Universiteit Utrecht

Master thesis
"Automating the Ingredients Ordering Process in Restaurants: A Machine Learning-Powered System Approach"

Hsieh, Sunny.
Artificial Intelligence
Faculty of Sciences, Utrecht University

First supervisor: Hans Bodlaender
Second supervisor: Alison Liu

January 30, 2023

Abstract

In this research, we aimed to automate the ingredients ordering process in restaurants. By labeling and analyzing a subset of dishes from an Indian restaurant's menu based in London, we were able to accurately predict the quantities of ingredients needed in a given time period. This has the potential to greatly improve the efficiency of the ingredients ordering process, reducing the need for manual tracking and ordering and potentially reducing waste. We designed a pipeline consisting of three machine learning components, a detector, a determinator, a promoter, and a reducer. The machine learning components predict the total ingredients needed for different time frames, and the detector and determinator prevent overstock and understock, respectively. The promoter suggests possible actions in the event of overstock, and the reducer produces a preliminary order. Linear programming techniques could be used to finalize the order if constraints are present. The machine learning components were evaluated using the R squared score, and all three had a very high accuracy. The overall performance of the pipeline was evaluated using three custom metrics, which showed excellent results. This demonstrates that it is possible to automate the ingredients ordering process, even for small, non-franchised restaurants. While our study focused on a specific restaurant, the algorithm may also be applicable to other food service organizations, such as supermarkets and fast food chains. However, further research is needed to fully evaluate its performance in a wider range of settings.

Acknowledgements

I would like to express my sincere gratitude to all the experts who were interviewed for this study and generously shared their insights and experiences with me. Their contributions were invaluable and greatly enhanced the quality of this research.

I would also like to thank my first supervisor, Hans Bodlaender, for his guidance, support, and encouragement throughout the research process. His expertise and invaluable feedback were essential in the development of this thesis.

Finally, I would like to express my gratitude to all the individuals who have contributed to this research in any way. Your assistance and support have been greatly appreciated.

Contents

1 Introduction **4**
- 1.1 Interviews . . . 5
- 1.2 Restaurant forecast . . . 6
  - 1.2.1 Which data should be used in the forecasting models? . . . 6
  - 1.2.2 Which model should be used for the forecasting models? . . . 7
  - 1.2.3 Constraints . . . 7
  - 1.2.4 Overstock . . . 8
- 1.3 Research goals . . . 8

2 Data **9**
- 2.1 Dataset and data analysis . . . 9
  - 2.1.1 Dishes and Ingredients . . . 9
  - 2.1.2 Useful data points . . . 9

3 Methodological design **11**
- 3.1 Pipeline . . . 11
  - 3.1.1 Storage system . . . 11
  - 3.1.2 ML1 . . . 13
  - 3.1.3 Detector . . . 13
  - 3.1.4 Promotor . . . 14
  - 3.1.5 ML2 . . . 14
  - 3.1.6 Determinator . . . 14
  - 3.1.7 ML3 . . . 14
  - 3.1.8 Reducer . . . 14
  - 3.1.9 LP . . . 15

4 Results **16**
- 4.1 Results ML-1, ML-2, and ML-3 . . . 16
- 4.2 Evaluation metric 1 . . . 16
- 4.3 Evaluation metric 2 . . . 17
- 4.4 Evaluation metric 3 . . . 17

5 Conclusion **19**

6 Discussion **21**
- 6.1 Data features . . . 21
- 6.2 Labelled partial dishes . . . 21
- 6.3 Ingredients were not in package size . . . 21
- 6.4 Absence of retraining . . . 21

1 Introduction

In the restaurant industry, revenue is essential for success. To maximize profits, it is important to keep operational costs low. One key factor that can impact operational costs is storage management. Effective storage management can have a significant impact on the success of a restaurant, while poor storage management can lead to waste and inefficiency.

One important aspect of effective storage management in the restaurant industry is monitoring the expiration dates of ingredients. It is crucial for restaurants to accurately track and label the expiration dates of ingredients to ensure that they are used in a timely manner and to prevent spoilage. This not only helps to prevent waste and reduce operational costs, but it is also essential for food safety and compliance with regulations. By properly tracking and labeling expiration dates, restaurants can improve their storage management processes and maintain the quality and safety of their food.

Furthermore, one of the challenges that the restaurant industry currently faces in regards to storage management is the difficulty of predicting the amount of ingredients needed. The restaurant industry traditionally relies on manual methods for tracking ingredient inventory, such as conducting periodic physical audits of storage rooms and ordering new ingredients based on estimated needs.

Even nowadays, most restaurants are still using men power to keep track of their storage and order manually when they notice a low storage in some ingredients. The restaurant management is still mainly relying on personal experience; there is no data system support [12]. Furthermore we see that in franchised stores such as Starbucks this is the same case as well [15].

However, this approach is prone to data inaccuracies, as it relies on manual data entry and estimation rather than more precise tracking methods. This can lead to overstocking, understocking, and waste, all of which can impact operational costs, profitability and customer satisfaction. Another challenge is maintaining an accurate inventory of the storage room, which is essential for effective planning and decision making.

To overcome these challenges, it is important for restaurants to implement more effective methods for tracking ingredient inventory. This may include utilizing inventory management software, implementing forecasting techniques, and regularly monitoring storage levels to ensure that inventory is accurately tracked and adjusted as needed. By implementing these strategies, restaurants can improve their storage management processes and reduce the potential for data inaccuracies.

The focus of this thesis is on the development of an optimization algorithm that can automate the process of ingredient storage management in the restaurant industry. This algorithm will be designed to perform the following tasks:

1. Accurately track and monitor ingredient inventory levels. It needs to keep track of the expiration date of ingredients, such that the user is aware of the current quantities in the inventory. It enables users to see if they are understocked, which means the restaurant should place an order, or if

they are overstocked.

2. Minimize orders, and at the same time minimize product waste. This also comes down to minimizing operational costs as each order costs extra money and each waste means unnecessary expenditures. The algorithm needs to be able to determine when to place an order and what to order. The output of the algorithm would be an order list of ingredients, and the desired quantities of these ingredients in kilograms while keeping possible constraints in mind.

3. Detect situations of overstock and let the user know which ingredient is in overstock when a restaurant is facing ingredients that almost reach their expiration date. If necessary, the system could also be able to suggest what to do with the ingredient.

Regarding the last case, we give a more specific example. If the algorithm detects a very large quantity of chicken in the storage, and also a large quantity of corn, but a low quantity of beans, and all three ingredients are facing their expiration date soon. If the algorithm also knows that the restaurant only sells a chicken dish with corn and a chicken dish with beans, then the algorithm needs to suggest to give a promotion on the chicken dish with corn. Also, if it is possible, the algorithm should determine what this promotion should be. (This thesis will first discuss the algorithm that uses a fixed amount of promotion).

Of course, this was a very simplistic example and in reality we are facing tens or even hundreds of different ingredients and combinations. Additionally, if dishes are made of ingredients A, B and C, but there is only a high quantity of ingredient B, then the algorithm could suggest to use more of ingredient B in each dish that contains ingredient B in it.

By automating this process, the goal of this research is to improve the efficiency and accuracy of storage management in the restaurant industry, ultimately reducing operational costs and increasing profitability.

1.1 Interviews

One of the key sources of data for this study is interviews with experts in the field. This approach was chosen because it allows for a deep understanding of the topic at hand and provides insights that may not be captured through other research methods.

The interviews were conducted with a semi-structured format, allowing for flexibility and the exploration of new ideas that emerged during the conversation. A total of 9 experts were interviewed, selected based on their expertise and experience in the relevant subject area, namely:

1. A. Schulze: 2 years experience in Starbucks as part timer and manager assistant.

2. M. Kloos: Assistant manager for 2 years at Albert Heijn.

3. D. Karakus: Assistant manager for 4 years at Mc Donald's.
4. A. Chao: 10 years experience in restaurants in different countries.
5. J. Wu: 10 years experience in his own Chinese Surinamese restaurants.
6. J. Hau: daughter of the owner of a franchised Japanese restaurant. Worked and helped in the restaurant for around fifteen years.
7. S. Hau: daughter of the owner of a franchised Japanese restaurant. Worked and helped in the restaurant for around a decade.
8. Junda: son of the owner of a Chinese restaurant. Worked and helped in the restaurant for a decade.
9. J. Jiang: daughter of the owner of a Chinese restaurant. Worked and helped in the restaurant for a decade.

The interviews were not recorded, but the key concepts discussed were written down and analyzed using thematic analysis. This method allowed for the identification of key themes and patterns within the data, providing valuable insights into the research questions. A final outcome analysis was conducted by comparing and contrasting the results of the interviews.

Overall, the use of interviews as a source of literature provided a rich and nuanced understanding of the topic and proved to be a valuable addition to the overall research.

1.2 Restaurant forecast

To automate the ordering process of restaurants, it is essential to have a model that accurately forecasts the number of ingredients that will be used by a restaurant. These days, the majority of the restaurant industry uses judgemental methods to forecast variables such as sales. However, numerous studies have concluded that quantitative methods provide a better forecast than judgemental methods [1, 13].

1.2.1 Which data should be used in the forecasting models?

In general, there are two ways to forecast the number of ingredients a restaurant will use.

The first method forecasts the number of dishes a restaurant will sell in the to be specified integer x days. Based on this approximation, it is possible to estimate the number of needed ingredients.

The second approach is to transform the data into the number of used ingredients in a day and then forecast the number of needed ingredients in the next x days.

According to an interview with Chao [2], it would be more suitable to train a model based on the number of dishes sold and transfer it to the number of

ingredients needed, i.e., to use the first method. Chao reasons that it would be easier to anticipate when the content on the menu or portion of dishes is changed. If, for instance, the volume of a dish is decreased, or if the restaurant owner decides to delete some dishes from the menu, it would not require the model to be retrained. But, if a model is trained according to approach 2, a much more serious problem would occur.

1.2.2 Which model should be used for the forecasting models?

To determine which model is most appropriate for restaurant forecasting, we will first investigate whether self-selectivity is present. Self-selectivity occurs when an independent restaurateur opens and closes during different times of the week or year based on the expectation of the success of sales. As a result, a self-selectivity bias may have formed. In this case, a truncated regression model would fit the data best overall [14].

When self-selectivity is not present, Cranage et al. [3] illustrated that time series models and features are accurate and efficient for using forecasting techniques in the restaurant industry.

However, Wang [16] showed that a prediction model based on back propagation artificial neural network is one of the most effective methods for forecasting problems. Using weather conditions as a feature in a BP neural network resulted in a good performance. In addition to this, Xinliang et al. [18] showed that the BP neural network model performed better than time series models in similar forecasting problems. When we consider the option of using gradient boosting trees as well, Giannakas et al. [4] shows that their XGBoost model outperforms neural networks. In this paper only four hidden layers were used for the neural network. The paper states that with no or very little finetuning of the hyperparameters, XGBoost can outperform neural networks easily.

We need to keep in mind that the goal of this paper is not to get the highest accuracy in predicting the number of dishes sold. This is certainly important as well, but this paper will mainly focus on whether it is possible to automatize the ordering process of restaurants. Taking this literature into consideration, we used XGBoost model as forecasting model. The data will be transformed into timeseries data for training the model.

1.2.3 Constraints

To simulate the real-world scenario as closely as possible, we need to consider a few constraints that regard the ordering process and storage capacity. One such constraints is the delivery time of the ingredients, and another constraint is the amount of storage for ingredients.

When an order is placed, we will use a fixed time frame of d days to indicate the delivery time of the order. So it takes d days for the ingredients in that order to be delivered to the restaurant. Each delivery has c euros delivery costs.

The sisters Hao mentioned that, on average, the delivery time of ingredients is between 1 or 2 days, but mostly one [5, 6]. Therefore we have chosen to use

$d = 1$.

Furthermore, both the sisters Hao and Wu mentioned that the delivery costs in most restaurants is 0 [5, 6, 17]. This free delivery cost is due to the enormous amount of ingredients the restaurant needs. Moreover, franchised restaurants often use the same manufacturer. In this kind of situations there is often a contract of collaboration of at least two years. Therefore, in this paper we will consider the cost of an order to be 0, so $c = 0$.

If possible, we would like to include constraints for the amount of storage. For this, we will define a couple decision variables. There will be room for storing $s_1 cm^3$ of frozen ingredients, $s_2 cm^3$ of cold ingredients and $s_3 cm^3$ amount of dry ingredients that do not require a low-temperature storage.

Nevertheless, as claimed by Jiang and Junda [8, 9], it is very seldom that a restaurant order exceeds the storage capacity. Combining this information, we will start with the following settings:

1. We omit the possibility that the amount of order will ever exceed the amount of storage. It is possible to modify these assumptions.
2. We assume a delivery time of 1 day, $d = 1$.
3. We assume no costs for delivery, $c = 0$.

1.2.4 Overstock

According to Chao [2], in the case of overstocked ingredients, a restaurant will often do one of the following:

1. Use overstocked ingredients for staff meals.
2. Create a "dish of the day" using the ingredients in overstock.

We will consider this when designing the pipeline of the algorithm.

1.3 Research goals

The problem we aim to solve is the automatization of the stock ordering process of a restaurant. To achieve this, we need to consider the design of a pipeline for automating the ordering process, as well as the automatic tracking of ingredients. Additionally, we need to address the issue of overstock and devise strategies to minimize waste and determine the appropriate amount of each ingredient to order. Finally, we need to consider how to minimize the number of orders.

2 Data

The relation between dishes and ingredients can be generalized to the following concept. The restaurant uses a ingredients, that are processed in b different dishes. Note that a, the quantity of used ingredients, could be smaller or larger than b, the number of dishes served by the restaurant.

Each ingredient a_i can be bought in quantities of $x_j \times 0.01$ kilograms, where x_j is a positive integer, and each ingredient a_i has an expiration date e_i.

2.1 Dataset and data analysis

For this research, we used a dataset from Kaggle, [1]. a website with open datasets. More specifically, we used the takeaway food order dataset, which contains approximately 20,000 orders from an Indian takeaway restaurant based in London. Each row in the dataset represents a single product within an order.

The average daily order volume of the restaurant is 133, while the average weekly and monthly order volumes are 927 and 3934, respectively. It is unknown whether this restaurant provides indoor dining, but for the purposes of this research, we will only consider the data on takeaway food orders and assume that the restaurant does not provide indoor dining.

2.1.1 Dishes and Ingredients

The restaurant serves up to 302 dishes, and we assume that all dishes are ordered at least once. However, the dataset does not contain any information about the ingredients of a dish. We will thus construct the ingredients per dish ourselves using the knowledge of domain experts (Indian chefs) and Google. In Figure 1 we can see the head of the dataframe.

	Order ID	Order Date	Item Name	Quantity	Product Price	Total products
0	25583	2019-08-03	tandoori mixed grill	1	11.95	12
1	25583	2019-08-03	madras sauce	1	3.95	12
2	25583	2019-08-03	mushroom rice	2	3.95	12
3	25583	2019-08-03	garlic naan	1	2.95	12
4	25583	2019-08-03	paratha	1	2.95	12

Figure 1: Head of the dataframe

2.1.2 Useful data points

Figure 2 shows the number of orders per day plotted against time in days. At the beginning of the timeline, the data appears to be unstable. It is unlikely that the restaurant did not make any sales for an extended period, so we assume that the instability is due to missing data.

[1] https://www.kaggle.com/datasets
A website with open datasets

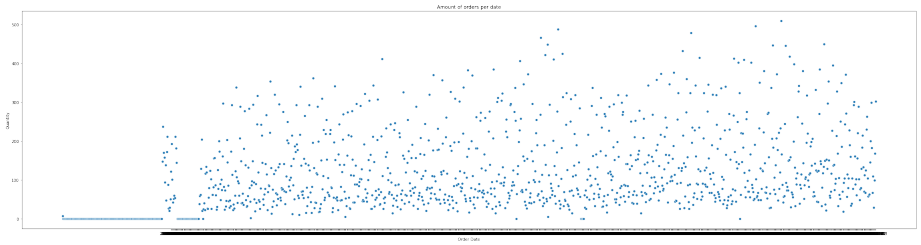

Figure 2: Quantity of orders per date

If we take a closer look at the data, see Figure 3, we see that starting from August 2016, the orders become more consistent. Hence, we will use the data points from this date onward till August 2019 in our model.

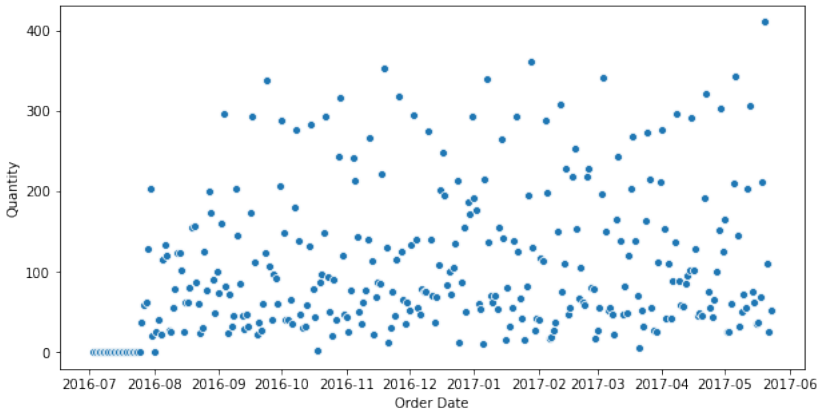

Figure 3: Quantity of orders per date on a zoomed interval

An obstacle was encountered during the construction of the algorithm. If the number of dishes b is greater than fifteen, the computer is not capable of handling the heavy training of the models. In addition, labelling dishes is a time-consuming task. Therefore, the algorithm was limited to fifteen dishes, which resulted in 47 different ingredients. The fifteen dishes were chosen randomly, meanwhile making sure that we have enough training data for these dishes which was indeed the case. These fifteen dishes resulted in a data set of approximately 1100 rows.

In addition to this, we will split the data points we are using into a training set of 80% and a test set of 20%.

3 Methodological design

The problem addressed in this study is the automatization of the ingredients ordering process of restaurants. To the best of our knowledge, there have been no previous of few studies on this topic. Therefore, we designed our own pipeline for automating the ordering process of restaurants, which can be seen in Figure 4. Our goal was to create a pipeline that could be used in any restaurant that wishes to automate their ordering process, but the hyperparameters may need to be adjusted for optimal performance for each individual restaurant.

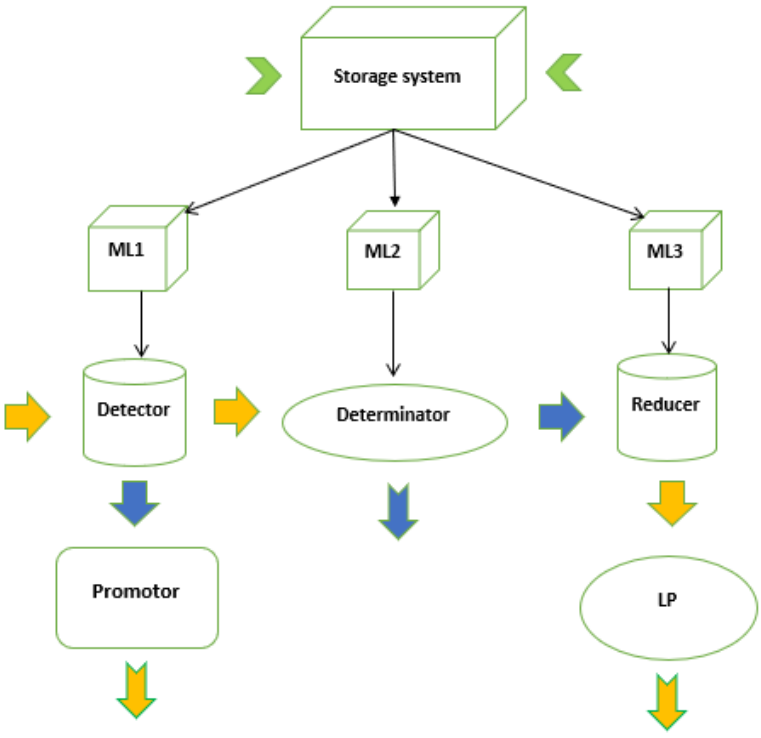

Figure 4: Pipeline

3.1 Pipeline

3.1.1 Storage system

The storage system can be seen as a database that keeps track of the following data flows:

1. How much of each ingredient is left at a certain moment?

So every time the restaurant sells a product, it deducts the number of ingredients that are used. On top of that, every time the restaurant refills their storage, it updates the number of remaining ingredients.

2. When will each ingredient expire?

3. How much of each product has been sold each day?

Note that a product is a combination of one or more ingredients. These information will be fed into the pipeline.

In Table 1, an example is given of a possible data frame where the quantities are displayed in kilograms. The columns under the dates represent the quantity of each ingredient.

Ingredients	**Current Total Amount**
A1	2.0
A2	2.5
B1	3.2
B2	4.0

Table 1: Simplified example of the current ingredients status dataframe of the storage system

Furthermore, a linked list class is created for each ingredient to keep track of the expiration dates. The first node is "the fastest in expiration date node". In figure 5 we see a visualized linked list class of an ingredient. The first number in a node denotes the number of ingredients left, and the second number indicates the number of days left until the expiration date.

Figure 5: Example of visualizing a linked list class of an ingredient

In Table 2, an example is given of a possible, simplified, data frame where the number of dishes sold, is kept track of.

	Product A	**Product B**
2022/09/01	100	120
2022/09/02	96	112
2022/09/03	103	110
2022/09/04	140	152

Table 2: Simplified example of the sold dishes data frame of the storage system

Thus, the system has two data frames, which it keeps updating, and a linked list for each ingredient.

The first-in-first-out principle is applied in restaurants, but not always executed perfectly. However, in this thesis, we will assume that this principle is perfectly executed.

3.1.2 ML1

At the end of each day, the first machine learning model predicts the number of dishes sold on the next n_1 days. n_1 can be different per restaurant and per case, in this case $n_1 = 1$. This number should be small since the ML1 looks into the short term future, whereas ML2 & ML3 looks into the middle to long term future. ML1 will be trained on historical data, using a data frame like 2. This model can be retrained every month using the newly collected data. Ingredient A will of course only be trained on the column A. Furthermore, the data will be transformed into time series features such as the amount of sales and average sales of dish d_i of the previous week up until the previous w weeks. In this case we have chosen w to be 15. The same kind of features will also be used in ML2 and ML3. On top of this, we used Gridsearch to fine-tune the hyperparameters in this model.

3.1.3 Detector

After ML1 has predicted the quantity of each dish that will be sold in the next n_1 days, it transforms this information to the number of ingredients that is needed for the next day. Recall that $n_1 = 1$ in this case. It looks at the system storage and determines whether there are enough ingredients for the next day. It might also look at ingredients that are not in stock anymore, even in the case where the model does not predict any of these ingredients to be needed. However, it is necessary to always have some back up, which is why a threshold will be set for cases like this.

Furthermore, the detector looks at how many ingredients expire the next day and compares it to the predicted amount of dishes that is needed the next day. In the case of overstock, i.e. when the number of ingredients that expires the next day minus the predicted amount of those ingredients that will be used the next day is a positive number, it invokes the Promotor.

In order to prevent understock, it always invokes the determinator.

3.1.4 Promotor

In case of overstock, this part of the pipeline will be invoked. The promotor returns the ingredients that are in overstock, and also the amount with which those ingredients are in overstock.

3.1.5 ML2

The second machine learning model, ML2, will predict how much of each dishes is needed for the coming n_2 days. Like n_1, n_2 could also be different per restaurant and situation. However, n_2 should always be bigger than n_1 since ML2 looks further away than ML1 in the future. Here we have chosen n_2 to be 3.

ML2 will also be trained on historical data, using a data frame like 2, but the difference here is that it takes the sum of every n_2 rows. To clarify this, if it is day 1, ML2 looks n_2 days ahead, in this case 3. The corresponding number to day 1 should thus be the sum of day 2,3 and 4. Again the model can be retrained every month using the newly collected data. Identically to ML1, Gridsearch has been used to fine-tune the hyperparameters, and the same time series features are used in this model.

3.1.6 Determinator

So using ML2, the determinator looks into the number of ingredients that are needed for the next three days and compares it to the current amount of ingredients. If there are not enough ingredients, the determinator will return a "False", otherwise, it will return a "True". The determinator is thus an indicator that tells us whether we should place an order. In case the determinator returns "True", no order will be placed. However, if the determinator returns "False", it will invoke the reducer.

3.1.7 ML3

The third machine learning model, ML3, will predict how much of each dishes is needed for the coming n_3 days. Like n_1 and n_2, n_3 could also be different per restaurant and situation. Here we have chosen n_3 to be 7.

ML3 is trained similarly to ML2, but instead of a rolling window of $n_2, 3$, ML3 uses a rolling window of $n_3, 7$. Identically to ML1, Gridsearch has been used to fine-tune the hyperparameters, and the same kind of features are used in this model.

3.1.8 Reducer

The reducer transforms the predicted amount of different dishes that will be sold in the coming n_2 days into the number of ingredients that is needed to make these dishes. Subsequently, it compares this number with the remaining amount of stock that is left in the system storage. It then determines the number of ingredients that need to be ordered. This first draft order list will be passed

into the last stage LP. It has been deliberately chosen for predicting the dishes and transform it later in this stage to the amounts of needed ingredients. The reason for this is that it would be much easier for a restaurant to adjust the amount of ingredients in a dish if they want, or in the case when a restaurant wants to remove or add a new dish into the menu. If the models were trained on ingredients, the performance would be less robust and stable.

3.1.9 LP

The LP is a model that uses linear programming and might reduce the order list because of the constraints. The output of this stage will be our final order list. In this paper we have loose constraints such that LP will not be used in this thesis. Nonetheless, we did include LP consciously in case it is needed in future work or research. It can be the case that for other restaurants or situations that this part is necessary.

4 Results

In order to automatize the stock ordering process of a restaurant, we established three machine learning models: ML-1, ML-2, and ML-3. The results of these machine learning models are crucial in automatizing the stock ordering process. In this section, the results of these three models will be discussed.

4.1 Results ML-1, ML-2, and ML-3

The results of the three machine learning models are stated in the Table 3.

Table 3: R squared scores of ML1, ML2 and ML3

Dishes	ML-1	ML-2	ML-3
Tandoori mixed grill	0.99736	0.99984	0.99938
Madras sauce	0.99577	0.99978	0.99960
Mushroom rice	0.99987	0.99976	0.99018
Garlic naan	0.99951	0.99869	0.99911
Paratha	0.99896	0.99411	0.93065
Plain rice	0.99989	0.99977	0.99927
Prawn puree	0.99988	0.99953	0.98311
Plain papadum	0.99955	0.99965	0.98945
Mango chutney	0.99974	0.99980	0.99873
Onion chutney	0.99560	0.99933	0.99899
Mint sauce	0.99134	0.99954	0.99679
Chicken tikka masala	0.99632	0.99986	0.99906
Tandoori king prawn masala	0.99989	0.98855	0.99993
Pilau rice	0.99902	0.99963	0.99708
Peshwari naan	0.98234	0.99315	0.99927

Note that all three machine learning models provide R-squared scores as their results. This results comes from the fact that all the models apply the loss function to evaluate the performance of the algorithm. Also, recall that the machine learning models predict the number of dishes and not the number of ingredients used in these dishes.

From the results in the table above, it can be observed that all R-squared scores are relatively high. In particular, all the scores are approximately 1. This means that, on the scale of the R-squared score, the model is more or less the perfect fit. These highly accurate results can be explained by the stable data and the quality of the tuned XGB boost parameters.

4.2 Evaluation metric 1

The second evaluation metric measures the accuracy of order decisions of the algorithm. For this, the evaluation metric observes the algorithm when it places

an order or when the algorithm decides to not place an order, based on predictions. It will then compare this to the real test set and evaluate if it was actually required to place an order. Equation 1 denotes the formula for evaluation metric 2, where $n_{testdays}$ denotes the amount of days in the testset. I is an indicator function that compares whether there was an order placed by the algorithm on day i or not denoted by $Order_i^{Actual}$, and whether the order was necessary denoted by $Order_i^{Actual}$.

$$\frac{\sum_{i=1}^{n_{testdays}} I[Order_i^{algo} = Order_i^{Actual}]}{n_{testdays}} \tag{1}$$

The algorithm made a prediction that the number of days in which an order will be placed, would be 100. This prediction resulted in an ordering percentage of 46.73%.

In the actual test set, the number of days in which an order was placed is 100. This result yields in an actual ordering percentage of 47.20%.

Above, the reciprocal results are presented. However, the evaluation metric should evaluate the algorithms accuracy of ordering per day. In order words, it should compare, per day, whether the algorithm decides to place an order, to when an actual order was needed that day.
When we consider the accuracy of the algorithm per day, we observe an error of 0.00465, which gives an error percentage of 0.465%. This results in a correction of right ordering timing of 99.535%.

4.3 Evaluation metric 2

The first evaluation metric checks the number of miscalculations, e.g. when the quantity of an ingredient ordered, is less than the quantity of that ingredient needed on a day.
For this evaluation metric, we choose to adopt a test set of 214 days, and a total of 47 ingredients that can be used. Each day, we will check whether there is a shortage in ingredients. If this is the case, i.e. more ingredients are needed than there is available, the the number of *miscalculations* will increase by 1. Our evaluation metric 1 is defined in Equation 2, where *miscalculations* is denoted as the number of times of miscalculation, $days_{testset}$ the amount of days is the testset and $n_{ingredients}$ the amount of ingredients we have.

$$\frac{miscalculations}{days_{testset} * n_{ingredients}} \tag{2}$$

Using Equation 2, with $days_{testset} = 214$, $n_{ingredients} = 47$ and the data that we in total observed 41 miscalculations, which yields an miscalculations percentage, i.e. evaluation metrics of 0.41%

4.4 Evaluation metric 3

The third evaluation metric looks at how many times the promotor has been invoked. The result will be a score in the range of 0 and 1. The formula is

defined in Equation 3, where $N_{promotor}$ stands for the amount of times the promotor has been invoked and $days_{testset}$ the amount of days in the testset.

$$1 - \frac{N_{promotor}}{days_{testset}} \tag{3}$$

The higher the score, the better the result. In our testset, the promotor has never been called so we resulted in a evaluation metric score of 1.

5 Conclusion

Based on the results of the three machine learning models, and the two evaluation metrics, we can conclude that our model is able to automatize the stock ordering process of a restaurant relatively accurately.

The three machine learning models yielded results, which were all relatively high. The lowest R-squared score obtained was 0.93. This is, on itself, already a relatively high score. These high R-squared scores signify that the model perfectly suitable.

We also found that the number of miscalculations that our model made, was relatively low. In total, only 41 miscalculations were made, which resulted in an evaluation metric of 0.41%. Naturally, a lower number of miscalculations is more desired. This low evaluation metric is therefore appropriate for a good model.

On top of this, the accuracy of the order decisions of the algorithm was also measured. The algorithm did an excellent job in predicting the number of days on which an order was placed. The algorithm predicted an ordering percentage of 46.73% which did not deviate significantly from the actual test set. In fact, the ordering percentage of the actual set, which was 47.20%, was almost similar to the predicted percentage of the algorithm. The model was also highly accurate in deciding when an order should have been placed. The error percentage was merely 0.47% and resulted in a correction of right ordering timing of 99.53%. At last we see with stable data, overstock has in our case never occurred. Recall that evaluation metric 3 resulted in a score of 1.

A lot of these results arise from the use of stable data and the quality of the tuned XGB boost parameters.

In conclusion, this study has demonstrated the potential for automating the ingredients ordering process in restaurants. By labeling and analyzing the ingredients used in a subset of dishes from a restaurant's menu, we were able to develop an algorithm that accurately predicts the quantities of ingredients needed in a given time period. This has the potential to greatly improve the efficiency of the ingredients ordering process, reducing the need for manual tracking and ordering and potentially reducing waste.

The results of this study have several potential implementations in the restaurant industry. First and foremost, the algorithm we developed could be used to automate the ingredients ordering process, reducing the need for manual tracking and ordering and potentially saving time and resources. This could be particularly useful in high-volume restaurants, where the quantity of ingredients used can be significant and the process of tracking and ordering them can be complex and time-consuming. Additionally, the use of the algorithm could help to reduce waste by ensuring that the appropriate quantities of ingredients are ordered and used, without excess.

In addition to its potential applications in the restaurant industry, the algorithm developed in this study may also be useful in larger food service organizations, such as supermarkets and fast food chains. These organizations may have similar needs in terms of tracking and ordering ingredients, and it is

likely that they also have in-house research on these issues. Our model has the potential to function successfully in these settings due to the inherent stability of the data in large organizations. For example, a supermarket chain like Albert Heijn or a fast food chain like McDonald's may have consistent and reliable data on the ingredients used in their products, which would enable the algorithm to accurately predict the quantities needed [10, 11]. This could help to improve the efficiency of their operations and reduce waste.

6 Discussion

6.1 Data features

In the machine learning models, we could also incorporate additional features, such as the size of the restaurant, the type of restaurant, the presence of outdoor seating, and weather conditions. These features could be beneficial because models are highly dependent on the settings of a restaurant [7]. However, the high accuracy of our model suggests that the inclusion of these additional features may be unnecessary.

6.2 Labelled partial dishes

In this study, we only labeled and analyzed 15 dishes, resulting in a dataset of 47 ingredients. It should be noted that the restaurant we studied offered a total of 302 dishes, and it may be interesting to evaluate the performance of our algorithm when applied to the entire menu. However, due to limitations on time and resources, we were only able to label and analyze a small subset of the available dishes.

6.3 Ingredients were not in package size

When valuating the output of the algorithm, we evaluated the number of ingredients needed in a given time period by using the quantity of ingredients in kilograms. However, it would be more efficient to express this quantity in terms of package sizes, as this would allow the algorithm to order the ingredients in the appropriate number of packages. In future research, it may be worthwhile to explore the use of linear programming techniques, or simple rounding algorithms, to address this issue.

6.4 Absence of retraining

Another potential limitation of our model is that it is not regularly retrained. This could be problematic in situations where the nature of the data changes significantly, such as if the restaurant begins serving only vegetarian dishes, or in the event of a major disruption like the COVID-19 pandemic. In such cases, it may be necessary to collect new data and retrain the algorithm to accurately reflect the system being modeled.

While our model was effective in the case of the specific restaurant we studied, it is worth noting that the results may not generalize to other restaurants. In particular, other restaurants may have different menus and ingredient usage patterns, and it may be necessary to retrain the algorithm to accurately reflect the data in these cases. Additionally, the algorithm may need to be retrained periodically in order to account for changes in ingredient usage over time. As such, it is important to carefully evaluate the performance of the algorithm and consider retraining as necessary when applying it to other restaurants.

References

[1] J. Armstrong. Relative accuracy of judgemental and extrapolation methods in forecasting annual earnings. *Journal of Forecasting*, 2:437–447, 1983.

[2] A. Chao. Private community, 2022.

[3] David A. Cranage and William P. Andrew. A comparison of time series and econometric models for forecasting restaurant sales. *International Journal of Hospitality Management*, 11(2):129–142, 1992.

[4] Filippos Giannakas, Christos Troussas, Akrivi Krouska, Cleo Sgouropoulou, and Ioannis Voyiatzis. Xgboost and deep neural network comparison: The case of teams' performance. In Alexandra I. Cristea and Christos Troussas, editors, *Intelligent Tutoring Systems*, pages 343–349, Cham, 2021. Springer International Publishing.

[5] J. Hau. Private community, 2022.

[6] S. Hau. Private community, 2022.

[7] Hallden P. Holmberg M. *Machine Learning for Restaurant Sales Forecast*. PhD thesis, University of Uppsala, Ångströmlaboratoriet Lägerhyddsvägen 1 Hus 4, Plan 0, 2018.

[8] J. Jiang. Private community, 2022.

[9] Junda. Private community, 2022.

[10] D. Karakus. Private community, 2022.

[11] M. Kloos. Private community, 2022.

[12] P. Lihui. Research on logistics cost management of logistics in colleges and universities. *Research Gate*, 1(1):63–64, 2011.

[13] K. S. Lorek, C. L. McDonald, and D. H. Patz. A comparative examination of management forecasts and box-jenkins forecasts of earnings. *Accounting Review*, 1(1):321–330, 1976.

[14] Michael S. Morgan and Pradeep K. Chintagunta. Forecasting restaurant sales using self-selectivity models. *Journal of Retailing and Consumer Services*, 4(2):117–128, 1997.

[15] A. Schulze. Private community, 2022.

[16] W. Wang. The commercial housing consumption forecasting based on grey theory and BP neural network. *J. Hebei Polytech. Univ. Soc. Sci.*, 1:65–67, 2018.

[17] J. Wu. Private community, 2022.

[18] Liu Xinliang and Sun Dandan. University restaurant sales forecast based on BP neural network - in Shanghai Jiao Tong University case. *Advances in Swarm Intelligence*, 2:338–347, 2017.