Maximal Order to Ship Time is negative, that means that the command cannot be satisfied. Therefore, a choice can be made:

- either the command is still processed, the order is prepared the same day as the command arrives and is shipped that same day, it will arrive later than the client expected, but he will receive it.
- Or, the command is not processed, the client receives a message saying that his order cannot be shipped to the delivery point, and he gets an immediate refund.

We will consider both scenarios in our simulations for this task.

For the smoothing algorithm, we will use the same as presented in the task 1.5. However, this time, we can take into consideration customer satisfaction. This means that when possible, we can ship the product so that it arrives during the clients' Preferred Delivery Date.

Once again similarly to Task 1.5, we can compute everything into python, and using the Tkinter interface, we can generate the following interface:

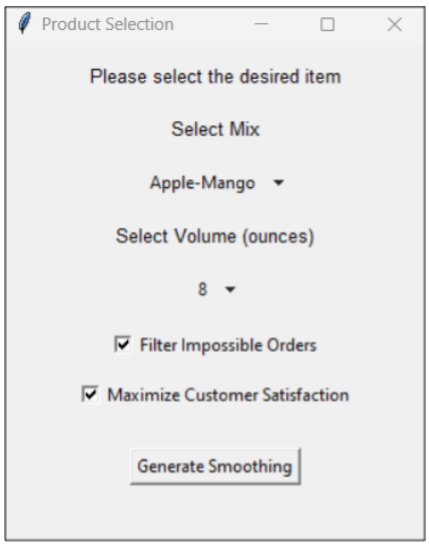


Figure 21: Python interface window for task 5

In this example, there is the possibility to filter out the impossible orders (the ones that cannot be shipped in time). There is also the possibility to ship the product so that it arrives the perfect day for him, thus maximizing customer satisfaction.

Let us compare the different outputs for each algorithm, for the product 'Apple-Mango 8 ounces':

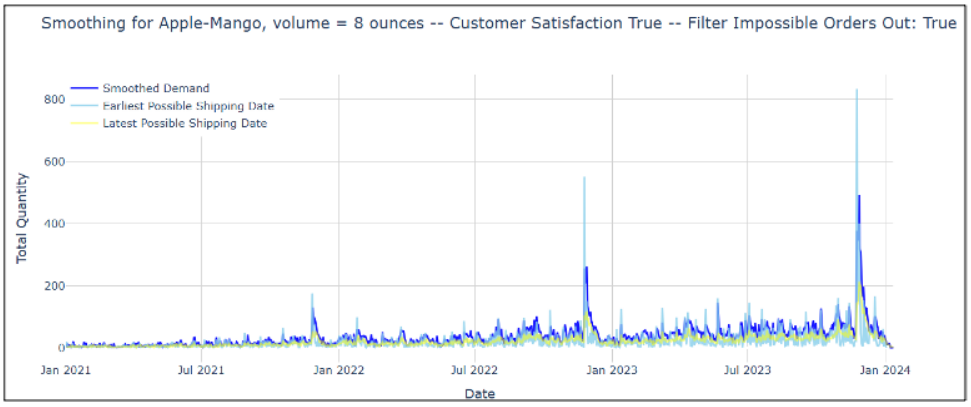


Figure 22: Total quantity and shipping date 1

In this example, the smoothing take the customer satisfaction into account, this explains why it is not as efficient as the yellow curve. It is however more efficient than when every order is shipped as fast as possible.

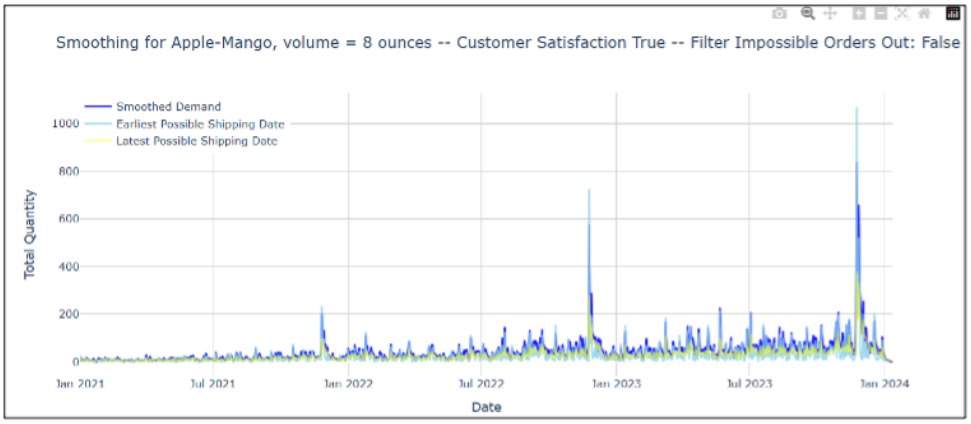


Figure 23: Total quantity and shipping date 2

In this example, we are not filtering out the impossible orders, this is why there are significantly more products to ship. Other than that, it is similar to the previous graph.

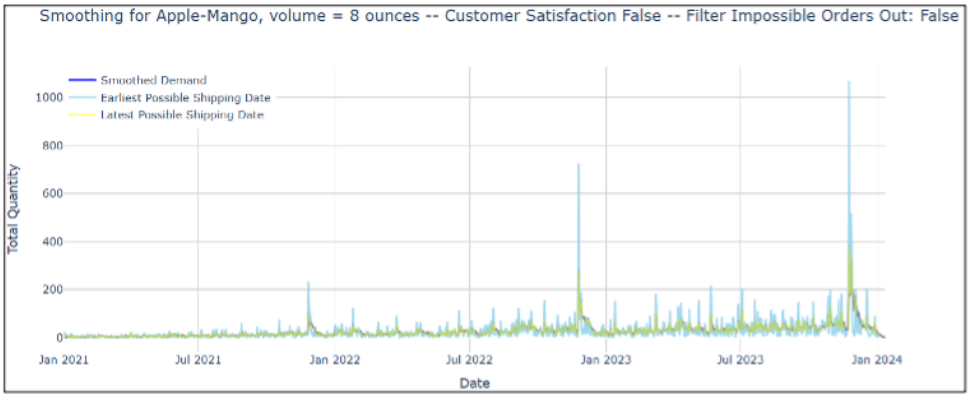


Figure 24: Total quantity and shipping date 3

In this case, we are not taking customer satisfaction into consideration, this explains why the smoothed algorithm gives significantly better results. Overall, he curve is very similar to the latest possible shipping date curve.

This can lead ultimately towards the final program for this task. The main idea is to try to find a compromise between smoothed shipping a customer satisfaction. We can do so by implementing a shipping threshold. If the shipping number per day is smaller than the threshold, than the objective will be to deliver the products the preferred date (when possible), however of the demand is too important, then the algorithm distributes the shipping orders as evenly as possible.

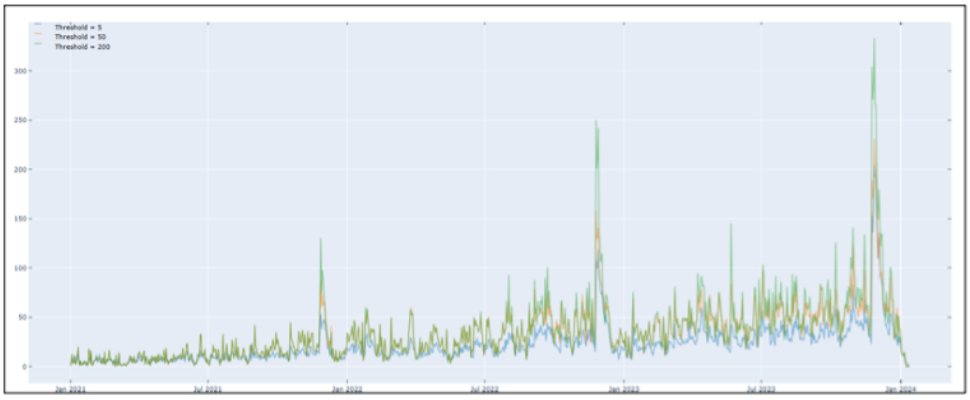


Figure 25: Apple-Mango 8 ounces for different threshold values

As expected, the most restrictive threshold (blue) distributes the shipping dates more evenly, but this is done at the detriment of customer satisfaction. The supply chain manager can feel free to adjust this value in depending on organizational goals.

TASK 6

A majority of Task 6 has already been treated in Task 5. In other terms, an order is considered impossible to satisfy if its Maximal order to Ship time is smaller than 0. This would mean that even if shipped the day when the order was made, the product would note arrive in time.

We can evaluate the proportion of impossible orders with the COUNTIF function on Excel, or by using Python.

We find that approximatively, **21%** at minimum of the orders cannot be shipped in time.

The filtered data frame is saved into a CSV format under the name 'Task 6.csv'.

Having over 1/5 of the orders that cannot be satisfied represents an important loss for the company. Some clients require that their package arrive the same day as their order, as their latest acceptable delivery date. This is obviously unreasonable. A way to prevent that these orders are actually lost, is by proposing another date to the client.

The good news is: 65% of the 21% of unrealizable orders fall short of the latest delivery date by one or two days maximum, so there is a good chance that the customer will still carry on with the order, and not abandon it right away. Indeed, FruitSoul sells food supplements, therefore the customer can live with waiting 1or 2 more days for his order to arrive, especially if he likes this product. There is a chance that a great part of the unrealizable orders are finally shipped out. However, the unrealizable orders are not considered for the rest of the study.

TASK 7

For this task, we will be using Python, as it is a fast option for doing multiple computations across the 60k lines of orders.

Firstly, we start by data preprocessing. We filter out the raw table (available under the name *Fruit Soul Task 6.csv*) to finally keep the following columns:

1. **'DemandID'**,
2. **'Mix', 'Jar Size (vol. ounces)'**,
3. **'State', 'Quantity'**,
4. **'Earliest Possible Shipping date'**,
5. **'Preferred Shipping Date.1'**,
6. **'Latest possible Shipping date'**.

We then create the following empty table (called Total Demand per Day).

Date	Early Mix	Smoothed Mix	Late Mix	Early Bottle	Smoothed Bottle	Late Bottle	Early Pack	Smoothed Pack	Late Pack

Table 3: Total Demand per Day

We then iterate through the filtered table, and for each date (for example *Earliest Possible Shipping Date* $= i$), we refer to the Total Demand Per Day, and add the following values:

```
for index, row in tqdm.tqdm(filtered_df.iterrows(), total=filtered_df.shape[0]):
    quantity = row['Quantity']
    volume = row['Jar Size      (vol. ounces)']
    earliest_date = row['Earliest Possible Shipping date']
    latest_date = row['Latest possible Shipping date']
    pref_date = row['Preferred Shipping Date.1']
    if earliest_date in demand_per_day.index:
        demand_per_day.at[earliest_date, 'Early Mix'] += quantity * volume
        demand_per_day.at[earliest_date, 'Early Bottle'] += quantity
        demand_per_day.at[earliest_date, 'Early Pack'] += 1

    if pref_date in demand_per_day.index:
        demand_per_day.at[pref_date, 'Preferred Mix'] += quantity * volume
        demand_per_day.at[pref_date, 'Preferred Bottle'] += quantity
        demand_per_day.at[pref_date, 'Preferred Pack'] += 1


    if latest_date in demand_per_day.index:
        demand_per_day.at[latest_date, 'Late Mix'] += quantity * volume
        demand_per_day.at[latest_date, 'Late Bottle'] += quantity
        demand_per_day.at[latest_date, 'Late Pack'] += 1
```

Figure 26: Extract of the Python code

Here, we are considering that the number of packings are equal to the numbers of orders that are placed for this day. Therefore, there can be multiple bottles of different sizes in a pack. There is about 66k orders to process, and 9 computations per iteration, therefore around 600k computations. This takes around 15 minutes to compute on a basic computer, using Python.

The result is the following table:

	Early Mix	Preferred Mix	Late Mix	Early Bottle	Preferred Bottle	Late Bottle	Early Pack	Preferred Pack	Late Pack
01/01/2021	1080	488	160	55	27	8	53	27	8
02/01/2021	5064	1848	1184	294	96	61	283	92	57
03/01/2021	2384	1704	1032	145	105	54	141	100	51
04/01/2021	528	1736	984	31	105	53	26	103	53
05/01/2021	3720	1512	1352	209	91	77	196	86	73
06/01/2021	1416	2288	1280	87	119	73	82	117	71
07/01/2021	2808	2096	1784	163	127	108	156	118	101
08/01/2021	872	1744	1560	51	106	95	46	98	89
09/01/2021	1008	2360	2160	53	135	122	53	127	115
10/01/2021	5608	2408	2184	308	129	130	300	128	127
11/01/2021	1096	2144	1744	59	122	105	56	118	104

Figure 27: Output of the Python code

With this information, we can plot the following graph:

Figure 28: Graph of the output of the Python code

We can see that the trend is similar to the demand. Like the previous tasks, we observe that the earliest demand pics are extremely constraining for the company.

Later on, this code will be slightly modified to take into account the weekends and the holidays (task 8) as well as the number of jars of 8, 16, 32 ounces necessary to evaluate the total cost of production (task 9).

TASK 8

For this task we'll want to compute the proper number of workers and cells we need to have so as to minimize cost, taking into account that storage is not possible. We'll be reasoning yearly but we will first calculate the number of workers and cells for each day. We will have two teams working on one shift a day each. We have decided that there will be no third shift, otherwise the weekly limit of working hours would be reached (55 hours/week). We are working on excel using the spreadsheet generated previously in which the demand is smoothed around the preferred date. We have modified this sheet on python first in order to take into account that workers do not work on Sundays nor 4th of July nor Labor Day. The demand of these days will be allocated to the previous or next day if possible (remaining between the earliest/latest range). We now have this spreadsheet to work on (see figure). We will proceed as follow:

1. Define the number of M workers and cells. We compute them first because they can work anywhere, so if they are not employed in mixing, they can go to bottling or packing.

2. Define the number of B workers and cells. We will first calculate the number of M workers who are not working in mixing and can therefore go to bottling. Then we'll hire B workers to finish.

3. Define the number of P workers and cells. We will regard the possibility of hiring Part time on demand P workers to complete the job as well in this part

	Smoothed Mix	Smoothed Bottle	Smoothed Pack	Number8	Number16	Number32
1/1/21	240	13	13	4	5	4
1/2/21	1936	95	88	36	15	44
1/3/21	0	0	0	0	0	0
1/4/21	1160	59	59	21	14	24
1/5/21	1624	103	99	51	28	24
1/6/21	1488	84	82	38	18	28
1/7/21	2024	119	110	55	29	35
1/8/21	1976	127	119	65	33	29
1/9/21	2240	127	121	56	30	41
1/10/21	0	0	0	0	0	0
1/11/21	2432	149	147	72	38	39

Figure 29: Initial data for task 8

On the spreadsheet, we will be looking constantly at these KPIs:

- Total cost without materials. Materials are too expensive, and do not depend on the number of workers and cell so they make the number bigger for no reason. But we will take them at the end, calculating the profit.
- Number of M and B workers not working on a particular day and on average. We will try to minimize this number so as not to pay people to do nothing.
- Number of hours of work per week to make sure we do not go above the limit.
- Proportion of on demand P workers among the workforce to make sure it remains below 30% and ideally below 15%.
- % of unsatisfied demand to leverage our options in order to reach the level required.

M workers

When computing the best duo process center/number of workers, we notice that we do not improve the satisfaction by making this quantity grow linearly. Here is a chart (on the top) that compares the total cost for each of these duos, given all other parameters remaining constants. We are going to compare it with the percentage of unsatisfied demand for each combination (on the bottom).

M/CELL / PCM	1.00	2.00	3.00	4.00	5.00	6.00	7.00	8.00	9.00	10.00
1	1,752,962	1,833,842	1,914,722	1,991,554	2,064,522	2,140,618	2,215,978	2,283,978	2,352,346	2,427,338
2	1,978,642	2,140,402	2,433,562	2,433,562	2,579,866	2,733,346	2,884,986	2,895,480	3,185,322	3,335,306
3	2,204,322	2,446,962	2,656,482	2,879,434	3,110,298	3,341,346	3,341,346	3,341,346	4,035,226	4,269,954
4	2,430,002	2,744,322	3,037,850	3,336,898	3,644,778	3,955,418	4,266,058	4,581,114	4,896,170	5,212,698
5	2,655,682	3,043,338	3,421,978	3,421,978	4,187,538	4,578,690	4,970,210	5,365,594	5,762,450	6,160,778
6	2,881,362	3,344,378	3,803,714	4,264,154	4,733,058	5,205,090	5,676,938	6,152,466	6,630,018	7,110,698
7	3,107,042	3,644,498	4,183,426	4,731,002	5,280,786	5,280,786	6,385,506	6,941,730	7,499,978	8,061,538
8	3,332,722	3,946,458	4,566,266	5,196,930	5,829,618	6,462,674	7,095,546	7,732,650	8,371,778	9,014,218
9	3,558,402	4,250,442	4,950,762	5,662,306	6,376,242	7,089,994	7,805,034	8,523,018	9,243,026	9,966,346
10	3,784,082	4,554,610	5,335,810	6,127,682	6,921,946	7,716,578	8,514,890	9,313,754	10,114,642	10,918,842

M/CEL / PCM	1.00	2	3.00	4	5.00	6	7.00	8	9.00	10
1	86%	65%	54%	46%	41%	34%	26%	22%	20%	18%
2	73%	39%	27%	22%	18%	14%	10%	8%	8%	7%
3	60%	25%	16%	12%	10%	8%	6%	5%	4%	3%
4	50%	17%	11%	8%	7%	6%	3%	2%	1%	1%
5	41%	12%	8%	6%	5%	3%	1%	0%	0%	0%
6	34%	9%	6%	5%	3%	2%	0%	0%	0%	0%
7	28%	8%	5%	3%	2%	1%	0%	0%	0%	0%
8	24%	7%	4%	2%	1%	0%	0%	0%	0%	0%
9	21%	6%	3%	1%	0%	0%	0%	0%	0%	0%
10	18%	5%	2%	0%	0%	0%	0%	0%	0%	0%

Figure 30: Costs and satisfaction for each value of M workers and each number of mixing process center

We notice that several costs lead to the same percentage of unsatisfied demand. We are going to compute the least cost associated with each percentage. Here is what we get:

M/CELL PCM	1.00	2.00	3.00	4.00	5.00	6.00	7.00	8.00	9.00	10.00
1	1,752,962	1,833,842	1,914,722	1,991,554	2,064,522	2,140,618	2,215,978	2,283,978	2,352,346	2,427,338
2	1,978,642	2,140,402	2,433,562			2,733,346	2,884,986	2,895,480		3,335,306
3	2,204,322	2,446,962	2,656,482	2,879,434			3,341,346	3,341,346	4,035,226	
4	2,430,002	2,744,322	3,037,850				4,266,058	4,581,114	4,896,170	
5								5,365,594		
6		3,344,378								
7	3,107,042									
8	3,332,722									
9	3,558,402									
10										

M/CEL PCM	1.00	2	3.00	4	5.00	6	7.00	8	9.00	10
1	86%	65%	54%	46%	41%	34%	26%	22%	20%	18%
2	73%	39%	27%	22%	18%	14%	10%	8%	8%	7%
3	60%	25%	16%	12%	10%	8%	6%	5%	4%	3%
4	50%	17%	11%	8%	7%	6%	3%	2%	1%	1%
5	41%	12%	8%	6%	5%	3%	1%	0%	0%	0%
6	34%	9%	6%	5%	3%	2%	0%	0%	0%	0%
7	28%	8%	5%	3%	2%	1%	0%	0%	0%	0%
8	24%	7%	4%	2%	1%	0%	0%	0%	0%	0%
9	21%	6%	3%	1%	0%	0%	0%	0%	0%	0%
10	18%	5%	2%	0%	0%	0%	0%	0%	0%	0%

Figure 31: Costs and satisfaction for each value of M workers and each number of mixing process center - data analyzed

The minimum costs associated with each percentage are not equally placed in the chart. Nevertheless, we notice that having few people or few process centers result in a cheap cost but very unsatisfied demand, which makes sense. We can plot the graph of the cost as a function of a percentage of satisfied demand.

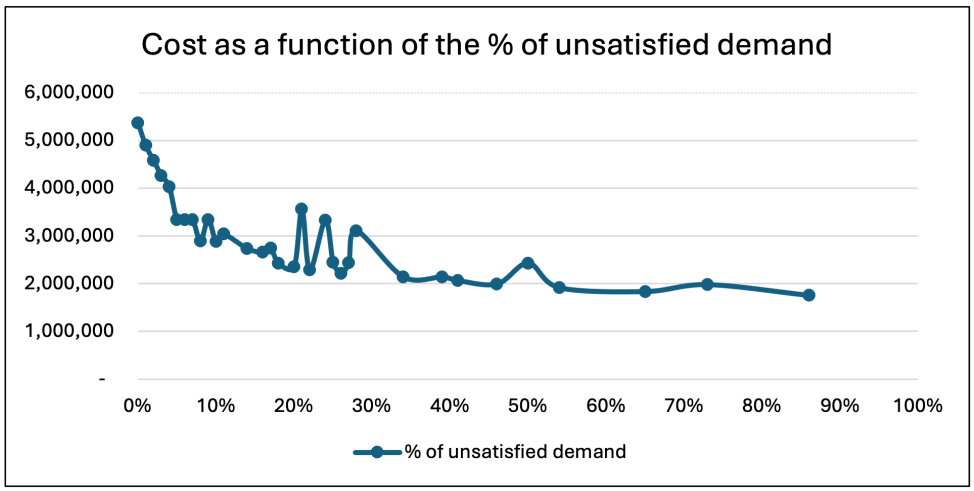


Figure 32: Cost as a function of the % of unsatisfied demand

We notice two facts. First this function is nonlinear, which makes sense since the capacity of the machines is not either. Second, the function is not strictly decreasing.

This second observation was less predictable, it means that sometimes, when willing to reduce the percentage of unsatisfied demand, we finally end up by reducing the cost as well. We will choose such a point in the case then.

At this moment we can compute the best number of M workers and cells for a certain level of satisfaction for one year.

B workers

Let's move on onto the bottling center. First, we will need to determine each day the number of M workers who are employed each year but are not employed in the mixing center each day. Then we will determine how many workers we need in the bottling center to fulfill the demand each day. Then we will assign as many M workers as possible to the bottling center. If the number of required workers is greater than the number of M available workers, we will compute the number of B workers we will need to add each day. One may want to take the maximum value of B workers to be hired for the whole year.

Nevertheless, we often do not need all the workers most of the time. So, in order to avoid having multiple B workers employed to do nothing, we will evaluate the influence of not hiring the maximum B workers with a different method. We will look how often do we need to have a specific number of B workers per day on a specific cell (considering it might be a very busy day so not being able to complete the demand might result in greater loss). We can also change the value of the number of process centers and see how it impacts the number of required B workers to be hired. Here for instance, most of the time 6 workers per day per cell is enough to cover the demand at a certain point. So, we can decide to have 6 workers per day per cell and put them on two different shifts.

values of need for bottling workers/cell	number of occurrences
0	240
1	40
2	5
3	19
4	10
5	14
6	4
7	9
8	10
9	3
10	11

Table 4: Values of need for bottling workers per cell and their occurrences

P workers

The last decision regards the packing center. We want to determine the number of workers required there, given a certain number of process centers. We will determine the number of M and B workers we can assign to packing. And then we'll fill in the empty spots with either full time P workers or on demand P workers. The idea here is to minimize the number of full time P workers so that they will always have something to do and hire on demand to be able to manage peek situations. We will reason as for bottling. We get the following chart for one situation. We need to keep in mind that we can hire on demand P workers as long as they don't represent more than 30% of the workforce on a given day. Here we will hire 6 P workers and hire on demand for the days when we need more.

need of P/cell	number of occurrences
0	207
1	33
2	28
3	12
4	5
5	2
6	24
7	11
8	18
9	9
10	16

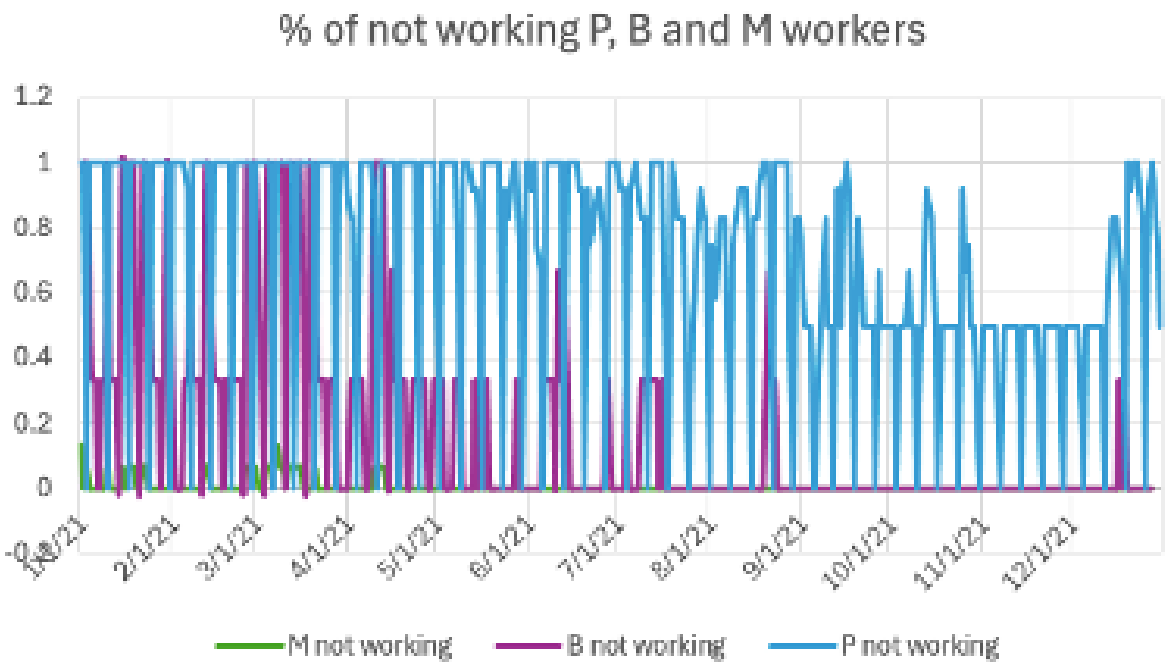
Table 5: Need of P per cell and number of occurrences

Here are the results we get:

	PCM	PCB	PCP	M	B	P	Padd	cost without materials	%unsatisfied demand M	%unsatisfied demand B	%unsatisfied demand P
2021	3	1	2	15	3	12	138	3,037,782	10%	4%	9%
2022	4	3	4	28	12	24	389	6,140,918	0%	2%	8%
2023	7	4	7	49	24	49	414	11,119,982	0%	1%	9%

Figure 33: Final results task 8

Let's analyze some KPIs for 2021:



The % of not working P, B, M workers in 2021

We wanted to make sure our employees have some work to do all year long. Here we succeeded with M workers who are not working only 0.06% of the time. As for B workers, who remain not working 19% of the time on average, we notice that depending on the day, they are all working or all not working. Indeed, since we cannot store the products, we have to meet the demand every day and since it varies a lot, so does the number of working B workers. Finally P workers spend a lot of time not working (79% in average) because we have to hire a sufficient number or full time P workers to make sure that on demand P workers do not represent more than 30% of the workforce, the maximum reached being 27% around Thanksgiving.

Figure 34: Time worked by workers

Here we have the number of hours the workers have to work each week. Since they are employed full time, this means 8 hours a day, 6 days a week, they are always below the week limit. The week of Labor Day has one more day off than usual weeks which explains the decrease during week 37. But even if they respect the time limit, they do extra hours, we decided that Saturday would be the day when they do it so we need not to forget about them when computing the final cost.

TASK 9

For this task we will use the same spreadsheet as for task 8 making sure to include all the costs:

- Worker's salary (normal rate and extra hours)
- Costs mixing, bottling, packing
- Costs of space
- Cost of cell set up (*taking into account the previous set up cells*)
- Daily cell cost

We will compute the revenue considering that market prices for 8-ounce, 16-ounce, and 32-ounce jars are respectively \$39, \$59, and \$89.

Then for each week and year, we will analyze the profit for the numbers of employees, and cells determined in task 8.

Here we have the day-by-day profit for each year. The profit follows the same trend each year with regular peaks, due to special events (Thanksgiving) or promotions maybe.

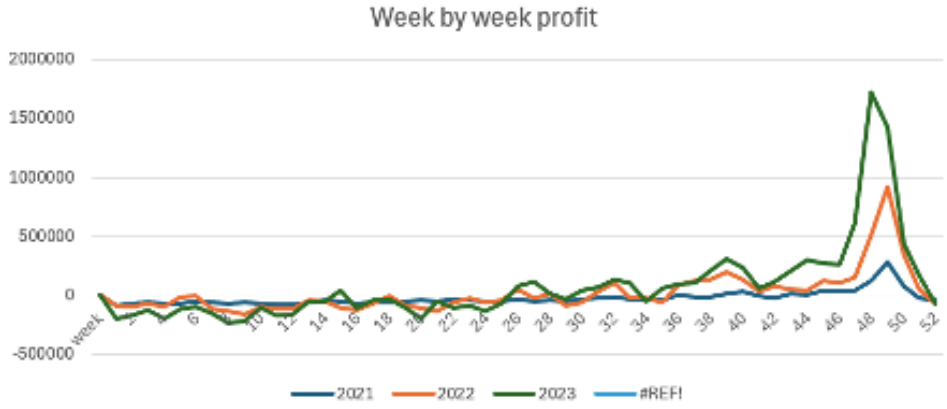


Figure 35: Week by week profit

When looking week by week, we notice that the 2021 profit is often negative, we'll have to check with the yearly analysis if they are not in deficit this year. We observe the same trend every year with a huge peak around Thanksgiving for Black Friday and regular peaks probably due to promotions. We notice as well that throughout the years the profit tends to raise. Indeed, the first year FruitSoul had to buy a lot of machines and do many investments that start being profitable the following years.

Figure 36: Year by year profit

Indeed, when looking at the yearly profit we notice that in 2021 FruitSoul has a deficit of $1,280,400. But the following years, this deficit is compensated by huge profits of several millions of dollars. We might try to reduce the initial deficit, but it would result in meeting less demand. So would lead to unsatisfied customers who won't buy again from FruitSoul, reducing the demand for the next years.

TASK 10

For this task, we'll compute first the number of cases that can be stored on a single shelving, depending on the jar size. Here is what we found:

Total cases per rack for 8-ounce jars: 420
Total cases per rack for 16-ounce jars: 306
Total cases per rack for 32-ounce jars: 130

We have then developed an algorithm that computes the number of racks required for different numbers of jars of each size. You can go into the algorithm and change the values in the script to do the computation for any number of jars. Here is what we have found for 1000 cases of 8-ounce jars, 200 cases of 16-ounce jars and 400 cases of 32-ounce jars.

Racks needed for 1000 cases of 8-ounce jars: 3.0
Racks needed for 200 cases of 16-ounce jars: 1.0
Racks needed for 400 cases of 32-ounce jars: 4.0

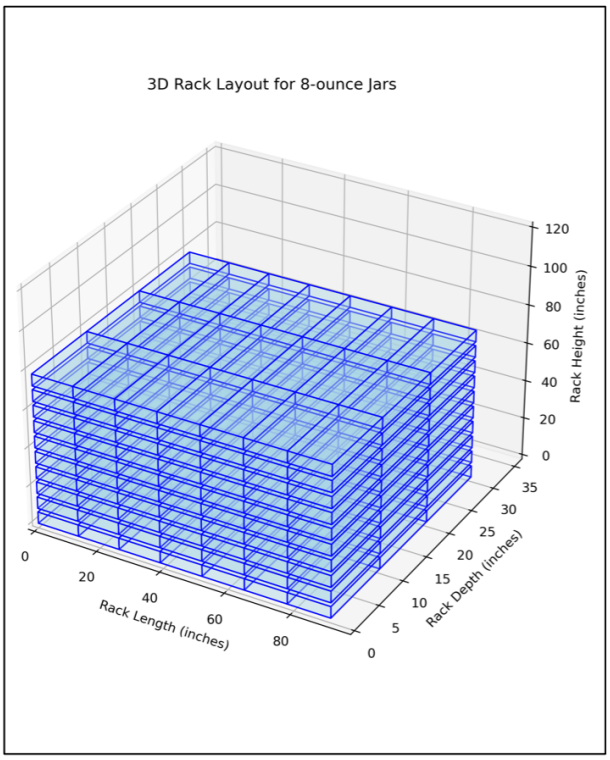


Figure 37: 3D Rack Layout for 8 ounces Jars

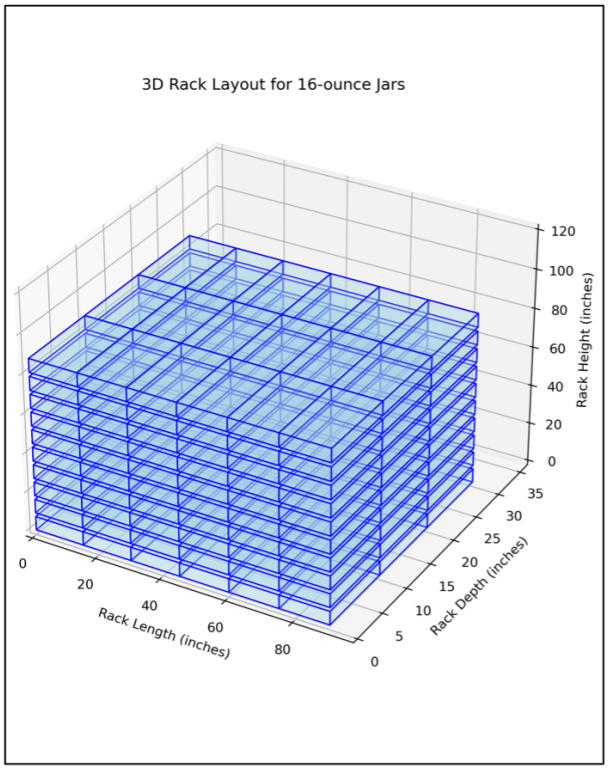


Figure 38: 3D Rack Layout for 16 ounces Jars

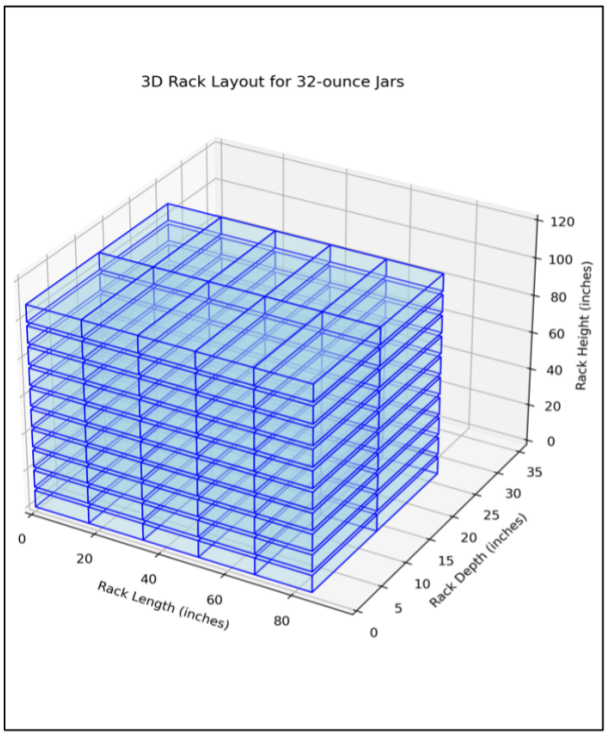


Figure 39: 3D Rack Layout for 32 ounces Jars

The total used space for the products in the rack, with this layout is approximatively:

$$85 \times 35 \approx 3000\,\text{inches}^2$$

over a total available space per rack of

$$36 \times 144 \approx 5200\,\text{inches}^2.$$

This leaves us with an available space per aisle of

$$2200\,\text{inches}^2.$$

If we state that the aisle width in front of the rack is 2 feet (enough to let a person pass), the total space occupied will be:

$$85 \times 35 + 2 \times 12 \times 85 \approx 5015\,\text{inches}^2 \leq 5200\,\text{inches}^2.$$

Therefore, the rack space is fully optimized.

TASK 11

a) In this situation, the workers don't have to think about when the product will go out for delivery, they only must take the package and store it the FPDC. The FPDC holds inventory until the packing center requests it. Then the only thing they need to make sure of is to make sure to respect the FIFO process, always picking the oldest cases first to minimize potential obsolescence or quality degradation.

In this process, all packages follow the same path, no time is waisted thinking about where to store it. The process is easily repeatable and ensures that no product gets overdate. All inventory is consolidated at the FPDC, making it easier to track, manage, and control stock levels.

Since all finished products are first stored in the FPDC, there could be delays in replenishing the packing center if there is an immediate or urgent need. Bottles might need to be handled multiple times (from bottling to FPDC, then FPDC to packing), which can increase labor costs and handling errors. Higher storage costs at the FPDC due to the need to hold all finished product inventory, even when some could be sent directly to the packing center.

b) In this situation, the product has the possibility of going directly to the packing area if it is about to be sent in H hours. In the other case, it will be sent to the FPDC. This method reduces the number of travels of some products, sending them directly to where they need to go.

The time is optimized here. By directly sending bottles to the packing center when there's an immediate need, the replenishment process is more responsive, reducing the waiting time. Direct shipping from the bottling center to the packing center reduces the number of handling steps, potentially lowering labor costs and the risk of handling errors. Only bottles not immediately needed for packing are stored at the FPDC, potentially reducing storage costs and space requirements. But if we need to pack product A and we have just produced it, the product will go directly to the packing area even if there are products A in the FPDC that have been waiting there for a long time. There might be some waste in the process.

Misjudgments in the demand forecast or errors in determining the H-hour window could lead to overstocking at the packing center or stockouts. This practice also requires dynamic decision-making and communication between the bottling center and the packing center to determine replenishment needs.

We would recommend practice (b) to FruitSoul if the packing center's demand is relatively predictable, and there is a robust system to forecast replenishment needs within the H-hour window or if the company aims to minimize handling and labor costs and optimize storage space usage at the FPDC. This process seems therefore more efficient and flexible. We would recommend practice (a) to FruitSoul if there is a high level of uncertainty or variability in packing center demand, making dynamic routing more challenging or if reducing the risk of stockouts and maintaining a consistent inventory turnover with FIFO is a higher priority.

TASK 12

For this task, we will have the possibility to store jars. This will help us meet the demand at a reduced cost since we will be able to anticipate and hold an inventory. We will proceed as follows. We will keep the same number of M workers and mixing cells as in task 8. Nevertheless, in task 8, we would only produce the required quantity. Now we will produce more mix than before by adding 1 more employee than needed if possible (if they are less than 5 per cell). For instance, when we had 1 employee working on a cell and producing 640 ounces on an 8-hour shift, we now will have two employees producing 1664 ounces.

As a result, we will have more mix to use for bottling. We will therefore pour all this mix into bottles of 8, 16, 32 ounces. We will split the mix proportionally to the number of 8, 16, 32 oz required each day, filling first 8 oz jars, then 16 then 32. We will put all of these jars into inventory and will take out the bottles of the inventory every day to fulfill the demand, respecting FIFO. When doing this, we can reduce the number of B workers since we will be able to anticipate.

We will keep on filling in the jars until the yearly demand is reached. At this point, we will not have to produce more mixes or bottles for the year, but just having to take out of the inventory.

We know the number of jars we can put on each rack, so we can determine the number of racks needed every year (maximum of daily racks needed).

To compute the number of workers required every day, we will do as in task 8, computing the number of M workers then B then P.

Figure 40: Final results for task 12

We need a lot of racks, so this storage might be expensive, and it need to be taken into account when computing the final cost. Nevertheless, we notice that we get a 100% satisfaction of the demand in bottling, which way better than before. We can compare the current results with task 8:

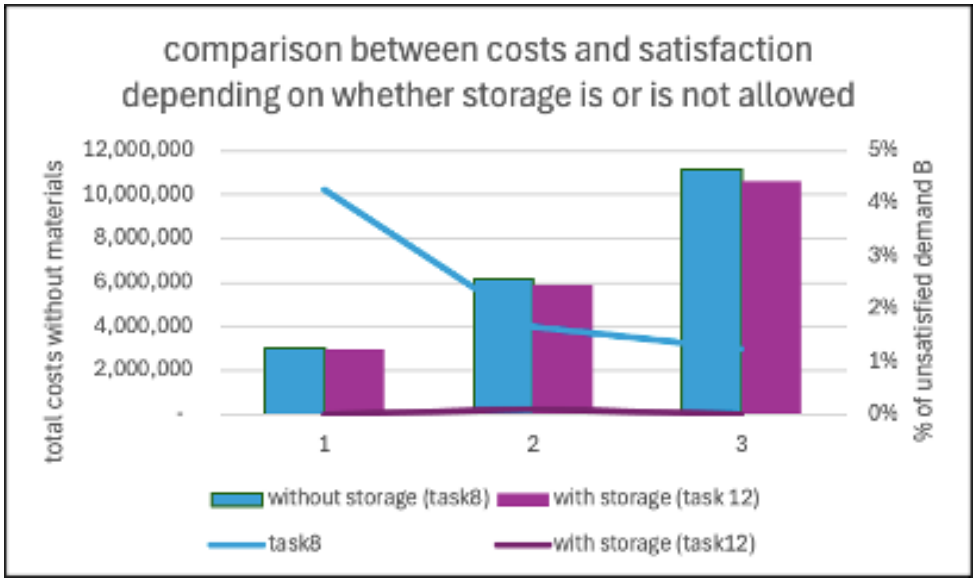


Figure 41: Comparison between costs and satisfaction depending on whether storage is or is not allowed

We have reduced the costs by 5% in average with the storage option but mostly we have succeeded in meeting 100% of the demand when only 92% was met in the previous option. This is great improvement.

Let's look at the workforce distribution in bottling and Packing in 2023.

Figure 42: Workforce in packing

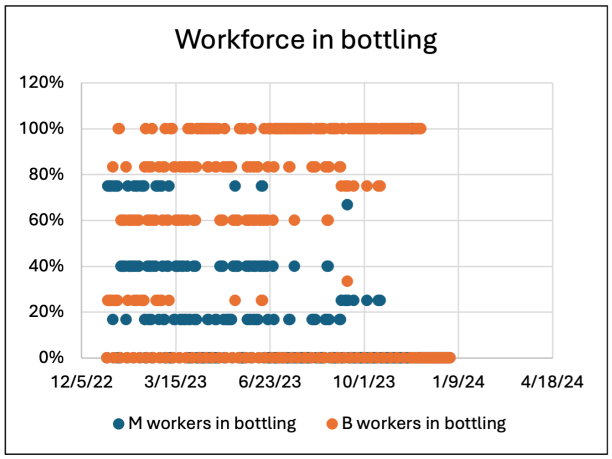
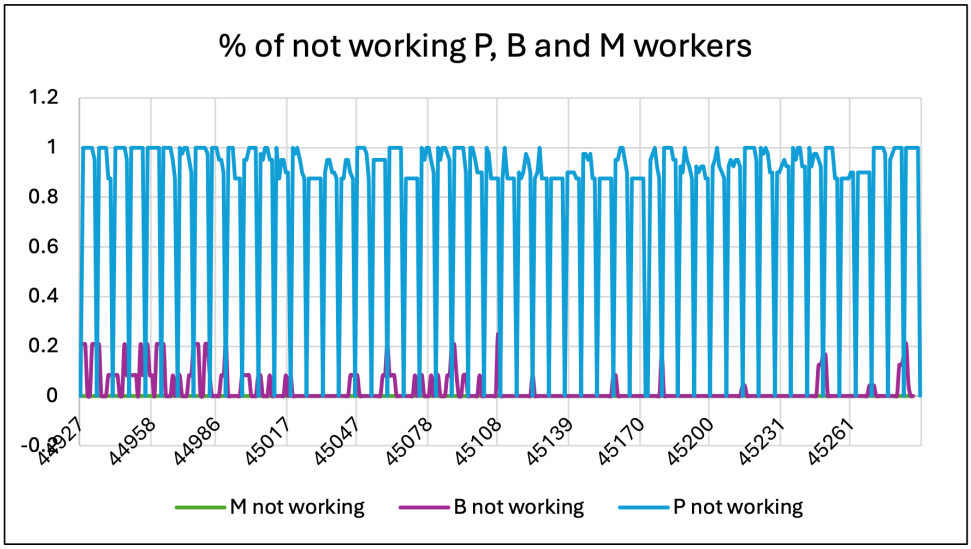


Figure 43: Workforce in bottling

When M and B workers are not working in mixing or bottling, we assign them to bottling then packing. We will always assign M workers first because they are more expensive, and we do not want them to be doing nothing on a day. Depending on the day, we notice that the proportion of M and B workers working in packing or bottling change a lot which means we are taking the most advantage of these workers. We note that very often, B workers do not represent the biggest proportion of workers in bottling and same for P in packing even if this is where they were supposed to be assigned firsthand. It is due to the fact that we start assigning M then B than P to all of these positions. The ideal would be to have as much B workers in bottling and P in packing as possible to make sure we do not have over qualified employees but since we have to hire the workers full time for one year, we want to make sure we get them working.



The new % of not working P, B, M workers

Indeed, when analyzing this plot, we note that B and M are working most of the time, with M workers working in average 100% of the year, and B workers working in average 97% of the year. As in task 8, in order to respect the limit of on demand P workers, we notice that full time P workers are not working 80% of the time. But this is the best we can do for now.

TASK 13

For this task, we now want to find out how to store the mix containers in the racks. As in task 11, we'll only be able to stack one pile of containers on each rack, otherwise there might be a height conflict since the rack are 12' high. We'll just have to compute the number of containers we'll be able to store on one rack and just present the layout. We do the computations on Python and find out that 288 containers fit on one rack.

Here is the 3D layout:

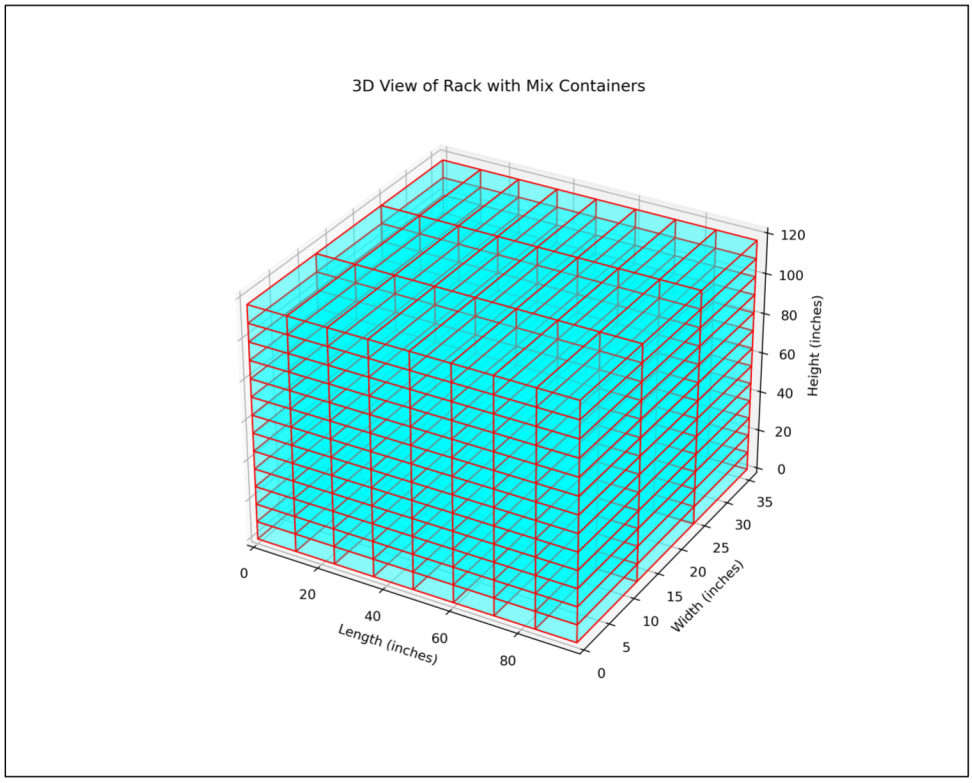


Figure 44: 3D Rack Layout for mix containers

TASK 14

a) In this scenario, all mix containers are first stored in the Mix DC, waiting to be transferred to the bottling center based on need. The FIFO principle is applied to ensure that older mixes are used first, minimizing potential spoilage or quality degradation.

   This ensures that older batches are processed first, reducing the risk of obsolescence or spoilage. This is especially important for mix containers, which may have shorter shelf lives than bottled products. The process is standardized and easy to follow. Workers always move the mix containers to the Mix DC first and then retrieve them when needed, simplifying decision-making and routing.

   Nevertheless, storing all mix containers first in the Mix DC and then transferring them to the bottling center adds an extra layer of handling, potentially increasing labor costs and handling time. Moreover, if there's a surge in demand at the bottling center, the time required to fetch the mix containers from the Mix DC might cause delays. Finally, holding all mix containers in the Mix DC increases the storage burden. This could result in higher storage costs if the Mix DC has limited capacity.

b) Mix containers that are about to be used in H hours are sent directly from the mixing center to the bottling center without being stored in the Mix DC.

   Containers that are needed soon go directly from the mixing center to the bottling center, eliminating the need for intermediate storage and reducing labor and handling costs. Moreover, direct transfer of containers to the bottling center improves responsiveness, especially when there's an immediate need for specific mixes. This can prevent bottlenecks in bottling operations.

   However, there's a risk that mixes waiting in the Mix DC may not be used in the correct order, as some containers may bypass the DC entirely and go straight to the bottling center. This could lead to mixes being stored for too long. Additionally, misjudging demand or the H-hour window could lead to imbalances. A lot of coordination between the mixing and bottling centers is required to ensure smooth operations

We would recommend practice (b) if the bottling center's demand is relatively stable and there is a reliable forecasting system. It minimizes handling costs and speeds up the bottling process by sending mixes directly to where they are needed. Additionally, it optimizes storage by only holding excess inventory at the Mix DC. We would recommend

practice (b) if there is greater uncertainty in demand or concerns about the perishability of mixes. This method also simplifies decision-making, as workers follow a straightforward process without the need for dynamic decision-making.

TASK 15

For this task, we now have the possibility of storing mixes in containers which will help us smoothing the mixing production and therefore reducing the costs associated with high demand. Indeed, we offer FruitSoul the possibility to produce as much as possible every day, making sure we do not produce more than the yearly demand. This way, we will be able to limit the number of workers by producing the same amount over the year. If possible, we will bottle this mix and if not, it will go into storage. We will proceed as before for bottling and packing.

Here are the results:

	PCM	PCB	PCP	M	B	P	Paid	cost without materials	%unsatisfied demand M	%unsatisfied demand B	%unsatisfied demand P	# racks for 8 ounces jars	# racks for 16 ounces jars	# racks for 32 ounces jars
2021	3	2	4	6	4	13	76	2,736,604	3%	2%	9%	16	10	62
2022	3	3	4	21	12	20	326	9,233,414	0%	0%	8%	61	41	189
2023	6	4	8	30	28	32	69	9,468,015	0%	0%	9%	90	38	297

Figure 45: Final results for task 15

We still need a significant amount of racks, especially for 32oz jars. Nevertheless, we have succeeded in meeting more demand than before with all the demand being met for mixing and bottling in 2022 and 2023. We can improve the satisfaction of packing by either hiring more employees or buying more process centers.

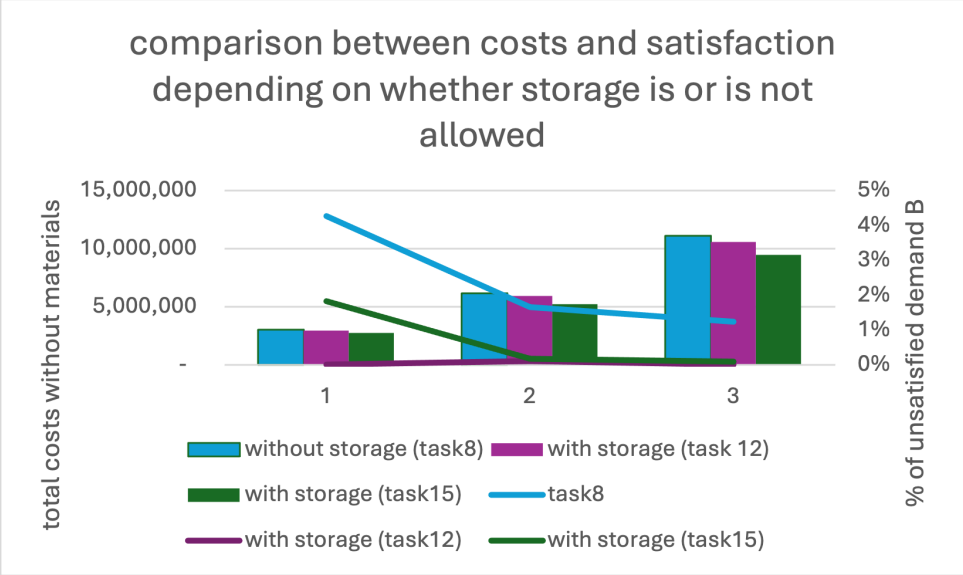


Figure 46: Comparison between costs and satisfaction depending on whether storage is or is not allowed

Indeed, since task 8, we have succeeded in reducing costs by 13% in average and

meeting most of the demand, which will result in higher profit since more sales will be performed. We have also succeeded in making sure that B and M employees are always working (100% of the time), but that means that most full time P workers do not work every day.

TASK 16

FruitSoul is a thriving company that has grown constantly over the last three years. It has taken advantage of the explosion of the fitness industry consequent to the end of Covid 19 and the surge of social media fitness influencers. The critical period for the company occurs during thanksgiving, when the demand greatly increases in a couple days, before returning to a 'normal' level at the start of December. During the span of a year, FruitSoul can expect to have 2 or 3 peaks in the demand (including the Thanksgiving weekend/Black Friday).

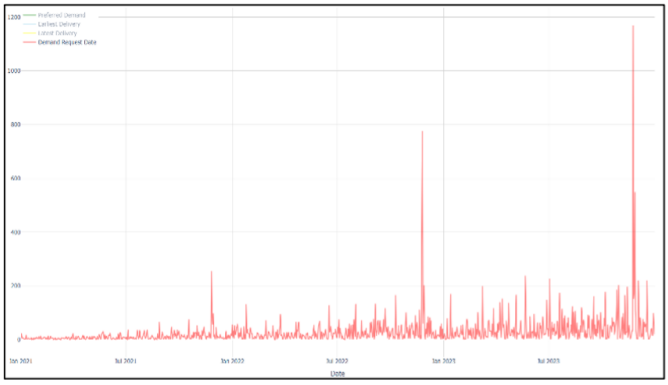


Figure 47: Fruit Soul Date of Order - Last 3 years

The bad news with high variability of the demand is that it requires from the company permanent adjustments. However, the good news is that the demand is seasonal over the span of one year. Therefore, it is easier for FruitSoul to prepare and adjust for key periods.

A strategy that proved to be effective was to smoothen the curve of the demand when it was getting to high. The idea is to spread out as much as possible the production and the shipping but continuing to deliver in the specified dates. The client might not receive his products at his preferred shipping date but it will be easier from a Mixing, Bottling and Packing capacity standpoint.

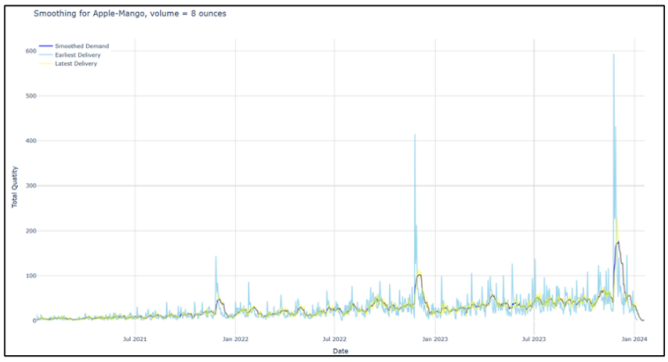
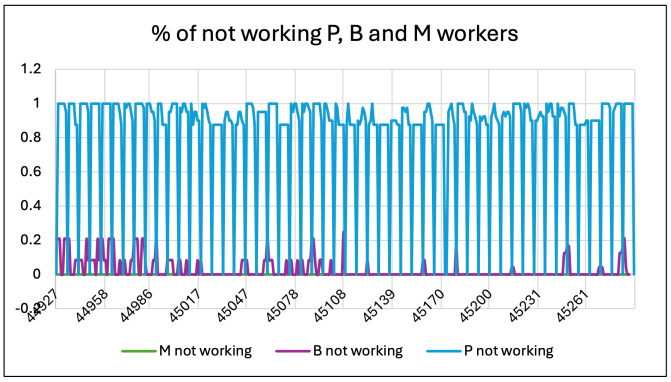


Figure 48: Earliest, Latest & Smoothed Demand, Apple Mango 8 ounces

The major difficulty for FruitSoul is to manage effectively it's employees and production capacity. The objective for every company is to satisfy a maximum of the demand at minimal cost. However, the high volatility of the Demand Log requires FruitSoul to aim for a high production capacity to buffer against variability. Another way to buffer against variability – and this is what we highly recommend FruitSoul to do – is to change its' stockless policy and massively invest in warehousing equipment. Racks and different layout possibilities have been studied during this casework, and it seems perfectly feasible to implement it in the company's strategy.



The % of not working P, B, M workers

Another issue that must be addressed in the company is the overall cost associated with full-time employees. For much of the year, their productivity is well below 100%. A potential solution would be to increase the number of part-time employees. To make this feasible, tasks could be simplified, allowing for the recruitment of less specialized staff. For instance, the Mixing process could be divided into two subsections—one requiring a high level of expertise, and the other much less. This would allow us to increase the share of part-time employees in each sector. While this strategy may potentially impact the company's image and values, it would lead to significant cost savings.

TASK 17

- Technical learnings Throughout this casework, we developed and mastered a wide range of technical skills. We worked on data preprocessing, allowing us to simplify analytics. We learned to conduct detailed data analytics over large datasets, making use of both Pandas and Excel to draw insights. We improved in cross referencing different data sets, and we combined Excel and Python to manage and analyze complex samples.

  Additionally, we applied heuristic optimization to minimize costs and tackled the challenge of yearly scheduling, aligning production and resources with demand.

  We worked on smoothing demand and production under constraints with Python, exploring different scenarios and constraints, this was particularly interesting and rewarding since we managed to develop and compute a smoothing algorithm by ourselves.

  Data visualization also became a strong skill—whether using Plotly, Matplotlib, or Excel—and we made sure our visualizations were user-friendly, sometimes by creating interfaces with Tkinter.

- Organizational financial learnings

  On the organizational side, we developed an approach to decision-making that leveraged KPIs to guide our choices. We explored multiple scenarios to see how different variables impacted our outcomes, giving us a solid foundation for making decisions even under uncertainty. We enhanced our financial forecasting and analysis using Excel, as well as decision analysis.

- Overall learnings

  One of the major learnings from the casework was understanding how to buffer against volatile demand, using warehousing solutions and by taking demand seasonality into account. We also learned how to design warehouse racks, considering the specific constraints we faced in this project. Finally, we learned how to find a balance between satisfying a certain percentage of demand and minimizing overall costs, discovering that "sweet spot" that made our solution both effective and efficient. Having a nonlinear production capacity made this last part challenging, but also very rewarding when we started converging toward what seemed to be an almost optimal solution.