Operations & Supply-Chain Management | 2024 Project

An Adaptative Service to Decrease Food Waste in French University Restaurants

**KARIM Mohamed Adnane - MARION Jim - RADIGUE Alexandre
ROUBEROL Paul - VIOLLET Baptiste**

Supervisors

ABDIN Adam
EL BECHARI Soufiane
LAMÉ Guillaume
MAHMOUD Ayman
NIFFOI Louis

Contents

Chapter 1

Introduction

Food waste is a critical issue, with an estimated one-third of global food production going to waste. In France, food waste amounts to slightly over 10 million tons, leading to a loss of 16 billion euros annually [3][8].

Concerned by this critical issue that is encountered on a daily basis, our primal goal was to tackle it by designing this service. This led us to consider and reconsider the aspects of the operation management in the French university restaurants.

A study involving 2,375 students revealed that more than half of the students in Francee go to a university restaurant at least once week. Annually, these university restaurants serve 65 million meals across 600 locations, employing 5,500 staff members. They purchase 32,000 tons of food with a total budget of 100 million euros [8].

However, the operations of university restaurants can fluctuate significantly, both over weeks and months, turning crowded to empty cafeterias. Thus being able to apply lean management principles to reduce waste and financial losses could be a real boon for the CROUS. Indeed, taking place in a Kaizen methodology with a PDCA (Plan-Do-Check-Act) quality management, it could reduce the cost with a growing quality of the service [1][3][4][6][7].

A study led by Swedish researchers established that 17,5% of prepared food in student or business canteen ends up in waste with 2,2% in kitchen waste, 11,3% in serving waste and 3,9% in customer leftovers [9]. Drastically reducing this unreasable amount of food waste appears to be vital in a global context of ecological sobriety.

Therefore our goal is to adapt the operations of French university restaurants to a lean management with reduced waste and a best use of purchasing and quantity management. With the use of tracking, monitoring and forecasting, alongside with basic awareness tools, we aim to propose a viable solution that could easily be implemented at all scales.

Our aim is to enable large scholar restaurants to automate the quantity and proportions chosen, according to a predictive follow-up AI algorithm using data tracking of waste, among others. This service would be a decision-making tool to monitor, simplify the processes and limit the waste.

After presenting our newly designed service, we will explore how its components have been separately studied and implemented in Sweden, where more than 75% of people eat at work or at scholar restaurants on a daily baisis. The financial consequences of implementing such a service will also be considered, wether at the scale of a specific university restaurant or at the broader scale of the CROUS. Lastly, a proposition of concrete forecasting solutions adapted to scholar restaurants alongside with modelling guidance were made at the end of this article, to test the strength of the designed service.

Chapter 2

Our Service

2.1 An adaptative toolbox to reduce food waste

Our study focuses on university restaurants. The goal of our service is to offer a comprehensive and personalized toolbox to each university restaurant to drastically reduce their food waste. Indeed, according to a study conducted by CROUS in Dijon, a university restaurant serving 800 to 1000 people generates nearly €60,000 in food waste ***(c.f. appendix 1)***. €60,000 is the cost of an entire year of meals at a CROUS for almost 300 scholarship students!

In a context of ecological concerns, state economy, and student austerity (each year the French State finances CROUS with €250 million to allow students to eat at low prices), it seems more important than ever to address this issue of food waste management, in order to reduce it as much as possible at CROUS facilities.

To achieve this, it is essential to understand where food waste in a university restaurant comes from. There are three main categories [5][8][9]:

- **Production waste**: raw materials not processed or lost during kitchen stages (e.g., peelings).
- **Service waste**: all dishes made available during a service but not consumed.
- **Plate waste**: leftovers on plates when clearing, not consumed.

However, according to numerous studies on the subject [5][8][9], it appears that the main categories responsible for overall waste are the latter two. Indeed, we can consider that university restaurants have a very good understanding of the transformation of their raw materials, and their production processes are designed to minimize production waste. Therefore, in the context of our study and service, we will focus solely on service waste and plate waste.

Nowadays in France, we certainly have a good understanding of the subject in terms of quantification, but we are still far from our Swedish neighbours who have managed to correctly grasp the issues related to this problem, while offering both qualitative and quantitative solutions.

After an in-depth study of their analyses and results (presented in the next chapter of this report), our solution aims to bring together the most impactful solutions to drastically reduce food waste in all types of French university restaurants.

This service is presented in the form of a toolbox that includes several tools for these canteens, such as:

- Quantification methodologies.

- Awareness tools.
- Trackers.
- Forecasting tools.

The complexity of this service lies in the fact that, in reality, even if each CROUS faces serious food waste problems, the underlying reasons for this are different. That's why, although methodologies and tracker tools can be generalized to all university restaurants, the forecasting and awareness tools must be defined specifically for each CROUS. That's why in our financial analysis section, we use a case study to evaluate our solution. However, it is clear that our service can be adapted to each university restaurant.

2.2 The proposed tools

In this section, we will qualitatively present the benefits and functionalities of each tool in our service. Quantitative evidence will be provided in the next chapter of this report.

2.2.1 Forecasting and predictive algorithms

Even though some university restaurants or canteens may have a "good understanding of the number of dishes to prepare," service waste remains very high ***(c.f. appendix 2)***. Therefore, the goal is to help these restaurants more accurately predict the number of students to serve on a given day and, consequently, the exact number of dishes to prepare to avoid unsold food being thrown away.

It has been shown that while the current operational system of university canteens (using predictive techniques like Optimal Proportion Quantity, such as the Hadley model) has a Mean Average Percentage Error (MAPE) of prediction between 20% and 40%, forecasting algorithms (predicting the number of students who will eat at a CROUS on a given day) have, in the best scenario, an error rate of 2 to 3% [1][3][7].

However, there are several predictive algorithms using different models:

- Benchmark.
- Prophet forecasting.
- Last-value forecasting.
- Moving average forecasting.
- Neural Network Models.

The key is to detect seasonality to choose the best model for a specific CROUS. One might think that selecting the best model for one restaurant would automatically apply to all other university restaurants. However, studies show that each restaurant has its own seasonality criteria. This is why our model is adaptable.

To implement an effective forecasting tool for a restaurant, several steps must be followed:

- Collect data on the occupancy of the university restaurant over a substantial period.
- Model the number of guests using different forecasting models, excluding known features.
- Select the best forecasting model, one with a low error rate while avoiding shortages.
- Implement it to assist in the preparation of the number of dishes.

With such a rigorous system, it is possible to drastically reduce service waste.

Why excluding known features? Simply using features like holidays, exam periods, and seasons results in predictions similar to what the CROUS already performs. Our goal is to intrinsically understand the patterns of a given CROUS.

Why avoiding shortage situations? Because a university restaurant cannot afford not to serve some students, and (especially for hot dishes) it is often impossible for university restaurants to prepare additional dishes during service. Therefore, it is crucial, when choosing the model, to ensure that the error rate is lower than that of the CROUS's previous predictive system while minimizing under-predictions as much as possible.

2.2.2 Trackers

Tracker tools focus primarily on plate waste and serve a dual purpose: awareness and prediction. These tools aim to measure, at the end of the chain, the weight of waste left on students' trays by categorizing and segmenting it (types of dishes, types of food, types of presentation, etc.). Today, several technologies enable these measurements, particularly using artificial intelligence with imaging.

Prediction purposes

Trackers collect valuable data on consumption patterns, which can be used to improve future meal planning. By analysing trends over time, these tools can help determine the appropriate portion sizes and identify the types of dishes that generate the most waste. This detailed insight is crucial for optimizing resource use and ensuring that the amount of food prepared closely matches the actual demand.

Indeed, university restaurants do not control the exact portions of their dishes, often serving portions that are 100 to 150 grams above the recommended nutritional values ***(c.f. appendix 2)***. By highlighting this issue and analysing which dishes are most likely to generate waste and the optimal quantities to serve, our service actively addresses the problem of plate waste.

All this data can either be analysed independently or included in the forecasting model for meal preparation and portion sizes. Trackers thus provide a better understanding of orders early in the production chain and insights into how to serve dishes better to reduce waste.

This method has already been used in France, notably with bread, which accounts for more than 10% of food waste in university restaurants. As a result, bread portions have been reduced from 50 grams to 40 grams in recent months.

In summary, by precisely analyzing portion sizes and waste patterns, our service enables university restaurants to optimize their meal preparation processes and significantly reduce food waste.

Awareness purposes

By displaying real-time data on food waste and its economic and environmental consequences to each student clearing their plate, tracking tools can be an essential element in raising awareness and combating waste. In Sweden, these tools are even interactive: in addition to indicating an individual's contribution to food waste, the tools also pose questions directly to students on a screen to better understand the reasons behind the waste.

2.2.3 Quantifications methods

For prediction models to be as accurate as possible, the data provided to the software must be both substantial and unbiased. Many guidelines have already been published on "how to measure the performance of a university restaurant most accurately"; it is just a matter of implementing them correctly.

There are several axes, including:

- Counting the students entering the restaurant.
- Counting the dishes produced, their compositions, etc.
- Counting the dishes served and not-served.
- Measuring the waste according to the three categories: production, service, plate.

A simple analysis over one year is sufficient to have a reliable model for the following year. By collecting data over the years, models can be refined, and other avenues for improvement can be considered, such as predicting raw material orders in advance, modifying dish compositions, etc.

2.2.4 Awareness tools

On the issue of plate waste, the challenge is to encourage each student to consume what they take entirely. This is an aspect that cannot be solved algorithmically. In addition to tracking tools, it is crucial that each university restaurant involves students in the fight against food waste.

What is the innovation here? Indeed, similar campaigns have already been implemented. Our approach aims to strengthen these campaigns while also offering "adaptive" strategies to each restaurant. With the data collected, our service allows us to better target the causes of food waste on served dishes and thus propose solutions more tailored to these causes. In other words, since each CROUS has different causes of waste, we offer campaigns that address their specific reasons (e.g., arguments based on the composition of students, targeted arguments on certain foods, etc.). This actively involves students and sensitizes them effectively, which can lead to more efficient and sustainable behaviour changes.

Chapter 3

State of the Art

On the issue of quantifying the impacts of intervention methods to reduce food waste in catering services, our Swedish colleagues are the most advanced. In order not to overwhelm the reader with information, we have selected 2 interesting studies from the Swedish University of Agricultural Sciences justifying the positive effects of our service. For each of them, we will present their context, results, and challenges.

3.1 Study 1 - Testing interventions to reduce food waste in school catering [5]

3.1.1 Global Context

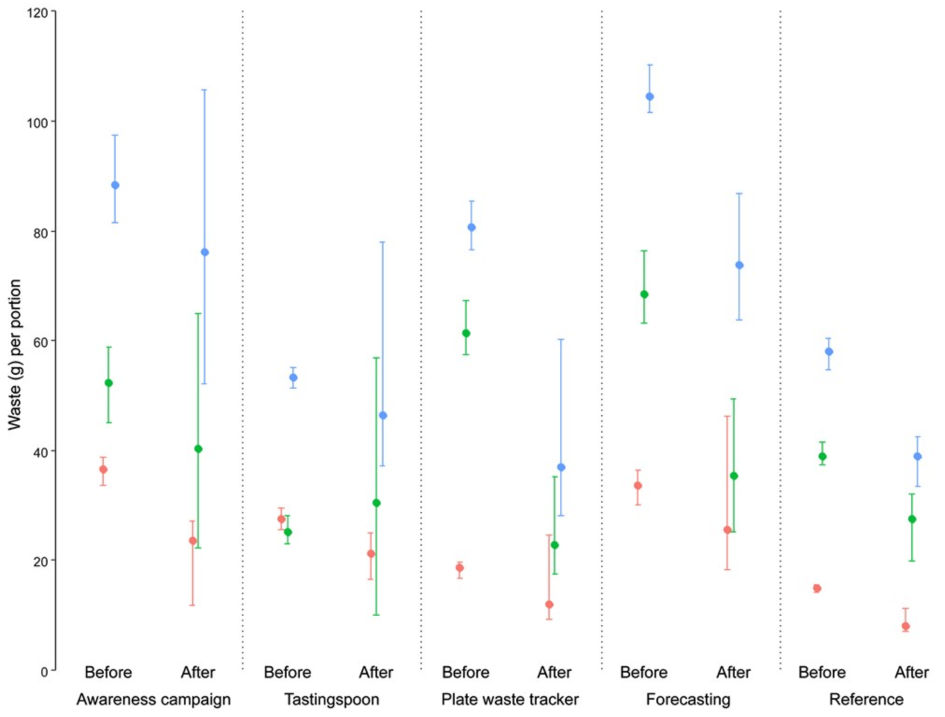
This study, highlighting the importance of combating food waste, measures the impact of various intervention measures on 15 Swedish school canteens on reducing said waste. With over 3222 observations, the initial finding is as follows: on average, food waste amounts from 66.8 g to 74.2 g per portion, with significant variability between each canteen, which observed between 4.9 g to 464 g per portion.

Four intervention measures were chosen for this study. To measure the difference in impact between each measure, each canteen was assigned one intervention, with 7 other canteens serving as the reference group. Those were the interventions:

- **Tasting Spoons**: the idea was to allow students to test a dish before taking it in order to reduce plate waste.
- **Awareness Campaigns**: the idea was to raise awareness on plate waste with traditional means such as posters, discussions, etc.
- **Plate-waste Trackers**: the idea was to increase the interaction with students in order to provide feedback on why they wasted food, with predetermined alternatives, in order to tackle plate waste.
- **Forecasting**: the idea was to anticipate the number of students coming to the school canteen in order to prevent serving waste, thanks to different predictive algorithms on the daily guests.

3.1.2 Results

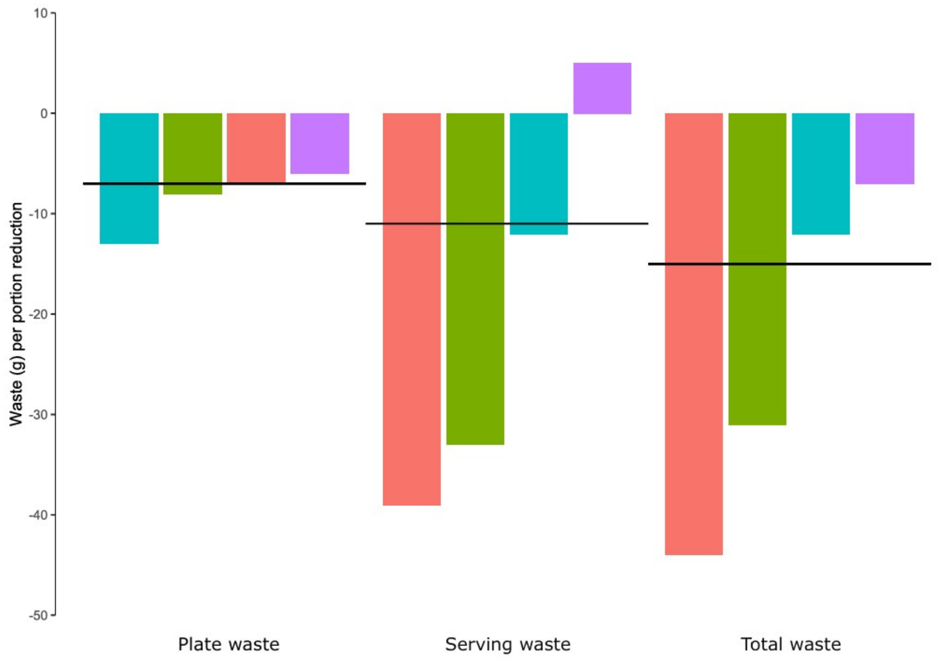
The findings from the different interventions, divided into total waste, serving waste and plate waste reduction per portion are presented in Figure 3.1.



Waste (g) per portion (median values with uncertainties as 95% confidence intervals) before and after implementation of measures to reduce food waste in school canteens. ● Total waste per portion, ● serving waste per portion, ● plate waste per portion.

Figure 3.1: Waste per portion before and after the implementation of the different measures

By examining, those results, it appears that forecasting measures and plate waste trackers are the most efficient ones, helping a decrease of total waste by 35% to 45%, as shown in Figure 3.2. However we also notice an interesting phenomenon: tasting spoons have a negative effects on serving waste, increasing it. And there is also another singular effect: even the reference group have seen its total waste reduced !



Median reduction in food waste (g) per portion after implementation of the interventions: 🟦 Awareness campaign, 🟩 forecasting, 🟥 plate waste tracker, 🟪 tasting spoons, in relation to the reference group ▬, for plate waste, serving waste and total waste per portion. Total waste also included waste from the kitchen if this was quantified.

Figure 3.2: Median reduction in food waste per portion after the implementation of the different measures

3.1.3 Discussions

The conducted study showed a reduction in total food waste per portion in a range from 6g to 44 g across the different interventions. Plate waste trackers and forecasting interventions achieved a reduction of 62 % and 49 % in serving waste, respectively. Which is a good source of hope for the French university restaurant system, as serving waste is one of the major issues.

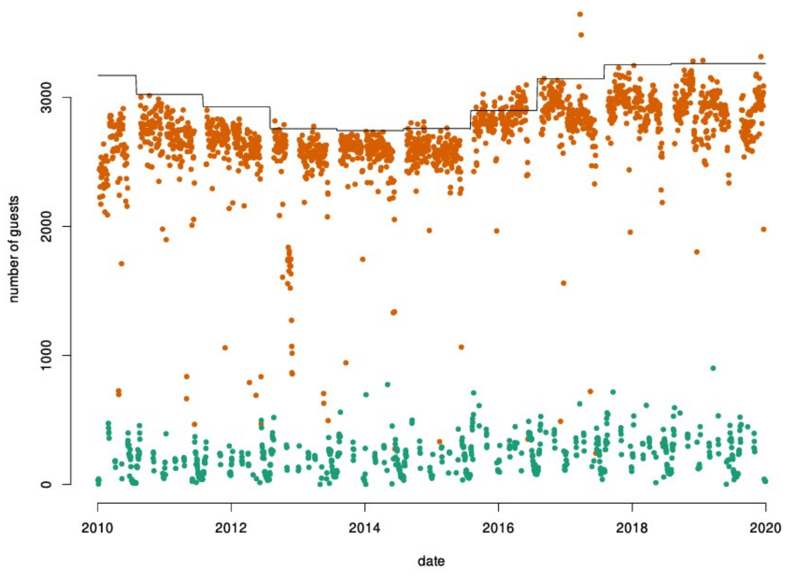
However this study needs now to be conducted in French Soil, at a randomised level, and targeting universities, as the patterns of food waste regarding young adults may be perceived as different from teenagers ones.

The study also concludes that a widespread of those simple tools in a global service may be the first and more impactful step towards tackling the food waste issue. However, it has been shown that schools may react differently to those, and that we are still lacking some understandings on the impact of meal compositions regarding food waste.

3.2 Study 2 - Potential for using guest attendance forecasting in Swedish public catering to reduce overcatering [7]

3.2.1 Global Context

This second study, also focusing on reducing food waste in Swedish school canteens, preferred to concentrate on one of the main factors of the problem: serving waste. The premise it formulates is that this waste is the result of poor prediction of the number of guests on a given day (t): an overestimation of this figure leads to a larger production, and therefore, inevitably, to unused and wasted dishes. By taking the filling data from about twenty school canteens over several years, this study aims to test the accuracy of different forecasting models on the daily number of guests in order to propose a better prediction solution for these canteens. The error indicator it seeks to minimize is the MAPE (Mean Absolute Percentage Error). The Figure 3.3 presents a visualisation of the collected data that was fed to the different algorithms. A pre-analyse of the seasonalities was made.



Number of guests over time at school kitchens in a municipality (containing data from 15 schools who had a complete data set of 10 years) where ● indicates normal day and ● indicates holiday with less activity, where the schools provides meals to other establishments within the public catering organization. The line shows the number of students enrolled, and can be taken as the maximum of guests that need to be provided with food.

Figure 3.3: Visualisation of the data used for prediction

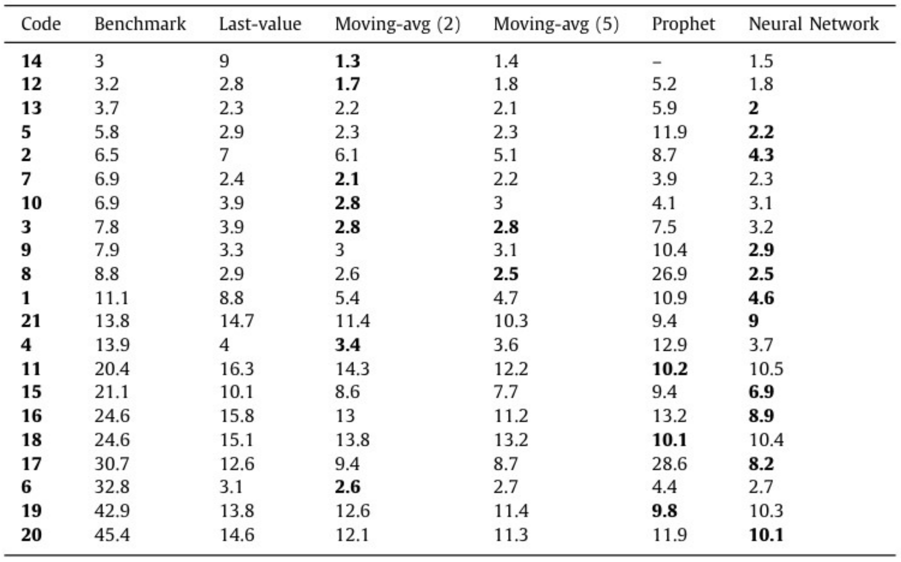
The tested methods or algorithms used to predict the daily guests number were:

- **The benchmark scenario**: original predictive methods used by the canteens, showing the average level of deviance regarding number of portions per day from the number of students enrolled per school year.
- **Last-value forecasting**: most-basic time-series forecasting method based on the observation of the value at t-1 to predict the value at t.
- **Moving-avergae model**: times-series forecasting method based on averaging the observed value on a certain time window.
- **Prophet forecasting model**: times-series forecasting method developed by Facebook, mostly used in the analysis of big-data bases.
- **Neural-Network Models**: more complex structure with a multi-layer network, with 32 neurons each.

The other challenge of this study, is not only finding the more suitable model for each canteen that reduces the prediction error, but also the one that prevents shortage. Indeed, shortage scenarios are the worst scenarios for a school canteen and cannot be allowed. That is why, after calculating the errors of each models, different forecasting margins were implemented in order to reduce the amount of underestimated days, while having avec MAPE error as low as possible. "It is easier to throw food than to cook new food, so a balance needs to be found to obtain a feasible margin that is acceptable and practical".

3.2.2 Results

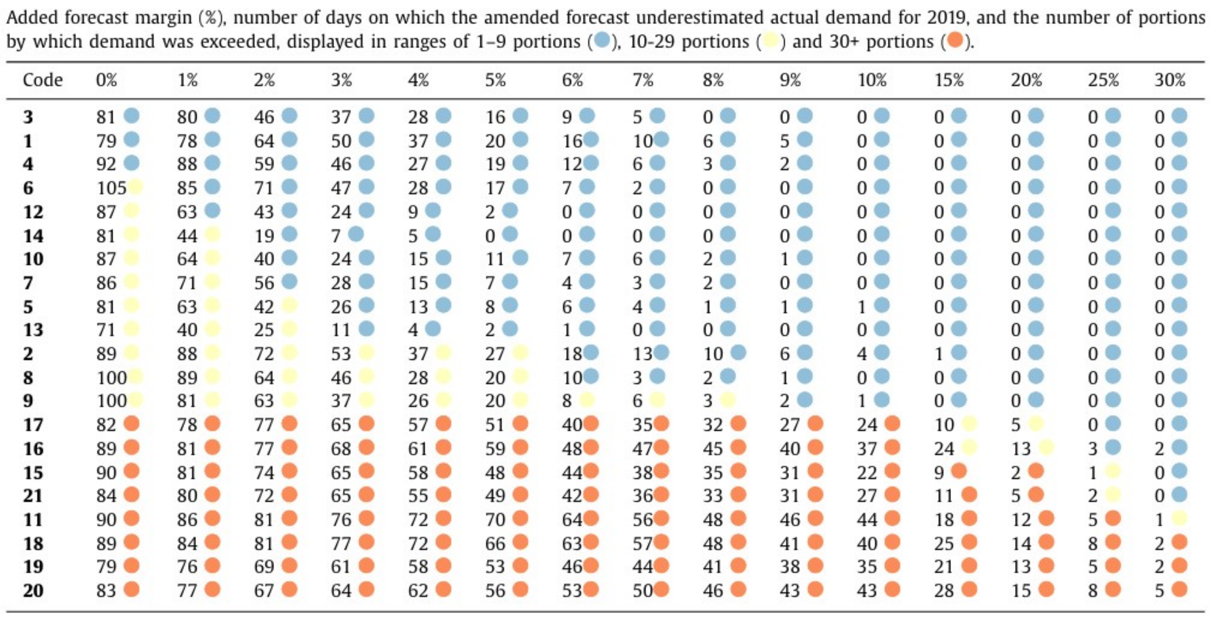
For every single canteen, it was shown that using a forecasting method was always better than using the classical benchmark scenario, in MAPE error, as seen in Figure 3.4.



Code	Benchmark	Last-value	Moving-avg (2)	Moving-avg (5)	Prophet	Neural Network
14	3	9	**1.3**	1.4	–	1.5
12	3.2	2.8	**1.7**	1.8	5.2	1.8
13	3.7	2.3	2.2	2.1	5.9	**2**
5	5.8	2.9	2.3	2.3	11.9	**2.2**
2	6.5	7	6.1	5.1	8.7	**4.3**
7	6.9	2.4	**2.1**	2.2	3.9	2.3
10	6.9	3.9	**2.8**	3	4.1	3.1
3	7.8	3.9	**2.8**	**2.8**	7.5	3.2
9	7.9	3.3	3	3.1	10.4	**2.9**
8	8.8	2.9	2.6	**2.5**	26.9	**2.5**
1	11.1	8.8	5.4	4.7	10.9	**4.6**
21	13.8	14.7	11.4	10.3	9.4	**9**
4	13.9	4	**3.4**	3.6	12.9	3.7
11	20.4	16.3	14.3	12.2	**10.2**	10.5
15	21.1	10.1	8.6	7.7	9.4	**6.9**
16	24.6	15.8	13	11.2	13.2	**8.9**
18	24.6	15.1	13.8	13.2	**10.1**	10.4
17	30.7	12.6	9.4	8.7	28.6	**8.2**
6	32.8	3.1	**2.6**	2.7	4.4	2.7
19	42.9	13.8	12.6	11.4	**9.8**	10.3
20	45.4	14.6	12.1	11.3	11.9	**10.1**

Figure 3.4: MAPE Error (%] values obtained for the different models compared against the benchmark scenario (%). Unreasonable results are indicated by -. The best model for each kitchen is high-lighted.

Figures 3.5 shows how often the forecast was underestimated, depending on the margin added on the model, for each canteens, and by how much.



Added forecast margin (%), number of days on which the amended forecast underestimated actual demand for 2019, and the number of portions by which demand was exceeded, displayed in ranges of 1–9 portions (🔵), 10-29 portions (🟡) and 30+ portions (🟠).

Code	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	15%	20%	25%	30%
3	81 🔵	80 🔵	46 🔵	37 🔵	28 🔵	16 🔵	9 🔵	5 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
1	79 🔵	78 🔵	64 🔵	50 🔵	37 🔵	20 🔵	16 🔵	10 🔵	6 🔵	5 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
4	92 🔵	88 🔵	59 🔵	46 🔵	27 🔵	19 🔵	12 🔵	6 🔵	3 🔵	2 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
6	105 🟡	85 🔵	71 🔵	47 🔵	28 🔵	17 🔵	7 🔵	2 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
12	87 🟡	63 🔵	43 🔵	24 🔵	9 🔵	2 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
14	81 🟡	44 🟡	19 🔵	7 🔵	5 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
10	87 🟡	64 🟡	40 🔵	24 🔵	15 🔵	11 🔵	7 🔵	6 🔵	2 🔵	1 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
7	86 🟡	71 🟡	56 🔵	28 🔵	15 🔵	7 🔵	4 🔵	3 🔵	2 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
5	81 🟡	63 🟡	42 🟡	26 🔵	13 🔵	8 🔵	6 🔵	4 🔵	1 🔵	1 🔵	1 🔵	0 🔵	0 🔵	0 🔵	0 🔵
13	71 🟡	40 🟡	25 🟡	11 🔵	4 🔵	2 🔵	1 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
2	89 🟡	88 🟡	72 🟡	53 🟡	37 🟡	27 🟡	18 🔵	13 🔵	10 🔵	6 🔵	4 🔵	1 🔵	0 🔵	0 🔵	0 🔵
8	100 🟡	89 🟡	64 🟡	46 🟡	28 🟡	20 🟡	10 🔵	3 🔵	2 🔵	1 🔵	0 🔵	0 🔵	0 🔵	0 🔵	0 🔵
9	100 🟡	81 🟡	63 🟡	37 🟡	26 🟡	20 🟡	8 🟡	6 🟡	3 🟡	2 🔵	1 🔵	0 🔵	0 🔵	0 🔵	0 🔵
17	82 🟠	78 🟠	77 🟠	65 🟠	57 🟠	51 🟠	40 🟠	35 🟠	32 🟠	27 🟠	24 🟠	10 🟡	5 🟡	0 🔵	0 🔵
16	89 🟠	81 🟠	77 🟠	68 🟠	61 🟠	59 🟠	48 🟠	47 🟠	45 🟠	40 🟠	37 🟠	24 🟡	13 🟡	3 🔵	2 🔵
15	90 🟠	81 🟠	74 🟠	65 🟠	58 🟠	48 🟠	44 🟠	38 🟠	35 🟠	31 🟠	22 🟠	9 🟠	2 🟠	1 🟡	0 🔵
21	84 🟠	80 🟠	72 🟠	65 🟠	55 🟠	49 🟠	42 🟠	36 🟠	33 🟠	31 🟠	27 🟠	11 🟠	5 🟠	2 🟡	0 🔵
11	90 🟠	86 🟠	81 🟠	76 🟠	72 🟠	70 🟠	64 🟠	56 🟠	48 🟠	46 🟠	44 🟠	18 🟠	12 🟠	5 🟠	1 🟡
18	89 🟠	84 🟠	81 🟠	77 🟠	72 🟠	66 🟠	63 🟠	57 🟠	48 🟠	41 🟠	40 🟠	25 🟠	14 🟠	8 🟠	2 🟠
19	79 🟠	76 🟠	69 🟠	61 🟠	58 🟠	53 🟠	46 🟠	44 🟠	41 🟠	38 🟠	35 🟠	21 🟠	13 🟠	5 🟠	2 🟠
20	83 🟠	77 🟠	67 🟠	64 🟠	62 🟠	56 🟠	53 🟠	50 🟠	46 🟠	43 🟠	43 🟠	28 🟠	15 🟠	8 🟠	5 🟠

Figure 3.5: Number of days on which the amended forecast underestimated the actual demand, based on the added forecast margin

3.2.3 Discussions

The study showed that using forecasting methods had promising results towards evaluating guest attendance with accuracy in school canteens. However, reliable data is crucial for effective forecasting: thus flaws and uncertainties needs to be controlled on the reported data. Therefore, school canteens may have to review their organizational structure and improve communication in order to enhance forecasting accuracy.

The study emphasize that there is no universal solutions: strategies need to be tailored to individual canteens. A solution to counter overcatering may be combining forecasting with long-life backup stocks approach.

Lastly, the implementation of a forecasting model in a school canteen has to be done iteratively: beginning with simple forecasting techniques and preparation to handle shortages, and then moving to more accurate ones.

Chapter 4

Financial Consequences

4.1 Calculation of the Annual Savings Achieved by a University Restaurant (RU)

In this section, we will try to estimate the amount of money that the CROUS can expect to earn by implementing one of the two solutions.

The notations will be as follows:

- P: the price of one gram of raw material (food) in €.g^{-1};
- β: the percentage reduction in waste;
- $\bar{m}_{\text{waste}}$: the average mass of waste per person;
- N: the number of people eating at the university restaurant per day.

The calculation is as follows:

$$\begin{aligned}
\text{Savings per day} &= \text{mass of waste avoided} \times P \\
&= (\text{mass of waste before solution} - \text{mass of waste after solution}) \times P \\
&= \text{mass of waste before solution} \times \beta \times P \\
&= N \times \bar{m}_{\text{waste}} \times \beta \times P
\end{aligned}$$

Thus, to obtain the savings achieved over a year, this result must be multiplied by the number of days the RU is open. Over 52 weeks, the restaurant is closed for 9 weeks for summer vacation, 1 week for All Saints' Day, 2 weeks for Christmas, and 2 weeks for February and April vacations. Thus, the RU is open 5 days a week for 38 weeks.

$$\text{Annual savings} = 5 \times 38 \times \text{Savings per day}$$

For numerical applications, we will take [2]:

$$P = 0.00612 \text{ €.g}^{-1} \text{ and } \bar{m}_{\text{waste}} = 120 \text{ g}$$

Finally, we will consider that a university restaurant serves about 800 people per day (a magnitude corresponding to a RU like the one at ENS Paris-Saclay).

4.1.1 Case 1 - Implementation of a Tracker

In the case of implementing a tracker, we find [5]:

$$\beta_1 \approx 56\%$$

The annual savings ΔE_1 achieved in a year is then:

$$\Delta E_1 \approx 36\ 800\ €$$

4.1.2 Case 2 - Implementation of Forecasting

In the case of implementing forecasting, we find [5]:

$$\beta_2 \approx 33\%$$

The annual savings ΔE_2 achieved in a year is then:

$$\Delta E_2 \approx 62\ 500\ €$$

4.2 Analysis of Results & Perspective

To make these savings more tangible, we will compare them in two ways: relative to the turnover of the RU considered and relative to the price of the meal.

4.2.1 Comparison to RU Turnover

The price of a meal prepared at the RU costs about 7 € to the CROUS. This takes into account all expenses related to meal preparation: cost of raw materials, personnel costs, etc.

The turnover of the university restaurant is:

$$\text{RU Turnover} = 5 \times 38 \times 800 \times 7$$

And we then find RU Turnover = 1 064 000 €.

Thus, the savings enabled by implementing forecasting represent about 3.5% of the RU's turnover. Implementing a tracker allows for savings representing 5.8% of the establishment's turnover.

4.2.2 New Equivalent Price

The idea is to calculate the new price the RU could offer students if it passed the savings made onto meal prices. This amounts to calculating the savings per meal.

In the case of tracking, it is sufficient to calculate:

$$E = \frac{62\ 500}{5 \times 38 \times 800} \approx 0.40$$

Thus, by adopting tracking, the CROUS could offer meals at 2.90 €!

In the case of forecasting, it is sufficient to calculate:

$$E = \frac{36\ 800}{5 \times 38 \times 800} \approx 0.24$$

Thus, by adopting forecasting, the CROUS could offer meals at 3.06 €!

Chapter 5

Forecasting and optimization technologies

To reduce waste in university dining services like a CROUS, leveraging an algorithmic approach integrated with supply chain strategies can focus on optimizing orders based on consumption forecasts and waste tracking. We developed a structured solution combining data analysis, algorithmic decision-making, and supply chain management. Despite the lack of real data on this market, we ended up providing some steps for the implementation of the solution after brainstorming on the possible solutions.

5.1 Forecasting Model

In this section we will present the Forecasting algorithm used to predict the consumption forecast, namely the number of consummer students for the next day, based on the seasonality and the trends of the previous days.

5.1.1 Data Collection and Analysis

Before developing algorithmic solutions, it is crucial to gather accurate data on consumption and waste. The platform Kaggle was very useful to get an insight on different datasets.

The necessary data are the following:

- **Daily Consumption**: Quantity of each type of dish consumed per day.
- **Waste by Type of Dish**: Quantity of waste generated by starters, main courses, and desserts each day.
- **Seasonal Data**: Seasonal variations in food preferences and attendance.

5.1.2 Predictive Modelling for Consumption

We could use machine learning techniques to predict future consumption based on historical data.

In order to implement the predictive modelling for consumption we could use several forecasting models such as *Random Forest, ARIMA, SARIMA*, or *LSTM neural networks* to predict future demand based on historical data. We chose to use the commonly used forecasting model *Prophet* in our model.

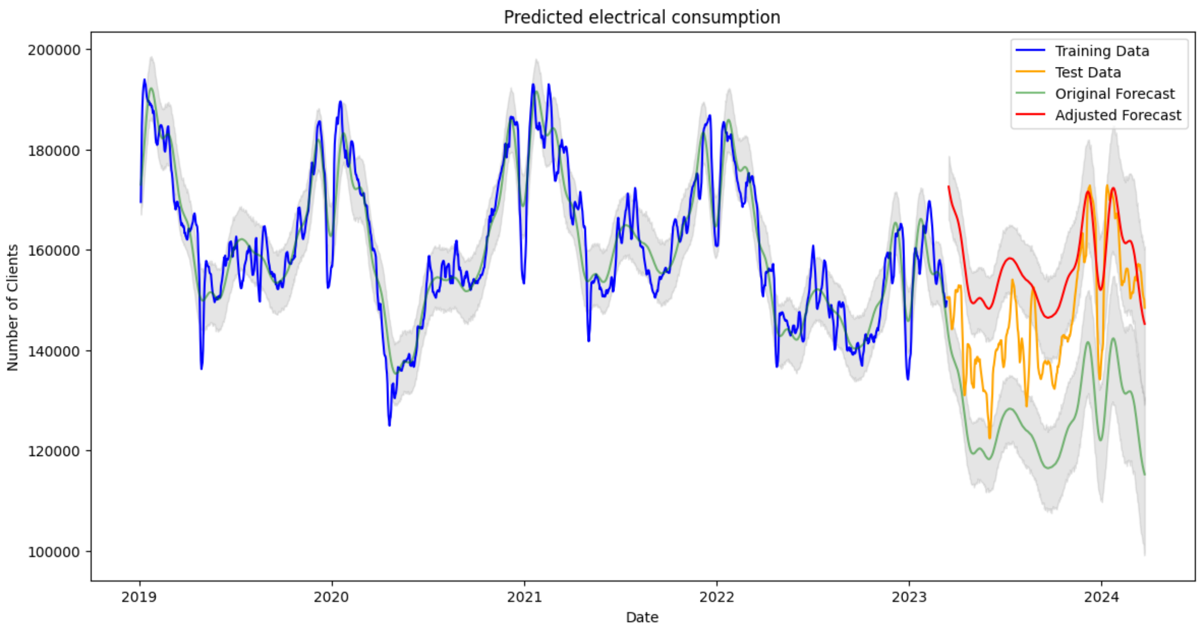
Then a cross-validation is necessary to refine models and prevent overfitting.

You will find in the Appendix 3 below the code developped to predict the future consumption in a CROUS based on the previous consumption data and seasonality.

Despite the absence of any dataset on university restaurants, we chose an initial data base on Kaggle that displays the electrical consumption over months.

You will find below the results of the previous code.

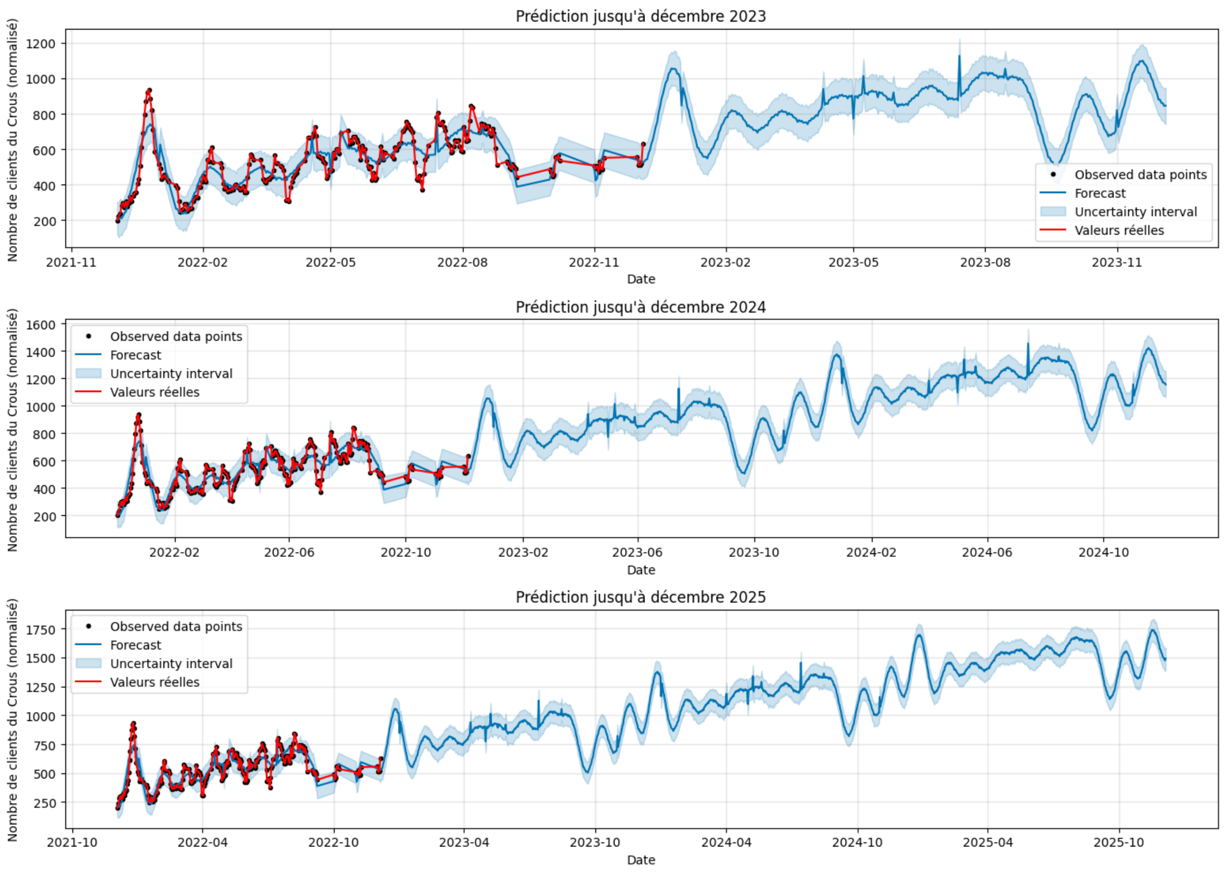
We trained the model on the Training Data (blue) for the past years and we predicted the trend of the Testing Data for the following years (orange). The results of this prediction are highly satisfactory since the prediction or forecast (green) nearly follows the trend of the testing dataset (orange). However the forecast underestimates in this case the original results. Thus we adjusted the forecast using a simple offset in order to slightly overestimate the Testing Data. Indeed, it is better to overestimate the number of consumers rather than facing shortages when it comes to serving the daily meals. The red curve representing the adjusted forecast overestimates the Testing Data (orange) which is a great result for the supply chain model and for the following optimization model. Thus each CROUS might want to fine-tune this offset to better forecast the real demand according to their needs.



While the previous dataset was displaying the electrical consumption over several years, we wanted to take a more realistic dataset for predicting the demand in a university restaurant. We found a dataset on a German Restaurant and we implemented our algorithm on it in order to predict the demand in the following months.

You will find in the Appendix 4 below the algorithm for the Forecast Model of the German Restaurant.

Here are the results of the predicted demand using the dataset from the German restaurant. We can clearly observe the seasonnal trends of the prediction compared to the Training Data. The forecasting model is meant to predict the weekly, monthly, and yearly seasonalities. Indeed, we wanted to show that during the weekend or the holidays there might be less students having lunch at the CROUS leading the prediction to underestimate the demand. Finally we want to remind that the initial aim was to reduce the amount of wastes after each day. This prediction model will clearly help them reduce their wastes.



5.1.3 Continuous Evaluation and Adjustments

The implementation of periodic reviews is compulsory to assess the effectiveness of strategies and make adjustments based on new consumption trends or feedback.

We will have to review the analyse consumption and waste data periodically in order to identify new trends or changes in eating behaviours. The model might be adjusted by updating the parameters of forecasting and optimization models to adapt to new data.

By combining an algorithmic approach with supply chain management strategies and a strong component of awareness, university dining services can significantly reduce their waste production.

5.2 Optimization Model

Once consumption demand is forecasted, we could use optimization algorithms to adjust orders to minimize waste while meeting anticipated demand.

We propose a method of Linear/Integer Programming. The aim is to develop an optimization model to minimize the difference between ordered quantities and predicted consumption, considering constraints like storage capacity, product shelf life, or the reduction of wastes amount.

5.2.1 Architecture of the Optimization Model

Here is the code we have implemented for the Optimization Model using the module Gurobi. It optimizes the number of orders made for day N+1 based on the forecasted demand (*forecasted_ demand*) of students from days N, N-1, N-2, etc.

We chose to separate the orders in three categories, namely starters, main courses, and desserts each day because they might have differents constraints and parameters such as the shelf life or the storage cost.

Several **Constraints** must be adhered to. We must take into account the **storage costs** (the more we order, the higher the storage costs are). We also need to consider the **amount of wastes** generated by such an optimized model in order to totally reduce it.

Furthermore, we have highlighted the factor of **expiration dates** which greatly influences the orders from one day to the next. Indeed, a dish that expires in one day is penalized more severely, and stocks must therefore be replenished daily, unlike a dessert that can be reused over several weeks, provided it is a yogurt or another dessert with a long shelf life. Thus, we tend to order desserts in larger lots, and prepare main dishes on the same day. Starters, on the other hand, have an intermediate status as they can be reused for a maximum of two or three days after preparation.

We define the **Objective Function** to maximize the benefits by taking into account the wastes :

$$\text{Maximize} \quad \sum_{\text{category} \in \text{categories}} \sum_{\text{day} \in \text{days}} \Big[\text{consumptions}_{\text{category,day}} \times \text{sale\_price} \\ - \text{orders}_{\text{category,day}} \times \text{prod\_cost} \\ - \text{stocks}_{\text{category,day}} \times \text{stock\_cost}_{\text{category}} \\ - \text{wastes}_{\text{category,day}} \times \text{waste\_cost} \Big]$$

$$\text{Maximize} \quad \sum_{\text{c} \in \text{categories}} \sum_{\text{d} \in \text{days}} \Big[\text{consumptions}_{\text{c,d}} \times \text{sale\_price} - \text{orders}_{\text{c,d}} \times \text{prod\_cost} \\ - \text{stocks}_{\text{c,d}} \times \text{stock\_cost}_{\text{c}} - \text{wastes}_{\text{c,c}} \times \text{waste\_cost} \Big]$$

where:

- $\text{consumptions}_{\text{category,day}}$ represents the consumption for a given category and day.
- sale_price is the selling price per unit.
- $\text{orders}_{\text{category,day}}$ represents the orders for a given category and day.
- prod_cost is the production cost per unit.
- $\text{stocks}_{\text{category,day}}$ represents the stocks for a given category and day.
- $\text{stock\_cost}_{\text{category}}$ is the storage cost per unit for a given category.
- $\text{wastes}_{\text{category,day}}$ represents the wastes for a given category and day.
- waste_cost is the cost associated with wastes per unit.

The numerical data used are only estimates taken by us to confront the model with reality.

You will find in the Appendix 5 the algorithm developed to optimize the orders for the three categories.

We calculated the amount of waste with and without this optimization model in order to evaluate the benefits of such an algorithm for the efficiency of the supply chain.

5.2.2 Results of the Optimiation Model

You will find below the results of the optimization model.

Graph 1

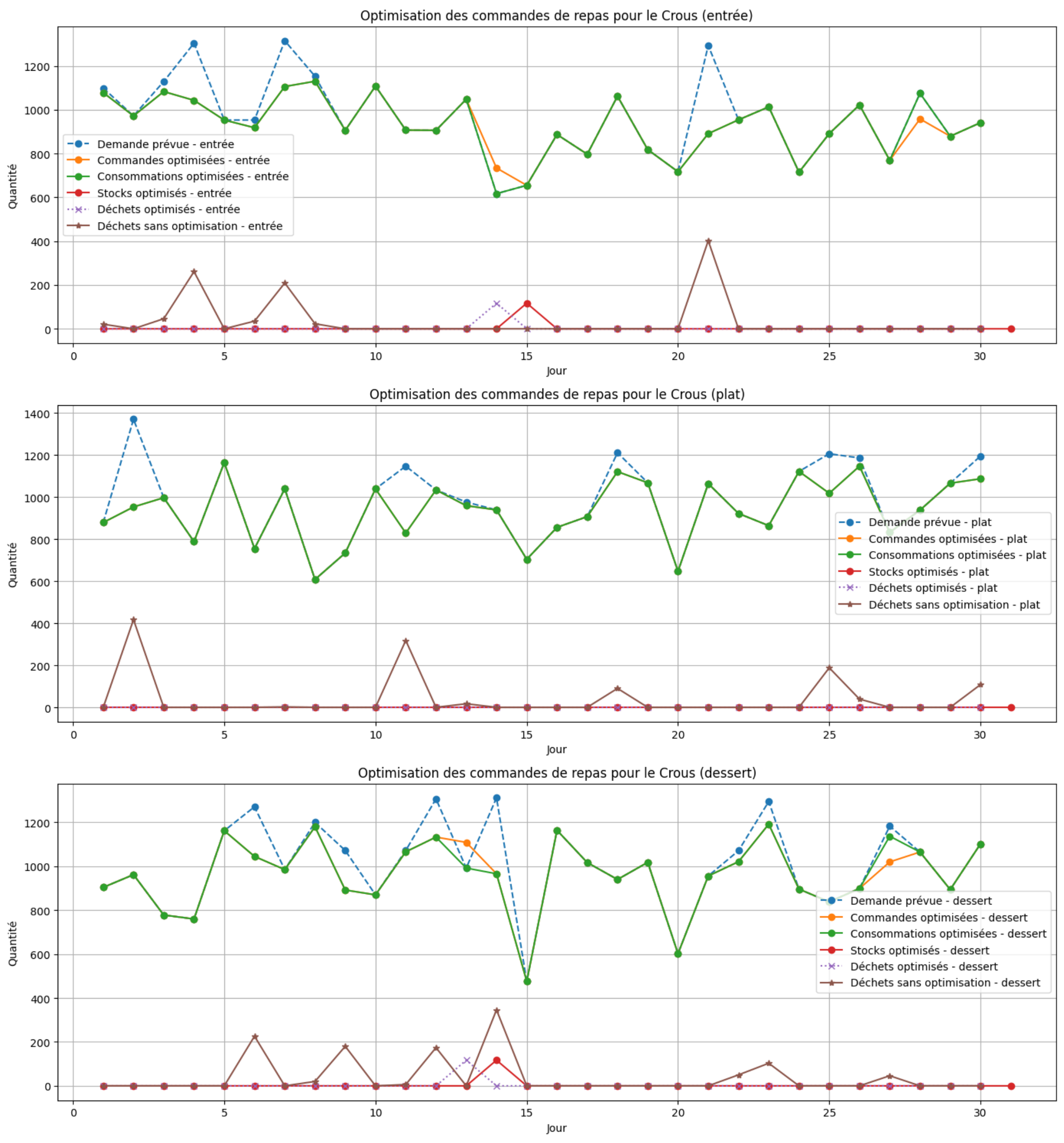
- Starter: 14 days (shelf life)
- Main dish: 1 day
- Dessert: 14 days

The first graph shows the optimization of orders for starters, main dishes, and desserts with perishability periods of 14 days for starters and desserts, and 1 day for main dishes. It can be observed that the optimized stocks and optimized wastes remain low compared to the optimized orders and consumptions. This configuration significantly minimizes waste.

Comparison of average wastes (before vs after optimization):

- **Starter:**
  - Before optimization: 33.23
  - After optimization: 3.90
- **Main dish:**
  - Before optimization: 39.30
  - After optimization: 0.00
- **Dessert:**
  - Before optimization: 38.37
  - After optimization: 3.90

The reduction in waste is very significant, especially for main dishes where the waste after optimization is completely eliminated. For starters and desserts, there is a significant reduction in waste after optimization. The optimization model helps reduce 88.3% of starter wastes, 100% of main dish waste (impossible in practice because we did not take into account the wastes due to student who did not finish their plates) and 89.8% of dessert wastes.



Graph 2

- Starter: 7 days
- Main dish: 1 day
- Dessert: 14 days

The second graph presents an optimization with the perishability period reduced to 7 days for starters, while maintaining 1 day for main dishes and 14 days for desserts. The results show an increase in waste for starters after optimization compared to the first case, but still a significant reduction compared to the non-optimized situation.

Comparison of average wastes (before vs after optimization):

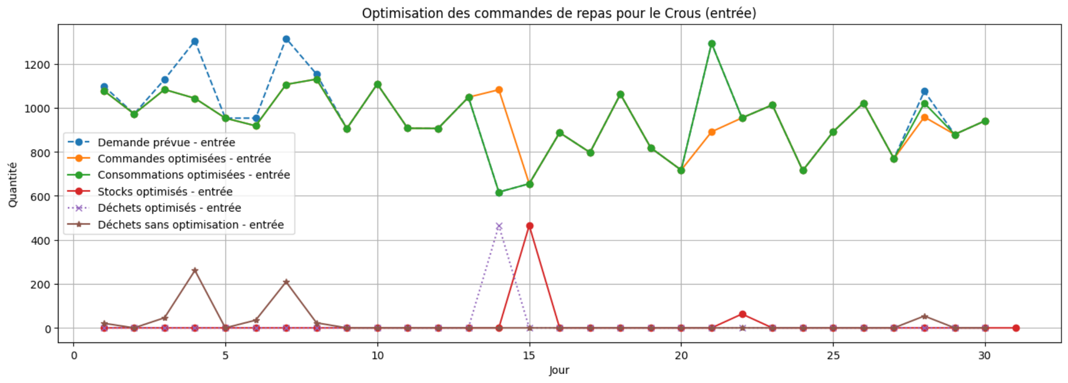
- **Starter:**
  - Before optimization: 21.63

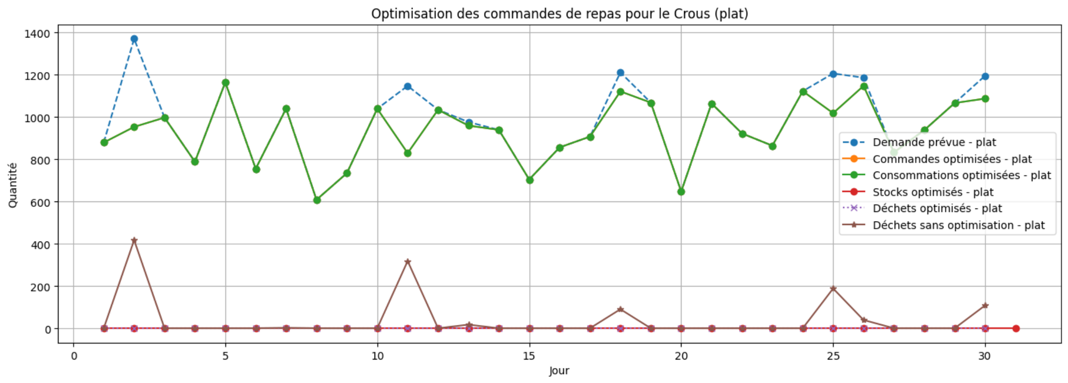
  - After optimization: 15.50

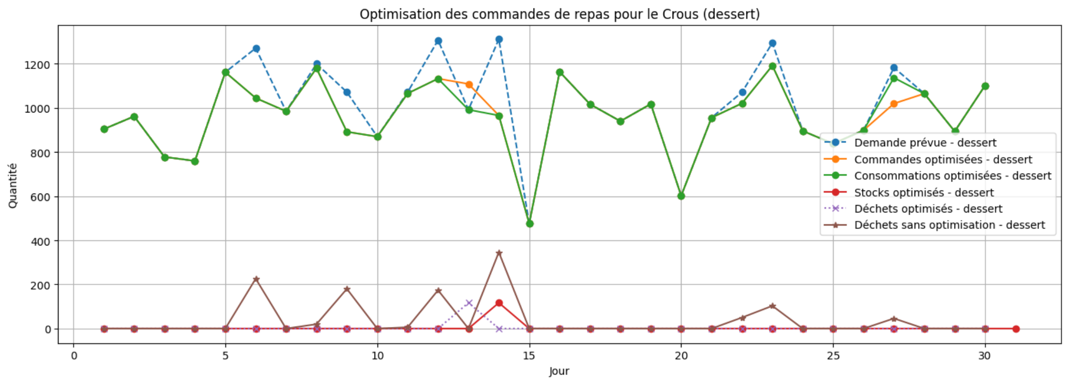
- **Main dish:**
  - Before optimization: 39.30
  - After optimization: 0.00

- **Dessert:**
  - Before optimization: 38.37
  - After optimization: 3.90

Although the waste for starters is higher than in the first case, it is still significantly lower than the levels before optimization. The main dishes continue to generate no waste after optimization, and the desserts show a similar reduction to that observed in the first case. The optimization model helps reduce 28.3% of starter wastes, 100% of main dish waste (impossible in practice because we did not take into account the wastes due to student who did not finish their plates) and 89.8% of dessert wastes.







Graph 3

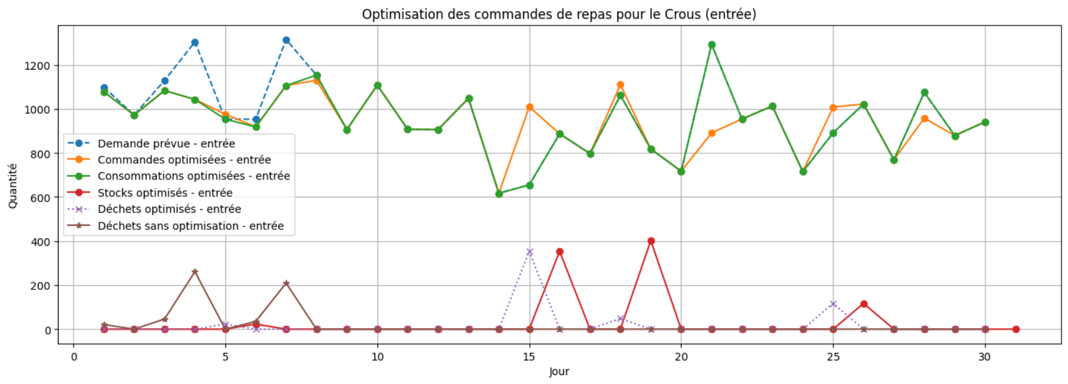
- Starter: 3 days
- Main dish: 1 day
- Dessert: 7 days

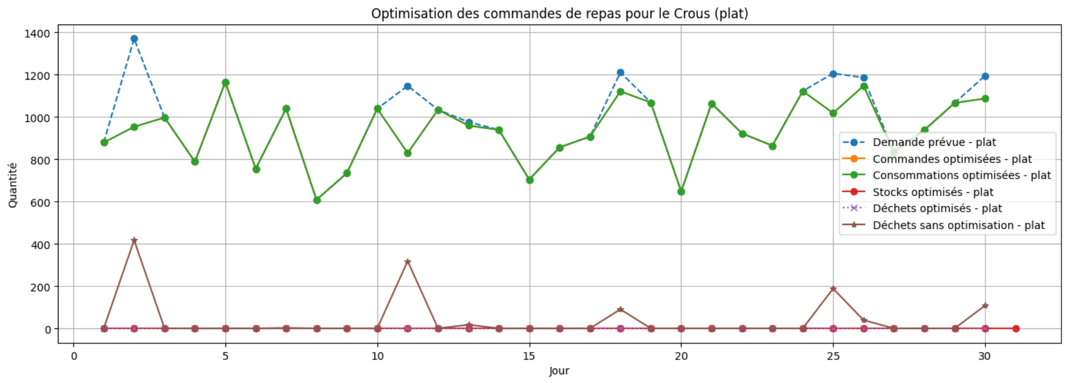
The third graph shows an optimization with an even shorter perishability period for starters (3 days) and desserts (7 days). This configuration shows that the waste for starters and desserts increases compared to the previous configurations, indicating greater difficulty in managing stocks with shorter perishability periods.

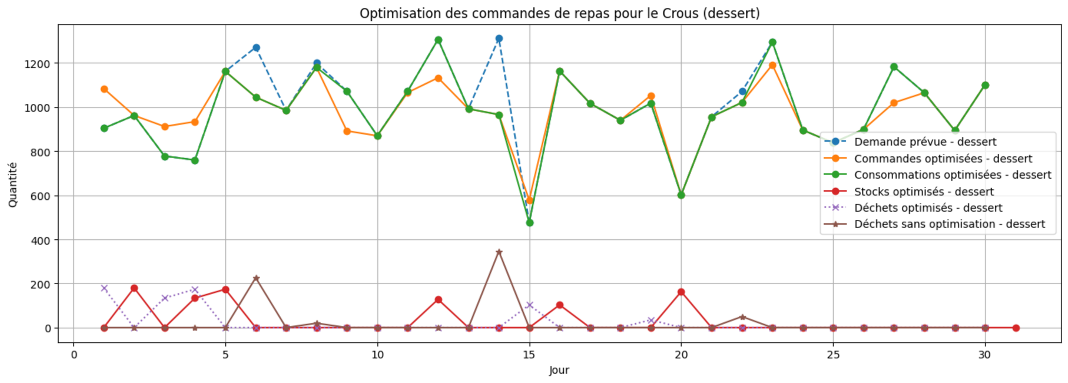
Comparison of average wastes (before vs after optimization):

- **Starter:**
  - Before optimization: 19.07
  - After optimization: 18.07
- **Main dish:**
  - Before optimization: 39.30
  - After optimization: 0.00
- **Dessert:**
  - Before optimization: 21.40
  - After optimization: 20.87

In this configuration, the waste for starters and desserts after optimization is slightly reduced compared to the non-optimized situation, but the reduction is not as significant as in configurations with longer perishability periods. The main dishes continue to generate no waste after optimization. The optimization model helps reduce 5% of starter wastes, 100% of main dish waste (impossible in practice because we did not take into account the wastes due to student who did not finish their plates) and 2.4% of dessert wastes.







These results show that managing perishability periods has a significant impact on waste levels. Longer perishability periods allow for better stock optimization and a more significant reduction in waste.

Moreover the model is meant to be fine-tuned in order to better optimize the orders as a function of the demand and the forecast window.

Summary Table

Graph	Category	Before Optimization	After Optimization	Reduction (%)
1	Starter (14j)	33.23	3.90	88.3
	Main dish (1j)	39.30	0.00	100.0
	Dessert (14j)	38.37	3.90	89.8
2	Starter (7j)	21.63	15.50	28.3
	Main dish (1j)	39.30	0.00	100.0
	Dessert (14j)	38.37	3.90	89.8
3	Starter (3j)	19.07	18.07	5.0
	Main dish (1j)	39.30	0.00	100.0
	Dessert (7j)	21.40	20.87	2.4

Table 5.1: Comparison of average wastes before and after optimization for each graph

However, there are some additional constraints and enhancements that could be integrated to make the model even more robust and realistic.

5.2.3 Production or Service Capacity Constraints

Kitchen capacity: We could add a constraint on the capacity to prepare dishes each day, which limits the maximum quantity of each type of dish that can be prepared based on available resources (staff, kitchen equipment, etc.).

Preparation time: Some dishes require more preparation time than others. A constraint on the total available preparation time per day could be added.

5.2.4 Diversity and Nutritional Balance

Balance of dish types: We could impose constraints to ensure a diversity of dishes served, making sure each category of dish (starter, main course, dessert) is available each day in sufficient quantities to meet nutritional needs and preferences of students.

5.2.5 Menu Flexibility

Dish substitution: We could also allow some flexibility in dish substitutions, for example, allowing one type of dessert to be replaced by another if it can reduce costs or better meet demand without increasing waste.

5.2.6 Optimization Based on Historical Data

Trend analysis: The use of historical data allows us to detect consumption trends and adjust demand forecasts accordingly. For example, if a dish is less popular or generates more waste at certain times of the year, we could adjust orders for those periods.

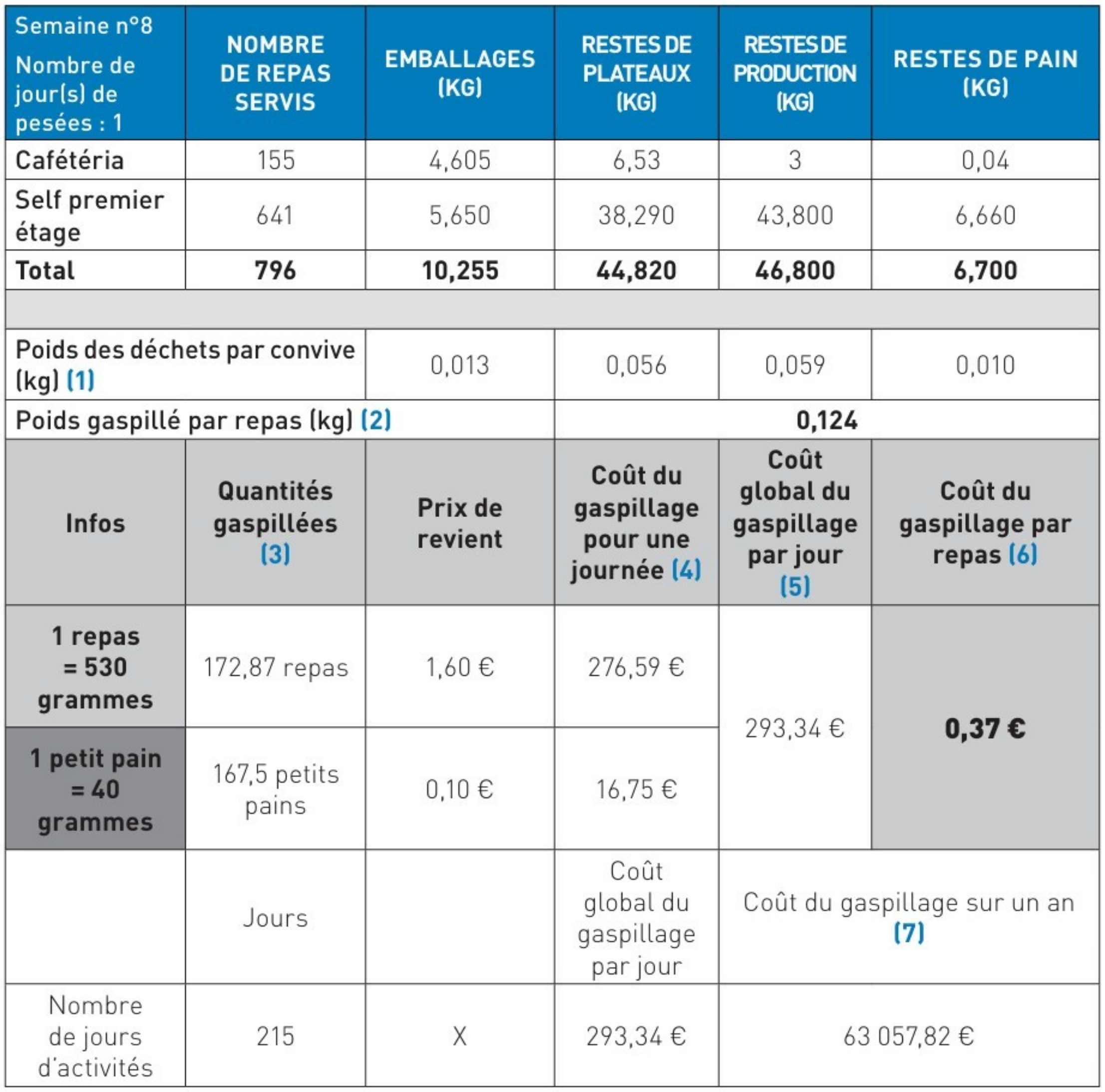
5.2.7 Environmental Impact

Reducing carbon footprint: Eventually we could incorporate a measure of the carbon footprint of different types of dishes and also optimize to minimize environmental impact, favouring local or less polluting products.

Chapter 6

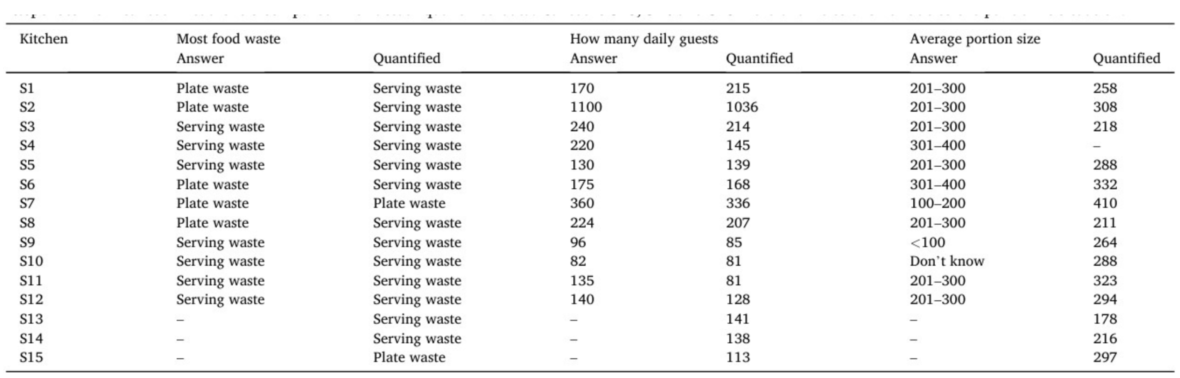
Appendices

6.1 Appendix 1: Reporting of the food waste in Dijon's CROUS [8]



Semaine n°8 Nombre de jour(s) de pesées : 1	NOMBRE DE REPAS SERVIS	EMBALLAGES (KG)	RESTES DE PLATEAUX (KG)	RESTES DE PRODUCTION (KG)	RESTES DE PAIN (KG)
Cafétéria	155	4,605	6,53	3	0,04
Self premier étage	641	5,650	38,290	43,800	6,660
Total	**796**	**10,255**	**44,820**	**46,800**	**6,700**
Poids des déchets par convive (kg) (1)		0,013	0,056	0,059	0,010
Poids gaspillé par repas (kg) (2)			**0,124**		
Infos	**Quantités gaspillées (3)**	**Prix de revient**	**Coût du gaspillage pour une journée (4)**	**Coût global du gaspillage par jour (5)**	**Coût du gaspillage par repas (6)**
1 repas = 530 grammes	172,87 repas	1,60 €	276,59 €	293,34 €	**0,37 €**
1 petit pain = 40 grammes	167,5 petits pains	0,10 €	16,75 €		
	Jours		Coût global du gaspillage par jour	Coût du gaspillage sur un an (7)	
Nombre de jours d'activités	215	X	293,34 €	63 057,82 €	

6.2 Appendix 2: Understanding of daily guests and plate proportion by Swedish Kitchen Chefs [5]



Kitchen	Most food waste		How many daily guests		Average portion size	
	Answer	Quantified	Answer	Quantified	Answer	Quantified
S1	Plate waste	Serving waste	170	215	201–300	258
S2	Plate waste	Serving waste	1100	1036	201–300	308
S3	Serving waste	Serving waste	240	214	201–300	218
S4	Serving waste	Serving waste	220	145	301–400	–
S5	Serving waste	Serving waste	130	139	201–300	288
S6	Plate waste	Serving waste	175	168	301–400	332
S7	Plate waste	Plate waste	360	336	100–200	410
S8	Plate waste	Serving waste	224	207	201–300	211
S9	Serving waste	Serving waste	96	85	<100	264
S10	Serving waste	Serving waste	82	81	Don't know	288
S11	Serving waste	Serving waste	135	81	201–300	323
S12	Serving waste	Serving waste	140	128	201–300	294
S13	–	Serving waste	–	141	–	178
S14	–	Serving waste	–	138	–	216
S15	–	Plate waste	–	113	–	297

6.3 Appendix 3: Algorithm for the Forecast Model of electricity

```python

import os
import numpy as np
import plotly.graph_objects as go
import pandas as pd
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta
import sys
from prophet import Prophet  # Assurez-vous que Prophet est importé correctement
from datetime import datetime, timezone

# Load and prepare the data
FILE_PATH = "electricity_data.csv"  # Assurez-vous que ce fichier contient les données adéquates
TARGET = 'Consumption'
df = pd.read_csv(FILE_PATH)
df['DateTime'] = pd.to_datetime(df['DateTime']).dt.date  # Si les timestamps ne sont pas utiles
df = df.rename(columns={'DateTime': 'ds'})

# Trier le DataFrame par la colonne 'ds' en ordre croissant et grouper les données
df = df.sort_values('ds')
df = df.groupby('ds')[TARGET].sum().reset_index()

# Application d'un moyennage mobile, adapté si nécessaire
window_size = 7  # Fenêtre plus courte pour des données journalières
df['smoothed_clients'] = df[TARGET].rolling(window=window_size, center=True).mean()

# Convertir la colonne 'ds' en datetime
df['ds'] = pd.to_datetime(df['ds'])

# Diviser les données en ensembles d'entraînement et de test
split_point = int(len(df) * 0.8)
```

```python
train = df[:split_point]
test = df[split_point:len(df)-8]

# Configuration du modèle Prophet
model = Prophet(
    daily_seasonality=True,
    weekly_seasonality=True,
    yearly_seasonality=True
)
model.fit(train[['ds', 'smoothed_clients']].rename(columns={'smoothed_clients': 'y'}))

# Préparation des dates futures pour les prédictions
future = model.make_future_dataframe(periods=len(test), freq='D')
forecast = model.predict(future)

# Définir l'offset
offset = 30000  # Ajustez cette valeur selon vos besoins, positive pour monter, négative pour
↪  descendre

# Appliquer l'offset
forecast['yhat_adjusted'] = forecast['yhat'] + offset
selected_yhat_adjusted = forecast['yhat_adjusted'][split_point:len(df)-8]
selected_dates = forecast['ds'][split_point:len(df)-8]

forecast['yhat_lower_adjusted'] = forecast['yhat_lower'] + offset
selected_lower_adjusted = forecast['yhat_lower_adjusted'][split_point:len(df)-8]

forecast['yhat_upper_adjusted'] = forecast['yhat_upper'] + offset
selected_upper_adjusted = forecast['yhat_upper_adjusted'][split_point:len(df)-8]

# Tracer les données
plt.figure(figsize=(14, 7))
plt.plot(train['ds'], train['smoothed_clients'], label='Training Data', color='blue')
plt.plot(test['ds'], test['smoothed_clients'], label='Test Data', color='orange')
plt.plot(forecast['ds'], forecast['yhat'], label='Original Forecast', color='green', alpha=0.5)
plt.plot(selected_dates, selected_yhat_adjusted, label='Adjusted Forecast', color='red')
plt.fill_between(forecast['ds'], forecast['yhat_lower'], forecast['yhat_upper'], color='gray',
↪  alpha=0.2)
plt.fill_between(forecast['ds'][split_point:len(df)-8], selected_lower_adjusted,
↪  selected_upper_adjusted, color='gray', alpha=0.2)
plt.title('Daily Client Numbers at Crous with Forecast Adjustment')
plt.xlabel('Date')
plt.ylabel('Number of Clients')
plt.legend()
plt.show()

# summarize the forecast
print(forecast[['ds', 'yhat', 'yhat_adjusted', 'yhat_lower', 'yhat_upper']].head())

# Ensure the forecast DataFrame is trimmed to the test set dates for accurate MSE calculation
forecast_filtered = forecast[forecast['ds'].isin(test['ds'])]

# Calculate MSE for the forecast
mse_yhat = mean_squared_error(test['smoothed_clients'], forecast_filtered['yhat_adjusted'])/1e5
print("Mean Squared Error for the prediction:", mse_yhat)

```

```python
# Calculate MSE for the lower bound of the prediction
mse_yhat_lower = mean_squared_error(test['smoothed_clients'], forecast_filtered['yhat_lower'])/1e5
print("Mean Squared Error for the lower bound of the prediction:", mse_yhat_lower)

# Calculate MSE for the upper bound of the prediction
mse_yhat_upper = mean_squared_error(test['smoothed_clients'], forecast_filtered['yhat_upper'])/1e5
print("Mean Squared Error for the upper bound of the prediction:", mse_yhat_upper)

```

6.4 Appendix 4: Algorithm for the Forecast Model of German restaurant

```python

import pandas as pd
from prophet import Prophet
import matplotlib.pyplot as plt
import holidays  # Assurez-vous d'avoir importé holidays

# Chargement et préparation des données
FILE_PATH = "german_data.csv"
TARGET = 'count'
df = pd.read_csv(FILE_PATH)
df = df.reset_index(drop=True)

# Renommer les colonnes pour Prophet et convertir les dates
df = df.rename(columns={'date': 'ds'})
df['ds'] = pd.to_datetime(df['ds'], utc=True)
df['ds'] = df['ds'].dt.tz_convert(None)

# Trier le DataFrame par la colonne 'ds' en ordre croissant et grouper les données
df = df.sort_values('ds')
df = df.groupby('ds')[TARGET].sum().reset_index()

# Application d'un moyennage mobile
window_size = 7
df['smoothed_orders'] = df[TARGET].rolling(window=window_size, center=True).mean()
df = df.rename(columns={'smoothed_orders': 'y'})

# Configuration des jours fériés avec le module holidays
fr_holidays = holidays.France(years=[2022, 2023, 2024])  # Inclure 2024 pour les prédictions
holidays_df = pd.DataFrame([
    {'ds': date, 'holiday': name} for date, name in fr_holidays.items()
])

# Configuration du modèle Prophet
model = Prophet(
    daily_seasonality=True,
    weekly_seasonality=True,
    yearly_seasonality=True,
    seasonality_mode='additive',
    holidays=holidays_df  # Inclure les jours fériés ici
)
```

```python

# Utilisation de toutes les données disponibles pour l'entraînement
train = df[df['ds'] <= pd.Timestamp('2023-11-30')]
# Entraînement du modèle
model.fit(train)

# Création des graphiques
plt.figure(figsize=(14, 10))

# Préparation des dates futures et prédiction jusqu'à mai 2024
future_mai = model.make_future_dataframe(periods=(pd.Timestamp('2023-11-30') -
↪  pd.Timestamp('2022-11-30')).days, freq='D')
forecast_mai = model.predict(future_mai)

# Premier subplot
ax1 = plt.subplot(3, 1, 1)
model.plot(forecast_mai, ax=ax1)
ax1.plot(train['ds'], train['y'], 'r-', label='Valeurs réelles')
ax1.set_xlabel('Date')
ax1.set_ylabel('Nombre de clients du Crous (normalisé)')
ax1.set_title('Prédiction jusqu\'à décembre 2023')
ax1.legend()

# Préparation des dates futures et prédiction jusqu'à novembre 2024
future_nov = model.make_future_dataframe(periods=(pd.Timestamp('2024-11-30') -
↪  pd.Timestamp('2022-11-30')).days, freq='D')
forecast_nov = model.predict(future_nov)

# Second subplot
ax2 = plt.subplot(3, 1, 2)
model.plot(forecast_nov, ax=ax2)
ax2.plot(train['ds'], train['y'], 'r-', label='Valeurs réelles')
ax2.set_xlabel('Date')
ax2.set_ylabel('Nombre de clients du Crous (normalisé)')
ax2.set_title('Prédiction jusqu\'à décembre 2024')
ax2.legend()

# Préparation des dates futures et prédiction jusqu'à novembre 2024
future_janv = model.make_future_dataframe(periods=(pd.Timestamp('2025-11-30') -
↪  pd.Timestamp('2022-11-30')).days, freq='D')
forecast_janv = model.predict(future_janv)

# Second subplot
ax3 = plt.subplot(3, 1, 3)
model.plot(forecast_janv, ax=ax3)
ax3.plot(train['ds'], train['y'], 'r-', label='Valeurs réelles')
ax3.set_xlabel('Date')
ax3.set_ylabel('Nombre de clients du Crous (normalisé)')
ax3.set_title('Prédiction jusqu\'à décembre 2025')
ax3.legend()

plt.tight_layout()
plt.show()

```

6.5 Appendix 5: Algorithm for the Optimization Model

```

import gurobipy as gp
from gurobipy import GRB
import matplotlib.pyplot as plt
import numpy as np

# Définir les jours du mois
days = list(range(1, 200))

# Générer des données aléatoires pour forecasted_demand et max_production
np.random.seed(42)  # Pour des résultats reproductibles
mean_demand = 1000
std_demand = 200  # Variance de la demande
mean_production = 1200
std_production = 100  # Variance de la capacité maximale de production

categories = ['entrée', 'plat', 'dessert']
forecasted_demand = {category: np.maximum(0, np.random.normal(mean_demand, std_demand,
↪  len(days))).astype(int) for category in categories}
max_production = {category: np.maximum(0, np.random.normal(mean_production, std_production,
↪  len(days))).astype(int) for category in categories}
waste_levels = {category: np.maximum(0, np.random.normal(100, 50, len(days))).astype(int) for
↪  category in categories}  # Déchets potentiels


# Modèle Gurobi
model = gp.Model('Crous_Order_Optimization')

# Variables de décision : quantités à commander, stock, surplus, pénuries et désordre
orders = model.addVars(categories, days, vtype=GRB.CONTINUOUS, name='orders')
consumptions = model.addVars(categories, days, vtype=GRB.CONTINUOUS, name='consumptions')
stocks = model.addVars(categories, range(len(days) + 1), vtype=GRB.CONTINUOUS, name='stocks')
wastes = model.addVars(categories, days, vtype=GRB.CONTINUOUS, name='wastes')  # Déchets

# Paramètres
initial_stock = 0
sale_price = 3.30
prod_cost = 4
stock_cost = {'entrée': 0.5, 'plat': 1.1, 'dessert': 0.2}  # Différents coûts de stockage
waste_cost = 10  # Coût associé aux déchets

# Contraintes : stock initial
for category in categories:
    model.addConstr(stocks[category, 0] == initial_stock, name=f'initial_stock_{category}')

# Contraintes : évolution des stocks et périssabilité des plats
for category in categories:
    for day in days:
        if category == 'plat':
            # Les plats se conservent uniquement 1 jour
            model.addConstr(stocks[category, day] == orders[category, day] - consumptions[category,
            ↪  day], name=f'stock_evolution_{category}_{day}')
        elif category == 'entrée':
            # Les entrées se conservent au maximum 3 jours
```

```python
            model.addConstr(stocks[category, day] == stocks[category, max(0, day - 14)] + orders[category, day] - consumptions[category, day], name=f'stock_evolution_{category}_{day}')
        else:
            # Les desserts se conservent au maximum 7 jours
            model.addConstr(stocks[category, day] == stocks[category, max(0, day - 14)] + orders[category, day] - consumptions[category, day], name=f'stock_evolution_{category}_{day}')

# Contraintes : les repas consommés ne peuvent dépasser la demande prévisionnelle
for category in categories:
    for day in days:
        model.addConstr(consumptions[category, day] <= forecasted_demand[category][day - 1], name=f'consumption_demand_{category}_{day}')

# Contraintes : les commandes ne peuvent dépasser la capacité maximale
for category in categories:
    for day in days:
        model.addConstr(orders[category, day] <= max_production[category][day - 1], name=f'order_capacity_{category}_{day}')

# # Calcul des déchets : différences entre commandes et consommations
# for category in categories:
#     for day in days:
#         model.addConstr(wastes[category, day] == orders[category, day] - consumptions[category, day], name=f'compute_waste_{category}_{day}')

# Contraintes : réduire les commandes si les déchets sont élevés
for category in categories:
    for day in days:
        model.addConstr(orders[category, day] <= max_production[category][day - 1] - waste_levels[category][day - 1], name=f'waste_constraint_{category}_{day}')
        model.addConstr(wastes[category, day] == waste_levels[category][day - 1], name=f'compute_waste_{category}_{day}')

# Fonction objectif : maximiser les bénéfices en prenant en compte les déchets
objective = gp.quicksum(consumptions[category, day] * sale_price - orders[category, day] * prod_cost - stocks[category, day] * stock_cost[category] - wastes[category, day] * waste_cost
                        for category in categories for day in days)
model.setObjective(objective, GRB.MAXIMIZE)

# Optimisation
model.optimize()

# Extraction des résultats
optimized_orders = {category: [orders[category, day].x for day in days] for category in categories}
optimized_consumptions = {category: [consumptions[category, day].x for day in days] for category in categories}
optimized_stocks = {category: [stocks[category, day].x for day in range(len(days) + 1)] for category in categories}
optimized_wastes = {category: [wastes[category, day].x for day in days] for category in categories}

# Calcul des déchets sans optimisation
def calculate_wastes_no_optimization(forecasted_demand):
    no_opt_wastes = {category: [] for category in categories}
    for category in categories:
        for day in range(len(days)):
            order = forecasted_demand[category][day]
            waste = max(0, forecasted_demand[category][day] - optimized_consumptions[category][day])  # dans le cas non optimisé, on commanderait autant que la prédiction de demande "forecast_demand"
            no_opt_wastes[category].append(waste)
    return no_opt_wastes

no_opt_wastes = calculate_wastes_no_optimization(forecasted_demand)

# Calcul des déchets avec optimisation
def calculate_wastes_with_optimization(forecasted_demand):
    no_opt_wastes = {category: [] for category in categories}
    for category in categories:
        for day in range(len(days)):
            order = forecasted_demand[category][day]
            waste = max(0, optimized_orders[category][day] - optimized_consumptions[category][day]) # dans le cas non optimisé, on commanderait autant que la prédiction de demande "forecast_demand"
            no_opt_wastes[category].append(waste)
    return no_opt_wastes

opt_wastes = calculate_wastes_with_optimization(forecasted_demand)

# Comparaison des déchets avant et après optimisation
def compare_wastes(before, after):
    comparison = {category: {'before': before[category], 'after': after[category]} for category in categories}
    return comparison

waste_comparison = compare_wastes(no_opt_wastes, opt_wastes)

# Visualisation des résultats
fig, axes = plt.subplots(3, 1, figsize=(14, 15))

for idx, category in enumerate(categories):
    ax = axes[idx]
    ax.plot(days, forecasted_demand[category], label=f'Demande prévue - {category}', linestyle='--', marker='o')
    ax.plot(days, optimized_orders[category], label=f'Commandes optimisées - {category}', marker='o')
    ax.plot(days, optimized_consumptions[category], label=f'Consommations optimisées - {category}', marker='o')
    ax.plot(days + [31], optimized_stocks[category], label=f'Stocks optimisés - {category}', marker='o')
    ax.plot(days, opt_wastes[category], label=f'Déchets optimisés - {category}', linestyle=':', marker='x')
    ax.plot(days, no_opt_wastes[category], label=f'Déchets sans optimisation - {category}', linestyle='-', marker='*')

    ax.set_xlabel('Jour')
    ax.set_ylabel('Quantité')
    ax.set_title(f'Optimisation des commandes de repas pour le Crous ({category})')
    ax.legend()
    ax.grid(True)

plt.tight_layout()
plt.show()

print("Demande prévue", forecasted_demand)
print("Commandes optimisées", optimized_orders)
print("Consommations optimisées", optimized_consumptions)
print("Stocks optimisés", optimized_stocks)
print("Déchets optimisés", opt_wastes)
print("Déchets sans optimisation", no_opt_wastes)

# Fonction pour comparer les moyennes des déchets
def compare_mean_wastes(no_opt_wastes, optimized_wastes):
    mean_no_opt_wastes = {category: np.mean(no_opt_wastes[category]) for category in categories}
    mean_optimized_wastes = {category: np.mean(opt_wastes[category]) for category in categories}

    print("\nComparaison des moyennes des déchets (avant vs après optimisation) :")
    for category in categories:
        print(f"{category.capitalize()}:")
        print(f" - Avant optimisation : {mean_no_opt_wastes[category]:.2f}")
        print(f" - Après optimisation : {mean_optimized_wastes[category]:.2f}")

compare_mean_wastes(no_opt_wastes, optimized_wastes)

```

Bibliography

[1] Maria Carmela Annosi, Federica Brunetta, and Marianthi Kostoula Francesco Bimbo. "Digitalization within food supply chains to prevent food waste. Drivers, barriers and collaboration practices". In: *Industrial Marketing Management* (2021).

[2] Article Intranet Data Center. In: *Thales Intranet Article* (2023). URL: `https://intranet.peopleonline.corp.thales/news/article/index.cfm?nid=921390`.

[3] Belén Derqui and Vicenc Fernandez. "The opportunity of tracking food waste in school canteens: Guidelines for self-assessment". In: *Waste Management* (2019).

[4] Alberto Garrea, Mari Carmen Ruizb, and Eloy Hontoriac. "Application of Machine Learning to support production planning of a food industry in the context of waste generation under uncertainty". In: *Operations Research Perspectives* (2020).

[5] Christopher MALEFORS et al. "Testing interventions to reduce food waste in school catering". In: *Resources, Conservation Recycling* (2022).

[6] Christopher Malefors et al. "Food waste reduction and economic savings in times of crisis: The potential of machine learning methods to plan guest attendance in Swedish public catering during the Covid-19 pandemic". In: *Socio-Economic Planning Sciences* (2022).

[7] Christopher Malefors et al. "Potential for using guest attendance forecasting in Swedish public catering to reduce overcatering". In: *Sustainable Production and Consumption* (2021).

[8] Restaurant Universitaire Montmuzard Restaurant Universitaire Mansart. "Guide : Mon CROUS Responsable". In: (2018).

[9] Silvennoinen, Nisonen, and Pietiläinen. *Food waste case study and monitoring developing in Finnish food services*. Waste management, 2019.