The Alan Turing Institute

Data Study Group Final Report: Morrisons

5 – 23 July 2021

Optimising the supply chain to minimise waste and delivery mileage

https://doi.org/10.5281/zenodo.6498140

Contents

List of Figures

1 Executive summary

1.1 Challenge overview

Morrisons is the fourth largest supermarket chain in the UK, with 500 stores nationwide that stock thousands of varied products. Ensuring the efficient logistics and distribution of products from suppliers to stores is key to ensuring customers' needs are met, as well as mitigating wider issues such as waste minimisation and reducing carbon emissions.

This challenge will focus on how to optimise the flow of products from suppliers through depots to the stores. The flow of products begins with forecasted sales data, which is the predicted quantity of sales for products on a specific day. Based on this forecast, products flow from suppliers to distribution depots and finally to stores.

Morrisons provided the challenge team with sales forecast data for a 28 day horizon, alongside the start of day stock position for day 0. The challenge is to confirm, for each day in the next 28 days, how much of each product Morrisons should order from the depot and how much they should send out to each of the three stores. This consists of simultaneously ensuring that there is enough stock in store to meet forecasted demand, while minimising the amount of food waste and delivery mileage that may result from Morrisons' supply chain operations. The overall cost function for optimisation is therefore to minimise the cost of wasted goods and the cost of transportation, subject to a variety of assumptions and logistical constraints.

1.2 Related works

In 2018, food and drink retail contributed 2.3% to the UK's national Gross Value Added [4], with the five main supermarket retailers taking around 50% of total market sales [2]. Perishables are key to supermarket profitability [5], but the waste caused by damage and spoilage can contribute to losses of up to 15% [6].

The literature on perishable inventories is extensive and has been reviewed periodically over the past few decades [7, 9, 11, 8]. This work stems from the *news vendor problem*, in which one seeks the optimal

number of newspapers to order for tomorrow given uncertain demand, but with known cumulative distribution [1, 9]. More recent extensions have considered demand that depends on factors such as market price, marketing, stock quantity, supplier pricing policies and buyer risk profiles [10].

Perishable products can have either a fixed known lifetime (e.g. newspapers) or a gradually deteriorating lifetime (e.g. fruit, blood and photographic film) [9]. Our focus will be on fixed-life products, but studies of deteriorating inventories are available in the literature [11, 7]. Early work on fixed-life perishables, prior to modern computing, focuses on the derivation of analytical results [9]. For fixed-life products it is always optimal to issue the oldest units first, known as first-in-first-out. The simplest case is when demand is deterministic, in which case an optimal policy can ensure that no items perish. An example of this is the Economic Order Quantity model. The problem is harder for stochastic demands and so early results focused on cases where the lifetime of a single product is one or two days.

1.3 Data overview

The challenge data comes from Morrisons' Supermarket Simulator, which is a system designed by their data scientists that uses known operational parameters, with the benefit of removing added noise in the data and removing the possibility of breaking anything in the real world. The data subset consists of 55 products across three stores. There are 91 days of data in total, with day 0 being a Monday. The first 63 days are included as historical operational data for training, and therefore have the corresponding actual sales and waste data along with depot-related data. The last 28 days are meant for predicting and optimising the movement of products. This consists of the predicted demand of products for each product, store, and day. The dataset consists of five subsets that were provided to the challenge team. These are:

- The 28-day forecast of product sales across each store and day.
- Store details (actual sales, sales forecast, waste and deliveries).
- Depot details (total stock ordered, total stock delivered).

- Product details (dimensions, shelf-life, minimum order quantity).
- Mileage between all stores and the depot.

1.4 Main objectives

The principal aim of this challenge is to optimise the flow of products from suppliers through depots to the stores, whilst ensuring stock is always available for customers, and minimising both excess product waste and carbon emissions from transportation.

The specific objectives are:

- Minimise the surplus and corresponding waste of products in stores whilst minimising the delivery mileage.
- Identify frequently wasted products and find alternative solutions that could reduce the waste of these products.
- Identify how changing global system parameters could improve the overall output. These parameters are:
  - the cost of transportation,
  - the minimum order quantity, and
  - the accuracy of the forecast.

1.5 Approach

The team approached the problem using three streams of work.

Firstly, in order to better understand the data, we used descriptive, statistical and econometric techniques to differentiate the main features, and analyse the characteristics of the data. This was used to identify where the most waste is occurring by store and product, and investigate possible reasons. We use a combination of descriptive statistics, Principle Component Analysis and K-Means clustering. We also use a simple regression analysis to examine the accuracy of the forecast, and determine if there is a relationship between store forecast and waste.

Secondly, the team compared different machine learning approaches to examine whether it would be possible to (1) predict waste in store using the forecast data and (2) predict the deliveries needed from the depot using the forecast data.

In a third stream of work, team members focused on solving the main optimisation problem along the supply chain. For this, the team:

1. re-framed the challenge as a Mixed-Integer Linear Programming (MILP) problem;
2. examined the possibility of solving the problem for one period following the *news vendor problem*;
3. explored the opportunity to use reinforcement learning as an alternative solution to MILP, as this may be more computationally tractable for larger data-sets when considering more products, multiple depots and a greater number of stores.

1.6 Main conclusions

In summary, the main conclusions are:

- The total waste generated is inconsistent between stores and products.
- Clustering methods are useful for grouping products, and lead to the discovery that products with a short shelf life are the most wasted.
- A model based on a well-performing store can be used to predict deliveries on poor-performing stores, resulting in reduced waste.
- Ordering new products every shelf-life period may help reduce waste.
- Reinforcement learning shows potential as a method of optimising product flow for a large number of products, stores and depots.

We find that store waste is inconsistent between the three stores, with two wasting considerably more than the other. Waste is also inconsistent between products, with some being wasted much more than others. One of the causes of waste is a result of products being delivered despite no sales being forecast.

It is possible to identify which stores are producing the least waste through data exploration tools such as principal component analysis (PCA). We also show that products can be grouped by their performance in total waste and sales using clustering methods. This can be useful for determining which products are performing poorly (through high waste or low sales) across multiple stores using computational methods rather than manual analysis. In general, we find the products that are wasted the most are those with the shortest self lives, but there are some exceptions.

We find that averaging forecasted sales over the last three days is the best predictor of deliveries made to stores, and Store 3 could be predicted with the best accuracy compared to the other stores. As a result, a model based on Store 3 only can be used to predict deliveries required in Stores 1 and 2. This results in less waste as these stores at the cost of fewer sales, where the reduction in waste is considerably higher than the reduction in sales.

Using Mixed Integer Linear Programming (MILP), we discovered the optimal order to sell products is by their expiry dates compared to a random a ordering. We find that ordering new products for every shelf-life period may be well-suited to reducing waste. That is, if a product lasts five days, new stock should be ordered every five days. However, this conclusion was observed when the ordering constraints at the depot (such as minimum order quantity) were removed. Removing this constraint enables smaller orders to be made, therefore helping to reduce waste. The MILP solution results in a lower waste than the historical baseline.

Reinforcement learning is a viable alternative to MILP that theoretically scales much better. Time constraints meant that a formulation was devised but a solution was not obtained. Further work would be needed to test this as a result.

1.7 Limitations

Using MILP to find the optimal solution for deliveries over all products, stores and days within the given problem is too computationally intensive and so cannot be solved in a feasible amount of time. As this is a limited example (with three stores and 55 products), it would be impossible to

find the optimal solution for all stores and products in the real-world. Therefore, as part of this case-study, simplifications were made to make the problem tractable. These include solving for different products separately, not aiming for full full optimality, and constraining the solution space so that stock is bounded by the sales forecast.

1.8 Recommendations and future work

We recommend using MILP to optimise sub-problems instead of tackling the problem as a whole. Alternatively, solvers that quantify the optimality gap may also be useful. The solvers can be stopped when the optimimality gap is below the desired level, rather than stopping after a fixed time.

In future work, it may be worth exploring if maximising profit instead of minimising waste is a more useful goal to pursue. Doing so will give different solutions due to different product profit-margins. We also expect that reinforcement learning will be more computational tractable than MILP.

Multi-level modelling techniques of statistical analysis may be used to unpick the variance of waste at store by dimensions of products, stores and days. It may also be beneficial to explore the temporal auto-correlations, to understand how days of the week affect forecast and waste, and to explore the relationship between forecast uncertainty and the uncertainty in the optimised solution.

2 Data Exploration and Statistical Analysis

2.1 Dataset Exploration

The challenge data comes from Morrisons Supermarket Simulator. The data subset consists of 55 products across 3 stores. There are 91 days of data in total, with day 0 being a Monday. The first 63 days are included as historical operational data for training and therefore have the corresponding actual sales and waste data, along with depot data. The last 28 days are intended for prediction of the best logistical optimisation, and consist of the predicted demand for each product in each store on

each day, but does not have corresponding sales and waste data. Five datasets were provided to the challenge team: the 28 day forecast data, store details data, depot detail data, product meta data and store mileage data.

2.1.1 What is driving waste in stores?

To begin, the team looked at some of the characteristics of individual stores and how they compare to one another. The most obvious finding on examining this data was that Store 3 showed the least waste over time, compared to Stores 1 & 2, as shown in Fig. 1. We noticed that deliveries were lower overall in Store 3 (shown in Fig. 2) despite the overall forecast being higher.

Fig. 3 shows the actual versus forecasted sales for each product. Closer examination of individual products in shows that there are specific products that are driving the majority of waste. There are two products (9 and 55) that see no sales over the whole period, despite the forecast showing sales. These products could easily be removed from the supply chain to cut waste in those areas. As displayed in Fig. 4, there are products where the delivery from depot is disproportionate to the sales being made. In addition, products 18, 19, 34, 50 and 52 are producing a lot of waste across stores.

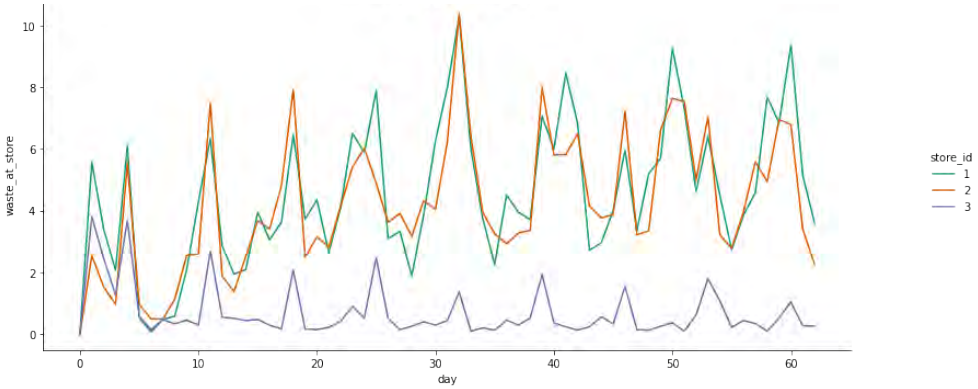


Fig. 1: Waste by day for all stores

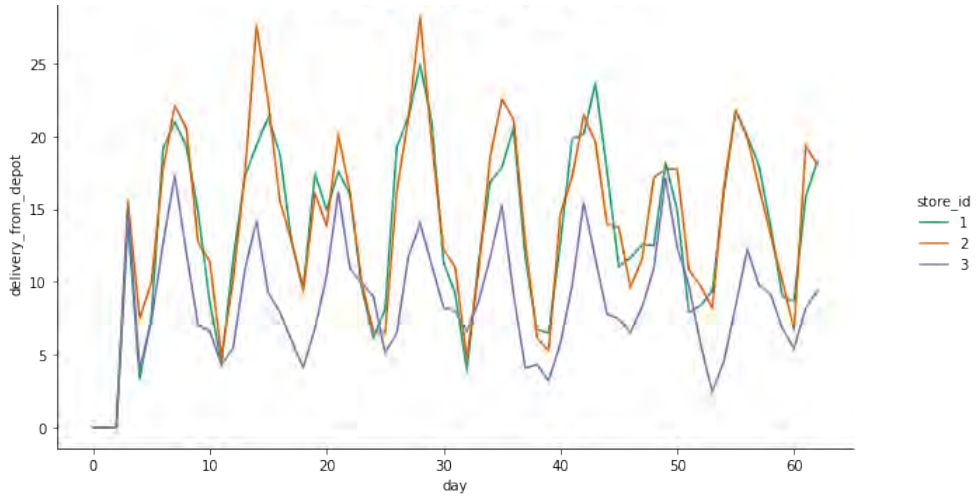


Fig. 2: Deliveries from depot to all stores

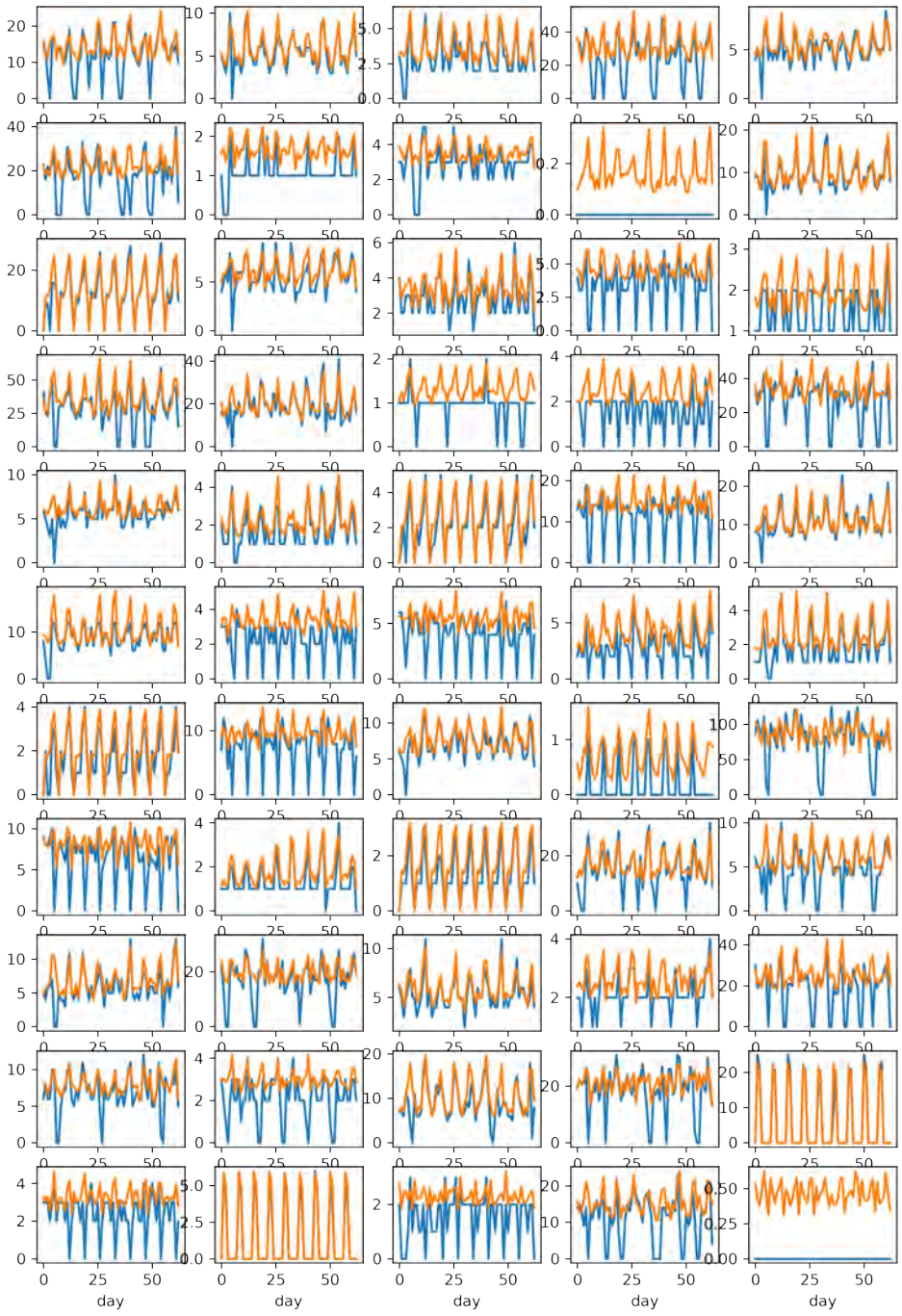
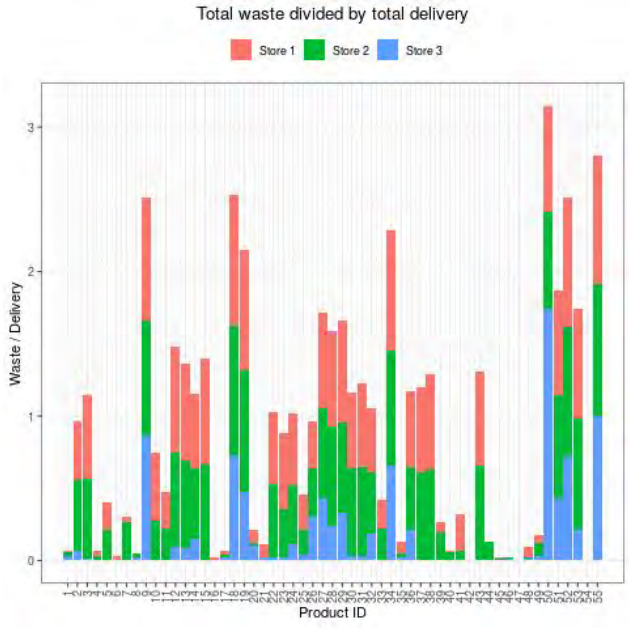
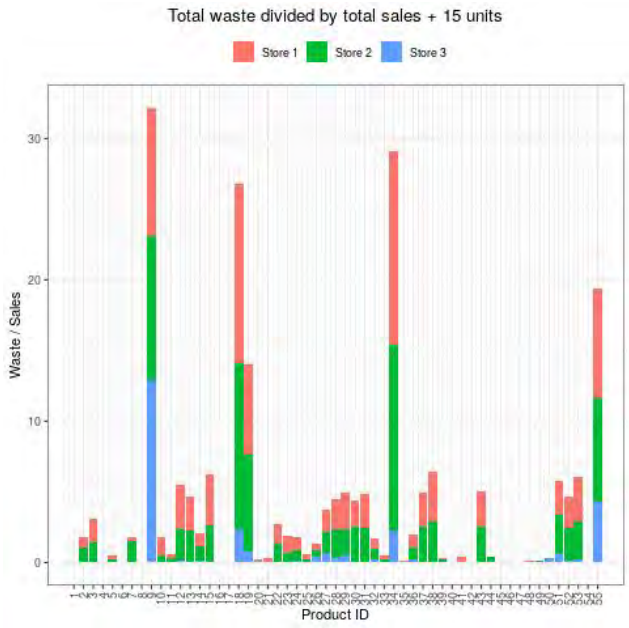


Fig. 3: Store 1 actual (blue) and forecasted (orange) sales



(a) Total waste to *delivery* ratio for each product and each store



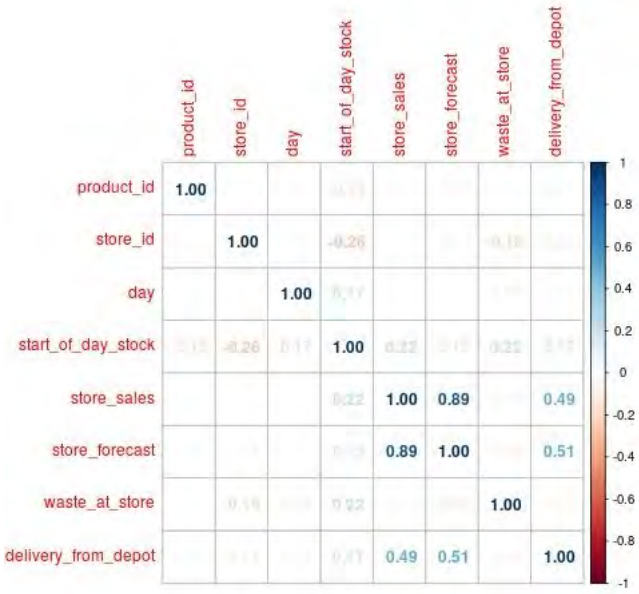
(b) Total waste to *sales* ratio for each product and each store

Fig. 4: Waste to delivery (top) and sales (bottom) ratio

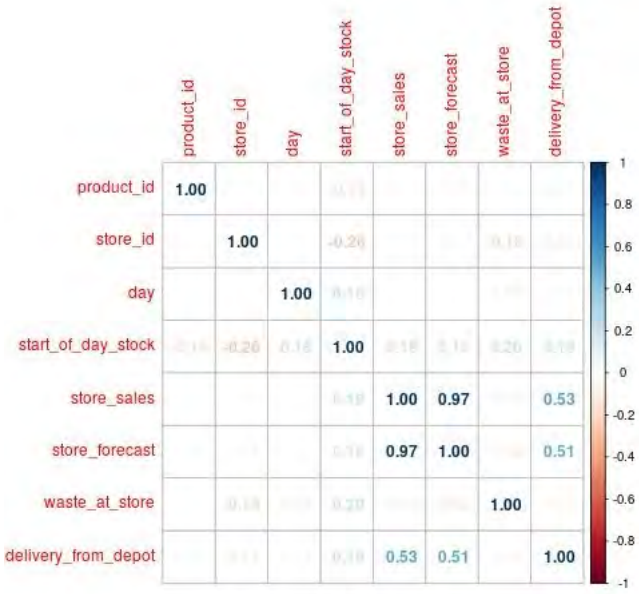
2.1.2 How accurate is the forecast?

Besides products 9 and 55, examination of the forecast does seem to show a high level of accuracy. As the correlation matrices in Fig. 5 show, the correlation between forecast and sales is 0.89, and when removing the observations when stock in store was zero (i.e. the product could not sell because it was not available), the correlation increases to 0.97.

We also found that the correlations between forecasted sales of products are mostly positive (see Fig. 6b). The correlations between products for actual sales are less significant, see Fig. 6a, with more correlations close to 0. Two products are highly negatively correlated with all other products on the forecasted sales. These are products 50 and 52, which exhibit a dark purple shade. For the correlation of actual sales between products, products 9 and 55 are missing as there are no sales of these products during days 0–62.

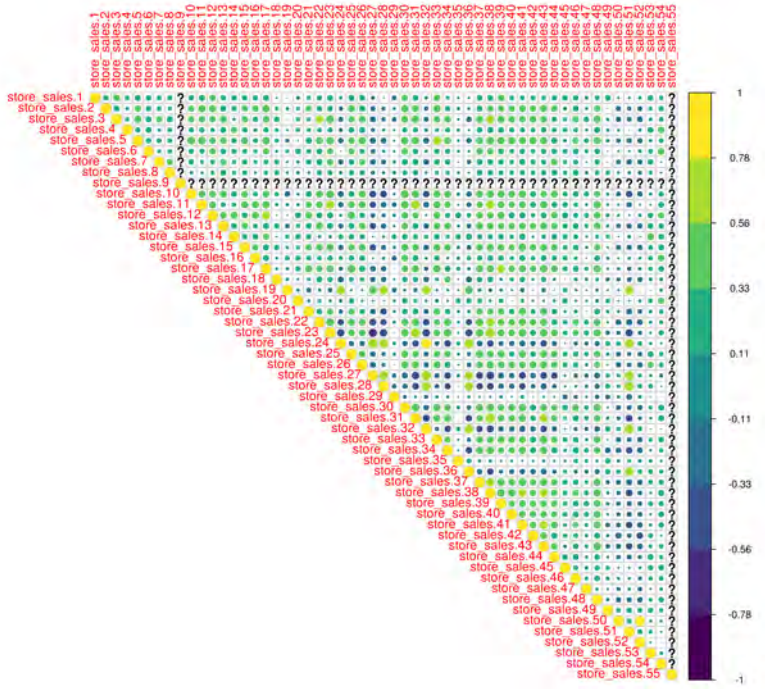


(a) Correlations without removing zero stock

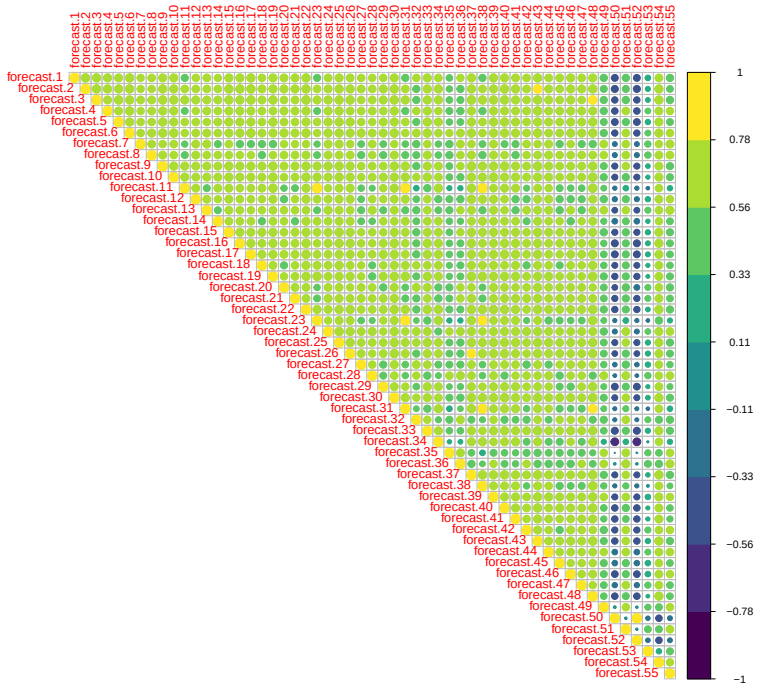


(b) Correlations after removing zero stock

Fig. 5: Correlations between variables in Store detail dataset



(a) Actual Sales between products



(b) Sale forecasts between products

Fig. 6: Correlations between products

When looking at specific products, plotting the residuals (forecast minus the sales), the accuracy of the forecast also seems to be high. Fig. 7 shows that the residuals have a long tail, with the means given as the dotted lines. However, after we removed data where the start of the day stock was zero and the delivery from depot was zero, the tails mostly subsided and the residuals are much closer to zero, suggesting a high accuracy. This is shown in Fig. 8.

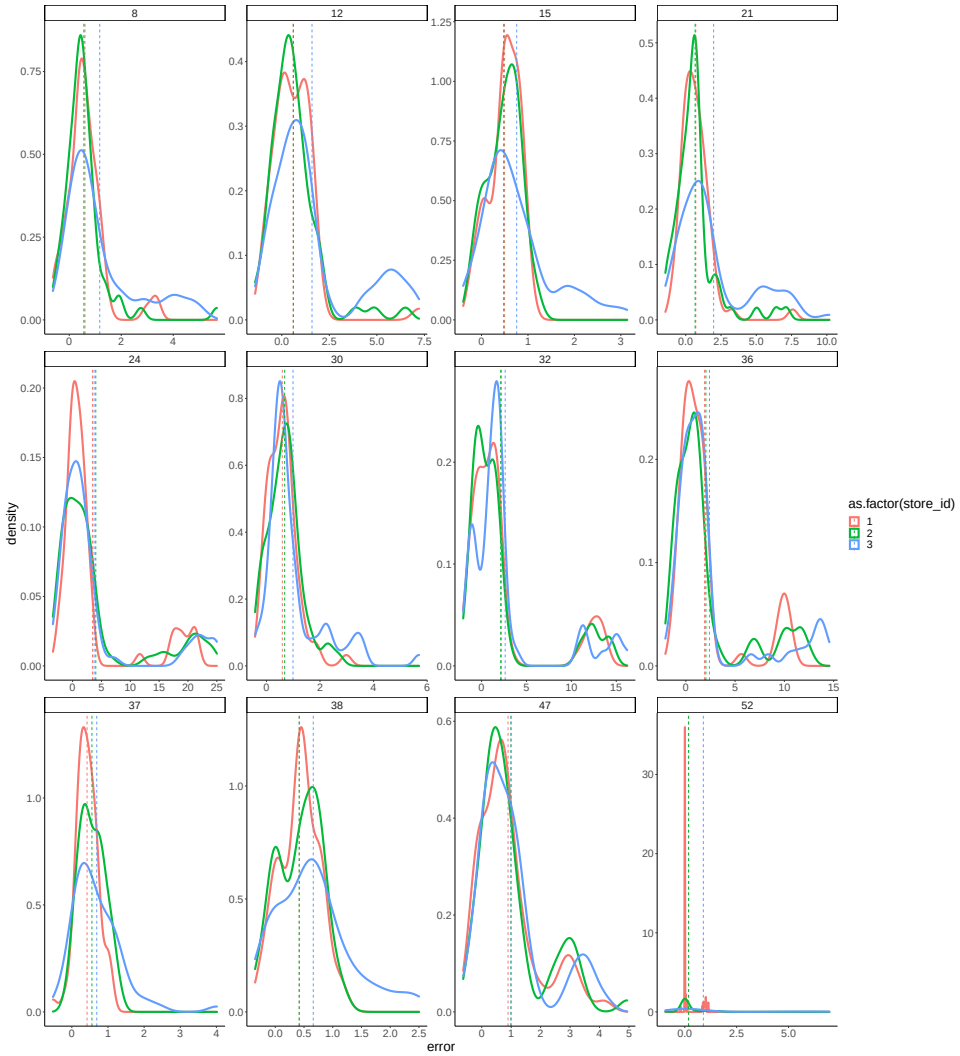


Fig. 7: Store forecast error of selected products

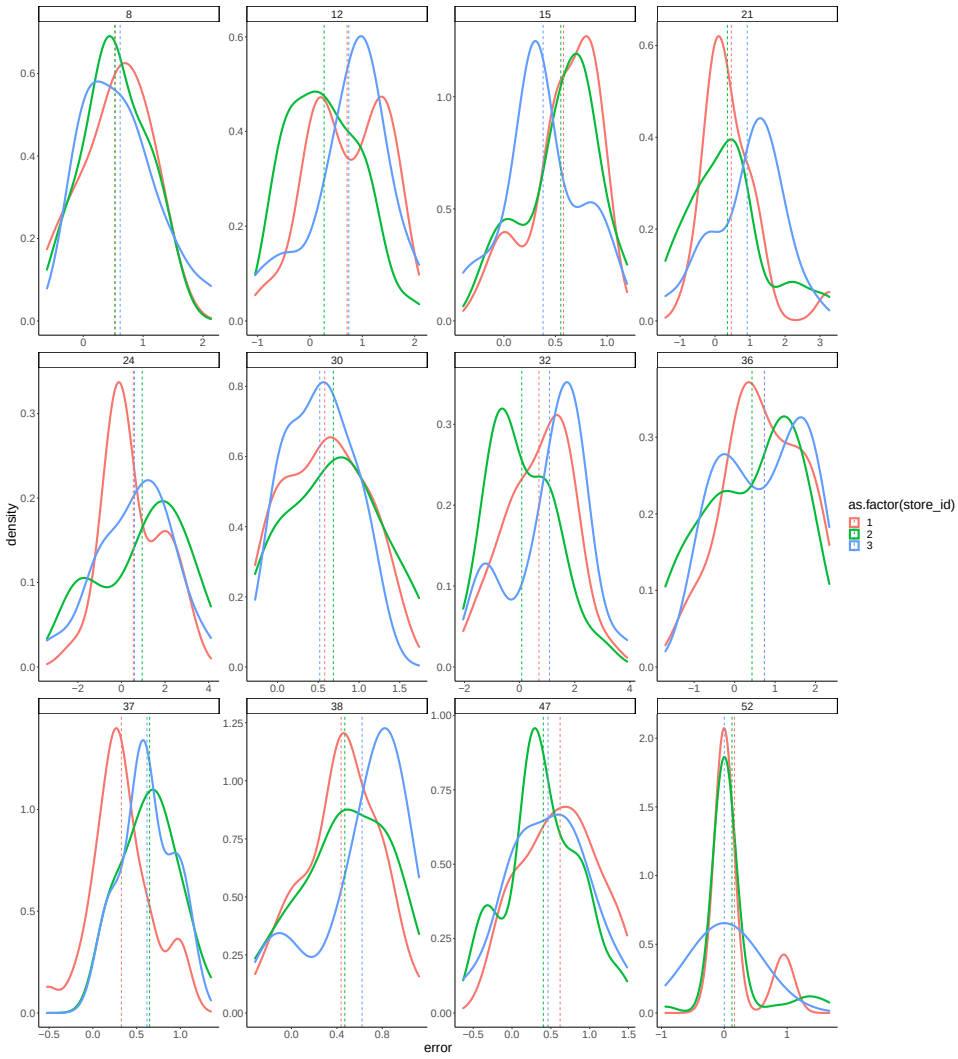


Fig. 8: Store forecast error of selected products after excluding zero ”start of day stock” and ”delivery from depot” data points

2.2 Data quality issues

The following are some data quality issues that were encountered during the project:

- **Null values** - This was intentional missing data, with null values representing no forecasted demand. Missing sale forecasts were

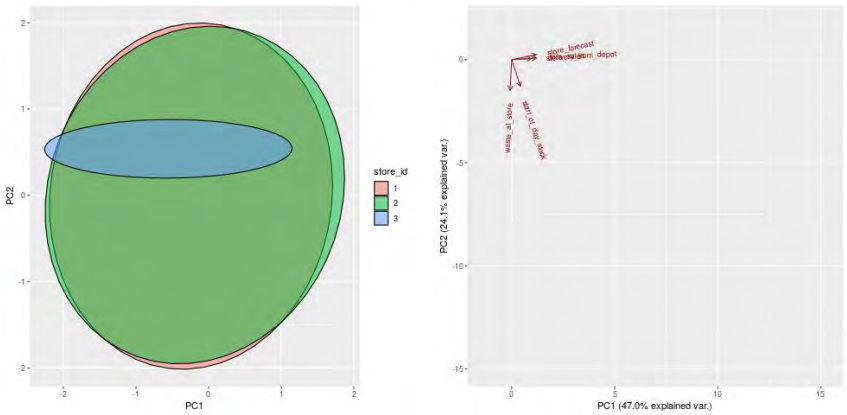
imputed as zeros. 3.6% of missing values were imputed this way.

- **Biased prediction of sales forecasts** - The forecasts are systematically positively biased and skewed against true historical sales. This is understandable to ensure the sufficient stock of the stores to meet customers' demands, but can result in overstocking and waste at store.
- **Inconsistent data types** - Whilst the sales are integers, the forecasts are actually floating values.
- **Incomplete information** - Information about the selling price of the products was not provided. We therefore are not able to calculate the profit margin, or use it to facilitate alternative optimisation objectives.

2.3 Statistical analysis

2.3.1 Principal Component Analysis and k-means clustering on stores and products

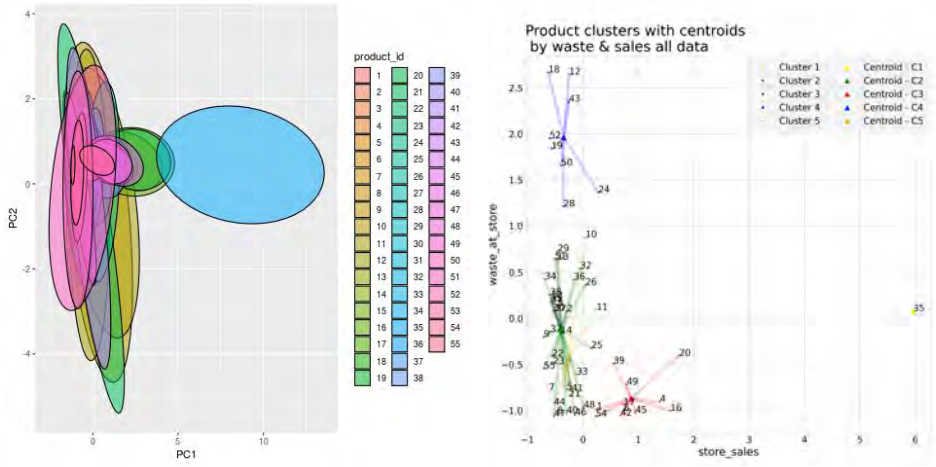
In a continuation of the product level examination provided in Section 2.1 above, Principal Component Analysis (PCA) was implemented to examine how variation between stores and products was best captured by existing variables in the Store Detail data. Using the variables of store forecast, store sales, start of day stock, waste at store, and delivery from the depot, we found that 71% of the variation could be explained by the first two principle components (PC), with PC1 explaining 47% of the variance whilst PC2 accounted for 24%. Variable store forecasts, store sales and deliveries from the depot mostly contributed to PC1, and the variables start of day stock and waste at store mostly contributed to PC2. In Fig. 9b it is shown visually that the two groups of variables that contribute the most to the principal components are roughly orthogonal, suggesting they are independent of each other. Fig. 9a shows that Stores 1 and 2 are very similar along both principal components and that Store 3 shows the least variation along PC2, suggesting that it varies least in terms of waste and amount of stock. Coupled with the analysis from Fig. 4, where it was shown that Store 3 produces the least waste, this further suggests that Store 3 minimises waste better than the others.



(a) PCA by stores (b) PCA by variables

Fig. 9: PCA of stores and variables

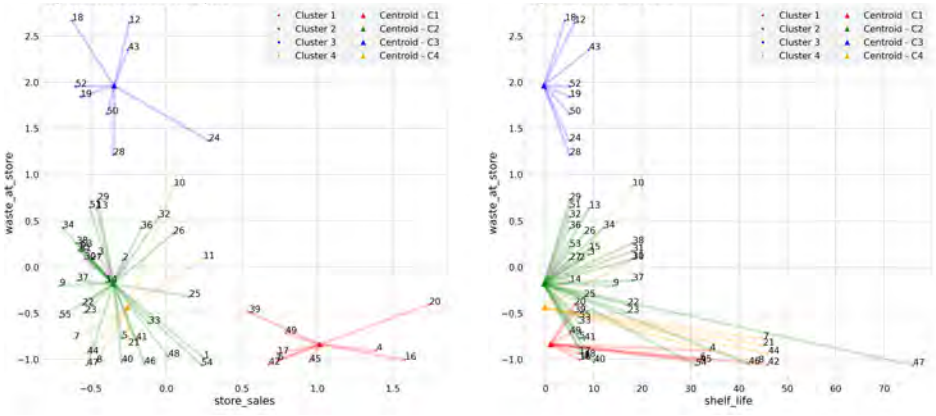
At the product level, PCA and k-means cluster analysis portray a similar story about a high-performing product outlier (product 35) which consistently sold more and wasted less. In Fig. 10b, product 35 is a clear outlier within the cluster plot. This is despite having a short shelf life, which was 5 days against the mean of 15.6 days and the minimum of 4 days. Fig. 10 also shows which products are clustered in terms of waste and sales in the PC space. The blue cluster also appears to capture products previously shown in Fig. 4 that are suspected of producing a lot of waste, even after accounting for sales and deliveries.



(a) Waste by day for all stores (b) All products

Fig. 10: PCA on products and Clustering to identify outliers

When removing product 35, it becomes clearer which products have combined higher sales and lower waste. Fig. 11a visually shows that the red cluster contains products with high sales and low waste in the PC space, with the blue cluster containing products that produce significantly more waste than others, whilst having comparable sales to the green and yellow clusters. In Fig. 11b, the high waste products are shown to be generally those with the shortest shelf-life. However, this is not an absolute relationship given that many of the green cluster products also have a short shelf life but considerably less waste. The products in the blue cluster clearly under-perform and could be further optimised.



(a) Waste vs. sales clusters (b) Waste vs. shelf life clusters

Fig. 11: Clustering on all products excluding 35

2.3.2 Statistical regression modelling of the store waste on the store forecast

We consider the Store dataset, in which the total observations is 10,395 (55 products $\times$ 63 days $\times$ 3 stores). The observations are product/day combinations over the first 63 days, with each observation having a corresponding product forecast and waste. The aim was to try to model the waste as the response (dependent) variable, denoted as y, by taking the forecast as the input (independent) variable, denoted as x. The new dataset was split into two parts, training data (80%) and testing data (20%).

In an initial data exploration, waste (an integer) takes an excessive number of zeros (78.6%). Statistical models which deal with the zeros (so-called Zero-inflated models) would be useful to predict store wastage. The computed mean of the waste is 3.14 and the variance is 74.96, implying that the data distribution of waste is very dispersed and thus varies considerably over the products/days. We observed that the correlation between the Store forecast (x) and the waste at Store (y) is very low (-0.0832), but it is significant $(p < 0.001)$.

Five regression models were fitted for the training dataset. A Classical Linear Regression Model (CLRM) was used as a simple baseline and is

shown mathematically in Eq. 1, where $\boldsymbol{\beta} = (\beta_1, \beta_2)$, β_1 is the intercept, β_2 is slope parameter and E denotes the expectation of y given x.

$$E(y|x) = \boldsymbol{\beta}^T x \tag{1}$$

A Poisson Regression Model (PRM) is typically used for modelling data where the mean and variance are the same for the response variable. It can be described mathematically in Eq. 2, where $\boldsymbol{\beta} = (\beta_1, \beta_2)$, and β_1 and β_2 are the intercept and slope parameters, respectively.

$$E(y|x) = \exp(\boldsymbol{\beta}^T x) \tag{2}$$

A Zero Inflated Poisson Regression Model (ZIPRM) is commonly used when the response variable has excessive zeros. Mathematically, it is given by Eq. 3:

$$E(y|x) = \exp(\boldsymbol{\beta}^T x) \tag{3}$$

where y follows the following distribution:

$$P(y|x) = \begin{cases} \theta + (1-\theta)p(\lambda, 0), & \text{if } y = 0 \\ (1-\theta)p(\lambda, y), & \text{if } y > 0, \end{cases} \tag{4}$$

In Eq. 4, $p(\lambda, 0) = \exp(-\lambda)$, and $p(\lambda, y) = \frac{\exp(-\lambda)\lambda^y}{y!}$. The θ term corresponds to the proportion of zeros in the response variable and λ corresponds to the mean of x.

A Negative Binomial Regression Model (NBRM) is used for modelling when the mean of the response variable is smaller than the variance. It is mathematically described in Eqs. 5 and 6, where τ is the shape parameter which quantifies the amount of over-dispersion, λ is the mean and Γ is the gamma function.

$$E(y|x) = \exp(\boldsymbol{\beta}^T x) \tag{5}$$

$$P(y|x) = \frac{\Gamma(y+\tau)}{y!\Gamma(\tau)}(\frac{\tau}{\lambda+\tau})^\tau(\frac{\lambda}{\lambda+\tau})^y \tag{6}$$

A Zero Inflated Negative Binomial Regression Model (ZINBRM) is used when the response variable has an excessive number of zeros with high variance. It is given as Eq. 7:

$$E(y|x) = \exp(\boldsymbol{\beta}^T x) \tag{7}$$

where y follows the following conditional distribution:

$$P(y|x) = \begin{cases} p + (1-p)(\frac{\tau}{\lambda+\tau})^{\tau}, & \text{if } y = 0 \\ (1-p)\frac{\Gamma(y+\tau)}{y!\Gamma(\tau)}(\frac{\tau}{\lambda+\tau})^{\tau}(\frac{\lambda}{\lambda+\tau})^{y}, & \text{if } y > 0. \end{cases} \tag{8}$$

After training, each model predicted expected store waste for the test dataset and the root mean squared (RMSE) for each of the model was calculated, the results of which can be found in Table 1. The R-software packages *pscl* and *mass* were used to compute the regression coefficients of the models.

It is observed that the zero inflated models (ZIPRM and ZINBRM) have the smallest RMSE, with a wastage prediction error of 9.07 units on average. The explanation for the greater efficiency of the zero-inflated models could be their ability to give appropriate weights to zero values of waste at store.

Table 1: Prediction RMSE for different methods

Model	**Prediction Error (RMSE)**	**Properties of the model**
CLRM	9.11	Classical Linear regression model
Poisson Reg.	9.31	Integer values regression
Zero inflated Poisson Reg.	**9.07**	Integer valued and zero weighted regression
Negative Binomial Reg.	9.30	Integer valued regression
Zero inflated Negative Binomial Reg.	**9.07**	Integer valued and zero-weighted regression

3 Machine learning to predict store waste and deliveries needed from depot

3.1 Predicting waste based on sales forecast

In this section, we explored the use of using the input variable of sales forecast to predict the waste at store. The sales forecast data is taken from the 28 days of historical data provided. The goal is to understand the accuracy of the waste prediction, and to provide a baseline model where the additional input variable (detailing deliveries from the depot) and the 12 logistical constraints are ignored.

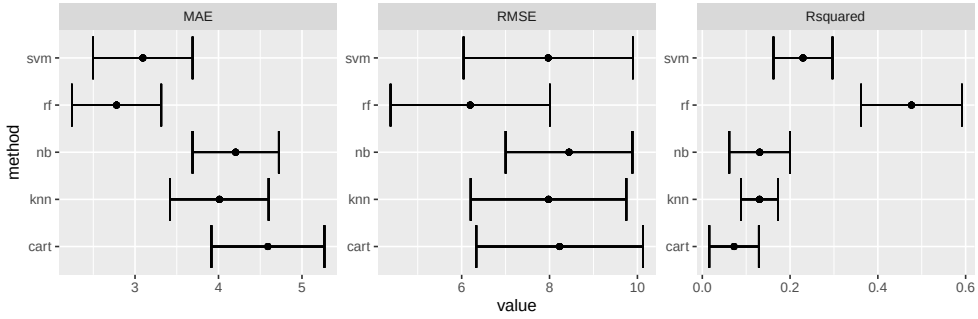


Fig. 12: Model comparison (mean values with 95% confidence intervals)

For simplicity, we used the same set of variables across all the algorithms and only included sale forecast, product id, store id, day, and day of the week in the model. We compared the results of five different machine learning algorithms:

- Classification and Regression Trees (CART).
- k-Nearest neighbours (kNN).
- Support Vector Machines (SVM).
- Negative binomial Generalised Linear Model (NB).
- Random Forest (RF).

Days 0–62 were used as training data, and days 63–90 were used for testing. We further split the training set, using 80% as a training subset

and 20% as a validation set. We then compared the performance of the five above-listed algorithms in the training subset using routine criteria, including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and R-squared (see Fig. 12). Among all five algorithms, Random Forest (RF, see the second row of Fig. 12) is found to be the most successful in predicting waste at store. It has the lowest MAE and RMSE, and its R squared is highest. However, even the most successful model (RF) is proven to be insufficient in predicting waste in stores, as it could only explain approximately 50% of the variance in store waste. This may be because the model is too simple. To address this, we next introduce deliveries from the depot into our models.

3.2 Predicting deliveries required from depot based on forecast sales

Building on the previous section, in which the tree-based models performed well, we continue exploring methods of predicting deliveries from the forecast using this type of model. More specifically, we are using the gradient boosting trees method, implemented using the extreme gradient boosting package [3].

3.2.1 Using the last n forecast days as predictors

The previous section used only one predictor (sales forecast). To improve the predictive power of the model, we included product features as predictors (shelf life, weight, price, minimum order quantity and pack size). We have also engineered features using the variable forcasted sales: lag 1 (forcasted sales lagged by 1 day), avg 3, avg 4, avg 7 (moving averages over the last 3, 4, and 7 days of the forecast), and the day of the week, from Monday to Sunday. The outcome variable was delivery to stores the next day. This is in contrast to the previous section, where the outcome variable was store waste. In this case, we are using delivery to store as an outcome variable to learn about delivery patterns that may reduce waste.

All predictors were centred (by the mean) and scaled (by the standard deviation). The categorical variable day-of-the-week was encoded using the one-hot encoding procedure.

Cross-validation (CV) was performed with five folds. We used a maximum of 100 rounds of iterations on the training data with a validation set stopping if the performance does not improve for 15 rounds. The learning rate η was set as 0.4 and the maximum tree depth was set as 6. The evaluation metric was the root-mean-square error (RMSE). CV was performed on 56 days of historical data (instead of 63). This is because we used a moving average of 7 days as one of our predictors which resulted in missing values for the first 7 days for that feature. We excluded all rows of data including missing values induced by this feature. The best iteration model had a training RMSE of 12 and a test RMSE of 14.9 on CV. The final model was then trained on the 56 days of the historical data using the number of iterations learnt in CV to further prevent over-fitting.

The cumulative deliveries predicted by the model are presented in Fig. 13. Note that there are no predictions for the first 7 days as we used a 7-day moving average as one of the features. We can see that the model approximates the deliveries well. The root-mean-square error for the model on the cumulative deliveries was 246.3.

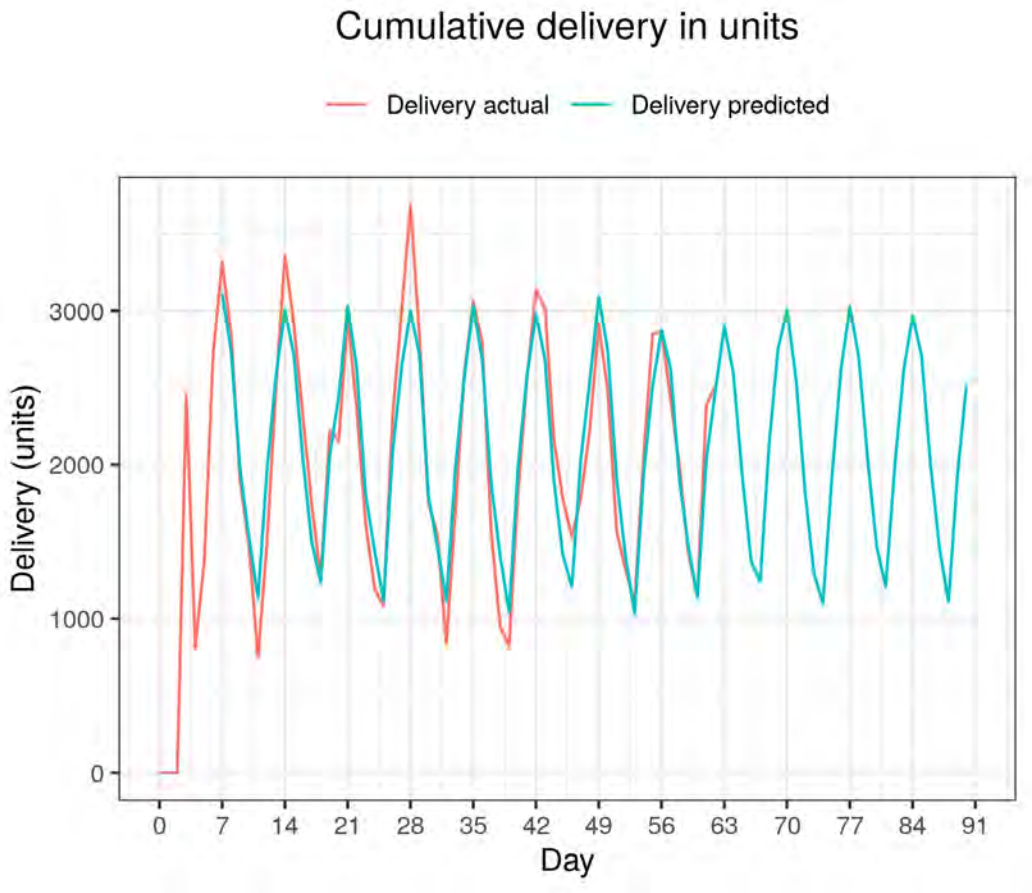


Fig. 13: Predicted and actual cumulative deliveries across all store and all products.

One of the advantages of using the extreme boosting R package is that it can provide us information about feature importance. The feature importance can be measured in different ways. Most commonly, in the case of boosting trees, it is measured using gain, which is an improvement in accuracy that each feature brings to the tree branches it is on. In other words, it tells us how much the accuracy of the model improves after adding branches that are split on this feature.

Fig. 14 shows the feature importance measured in gain for the learnt model. We can see that the moving average of 3 days was by far the most important feature, followed by the moving average of 4 days and a lag of

one day. All other features had less importance. This suggest that perhaps the 3-day forecast was used by stores to set the delivery schedule.

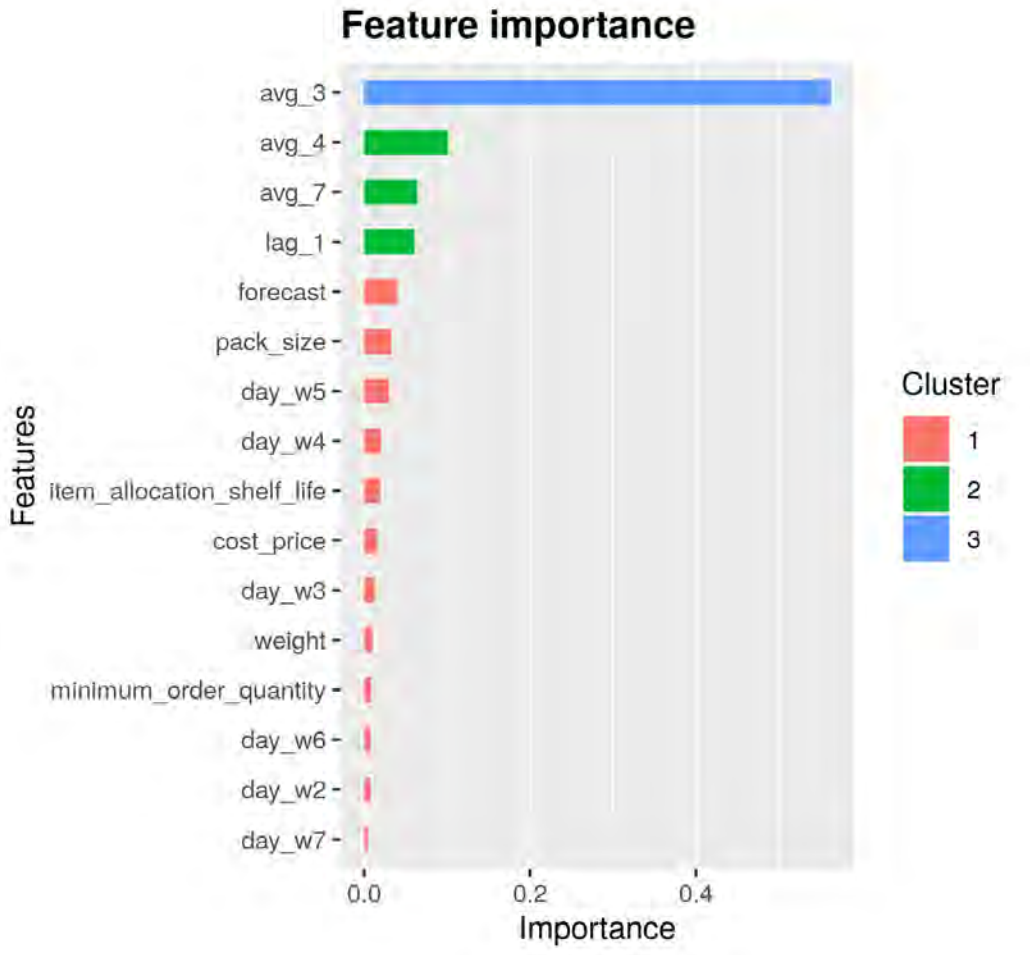


Fig. 14: Importance of each feature in making predictions (measured in gain). Methods are clustered by their importance.

3.2.2 Using the next n forecast days as predictors

Using lagged variables and moving averages over the past n days are standard ways of creating new features when dealing with time series data. However, a potential alternative method is to consider the amount of each product to be delivered to each store when deciding the company's

forecast over the next n days. Therefore, we explore a model that uses the next n days of forecast deliveries (not actual) as a predictor.

Using the same parameters as described in the previous section, we built a boosting tree model that uses forcasted sales for *the next* day (referred to as lead 1) and the average over the next 3, 4 and 7 days (referred to as avg 3, avg 4, avg 7) of the forecast. We have kept other features in, including the day of the week and features describing products (size, shelf life, etc.). The best iteration model had a training RMSE of 11.2 and a test RMSE of 14.2 on CV. The final model was trained on all 63 days of historical data.

Fig. 15 shows the cumulative deliveries predicted by the new model. Note that there are no predictions for the last 7 days as we used a 7-day moving average over the next 7 days, which resulted in missing values for these days. We can see that this model is a worse fit compared to the one we built in the previous section. The RMSE on the cumulative deliveries was 577.

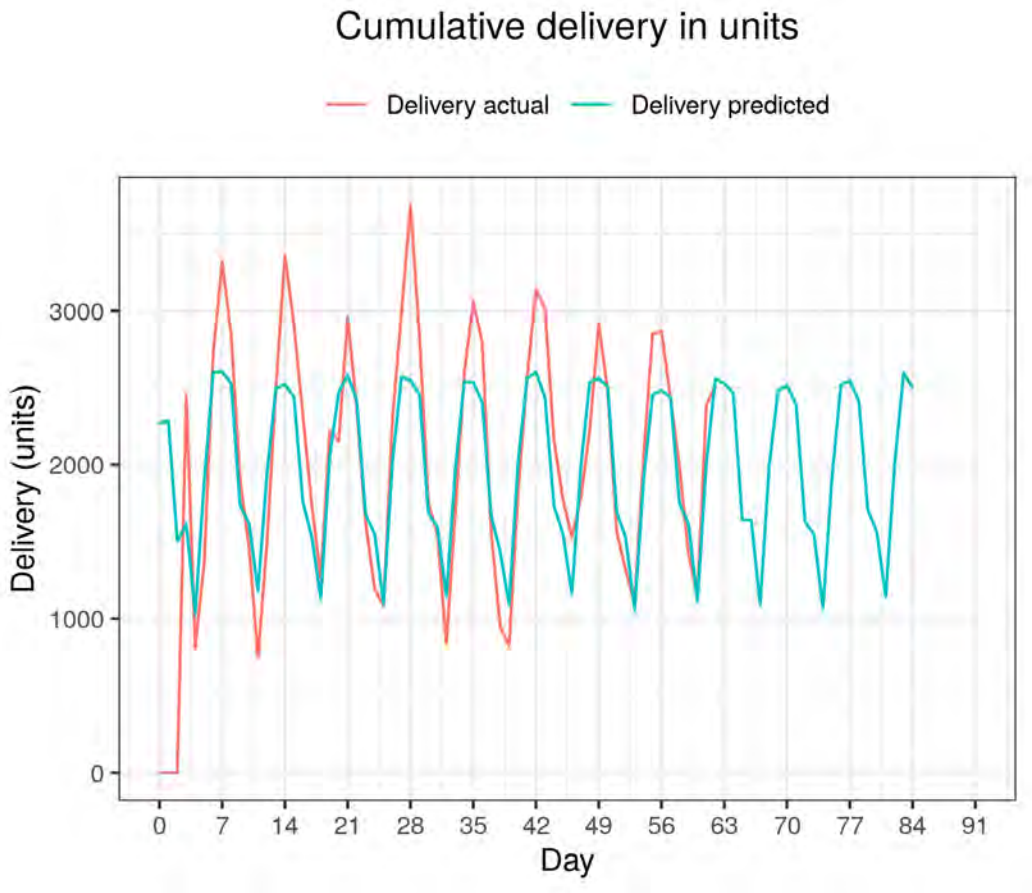


Fig. 15: Predicted and actual cumulative deliveries across all store and all products using the next n days a predictors.

Fig. 16 shows the importance of each feature in predicting deliveries. This shows the moving average of 3 days was the most important feature. However, other features, such as moving averages of 4 and 7 days, were significantly more important in predicting deliveries than they were in the model described in the previous section.

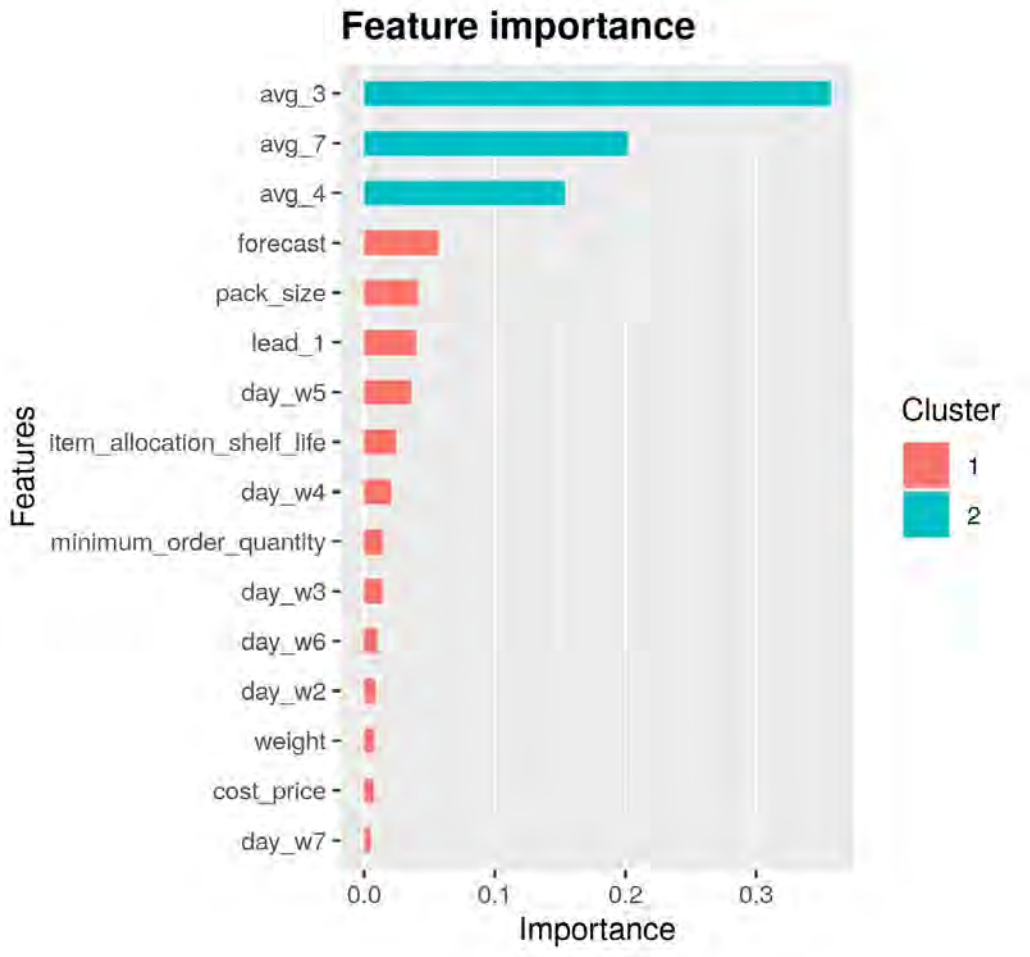


Fig. 16: Importance of each feature in predicting deliveries (measured in gain) for the model using moving averages over the next n days as predictors. Methods are clustered by their importance.

From Fig. 15, we can also see that actual deliveries for all products and all stores were 0 in the first three days. This could induce unjustified bias in predicting deliveries as nowhere else in the time series were deliveries equal to 0 for all stores and all products. We thus excluded the first 7 days from the training set and retrained the model. Fig. 17 shows predicted deliveries of the new model. We can see that these look similar to those in the previous section, with an RMSE on the cumulative

deliveries of 252.6.

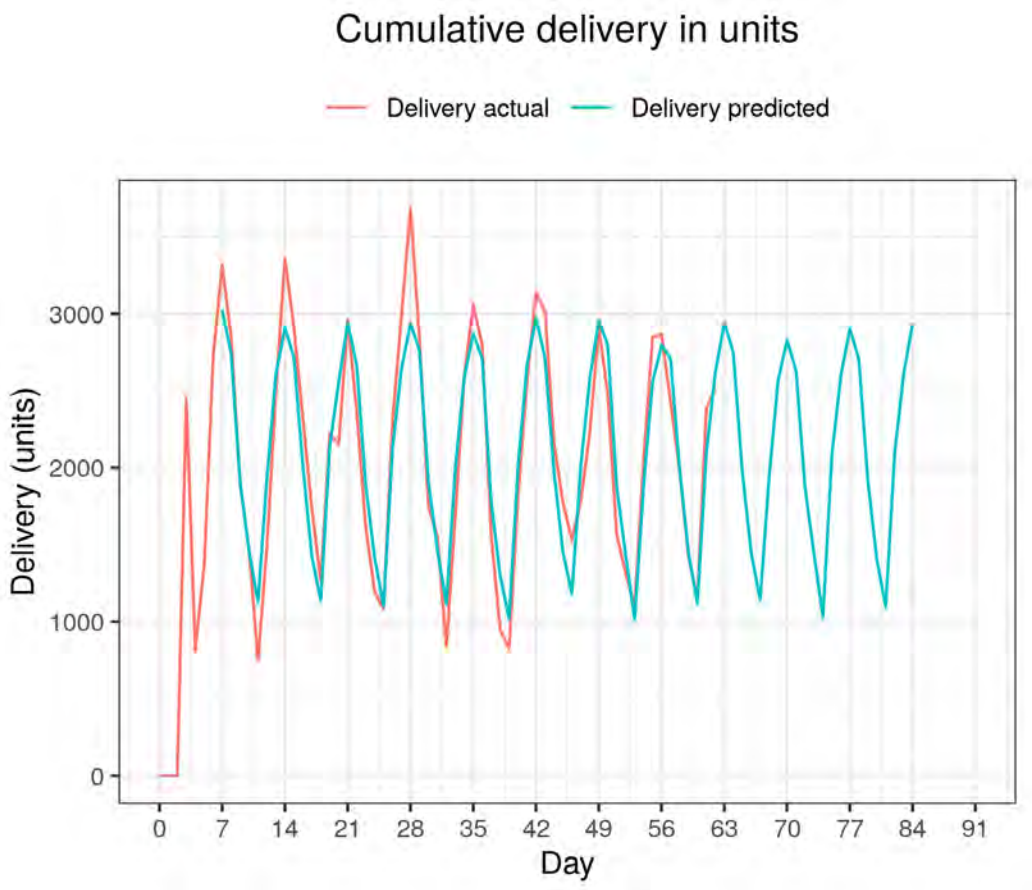


Fig. 17: Predicted and actual cumulative deliveries across all stores and all products using the next n days as predictors.

Fig. 18 shows the importance of each feature in predicting deliveries to stores. We can see that this plot is closer to the one where we used moving averages over that last n days (see Fig. 14). It shows that the moving average over the next 3 days was the most important predictor. However, the moving average over the next 7 days is a more important feature than the moving overage over the last 7 days in the model. Overall, the models using last n and next n days have similar predictions on the cumulative deliveries.

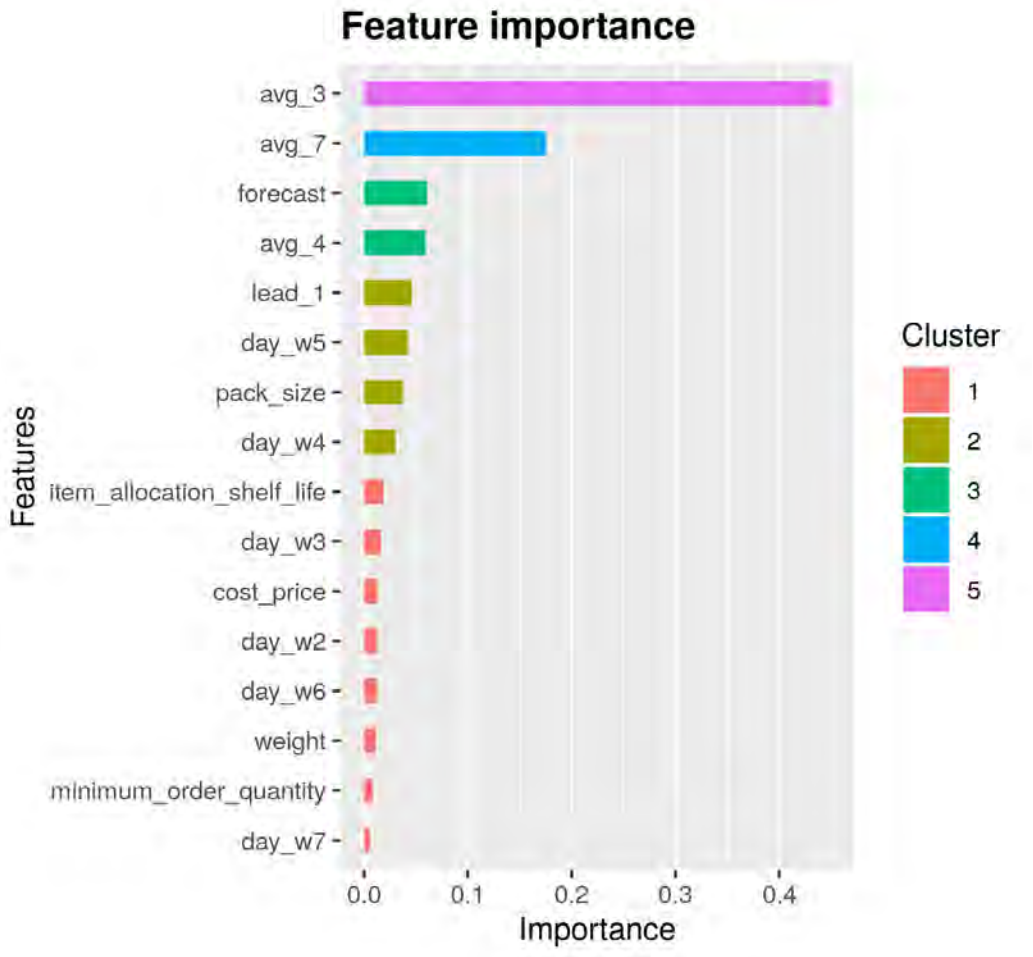


Fig. 18: Importance of each feature in making predictions of store deliveries (measured in gain) for the model using moving averages over the next n days as predictors.

3.2.3 Learning deliveries from Store 3 data

The exploratory analyses have shown that Store 3 is producing the least waste and that lower deliveries and sales do not account for this lower waste. One potential explanation of this result is that product deliveries to Store 3 have been better optimised than those of Stores 1 and 2. If this is the case, we can build a model for Store 3 only and use this model to predict deliveries for Stores 1 and 2, which may help reduce waste at

these two stores.

We built a model based on Store 3 data. As there is minimal difference between using the last or next n days of forecast as predictors, we opted for using the last n days as in this case we automatically do not use the first 7 days of delivery data. All other parameters stay the same as in the models above. The best model had a training RMSE of 5.4 and a test RMSE of 10.1 on CV.

Fig. 19 shows actual and predicted cumulative deliveries for Store 3. The RMSE on these deliveries is 59.3.

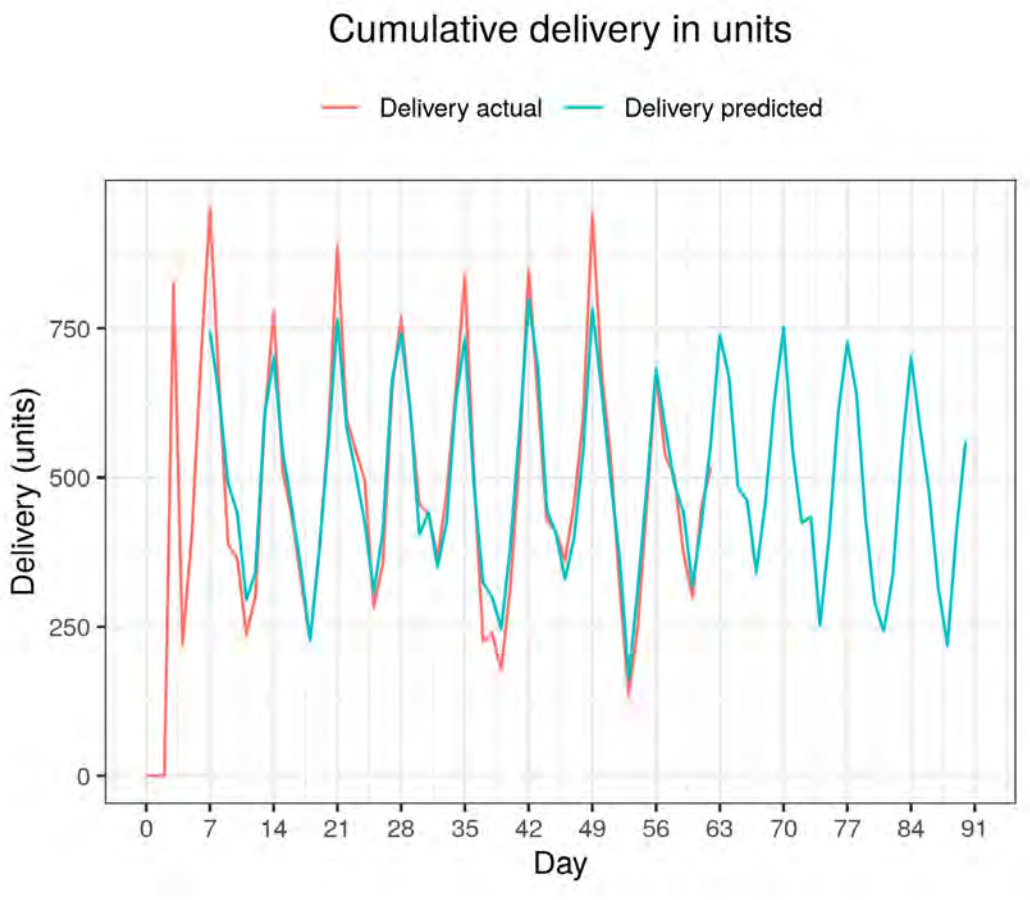


Fig. 19: Predicted and actual cumulative deliveries across all products for Store 3 using the last n days a predictors.

Fig. 20 shows the importance of features in predicting deliveries. This plot looks similar to those we have seen before after we learnt the models, with the moving average of 3 days having the most weight in predicting deliveries.

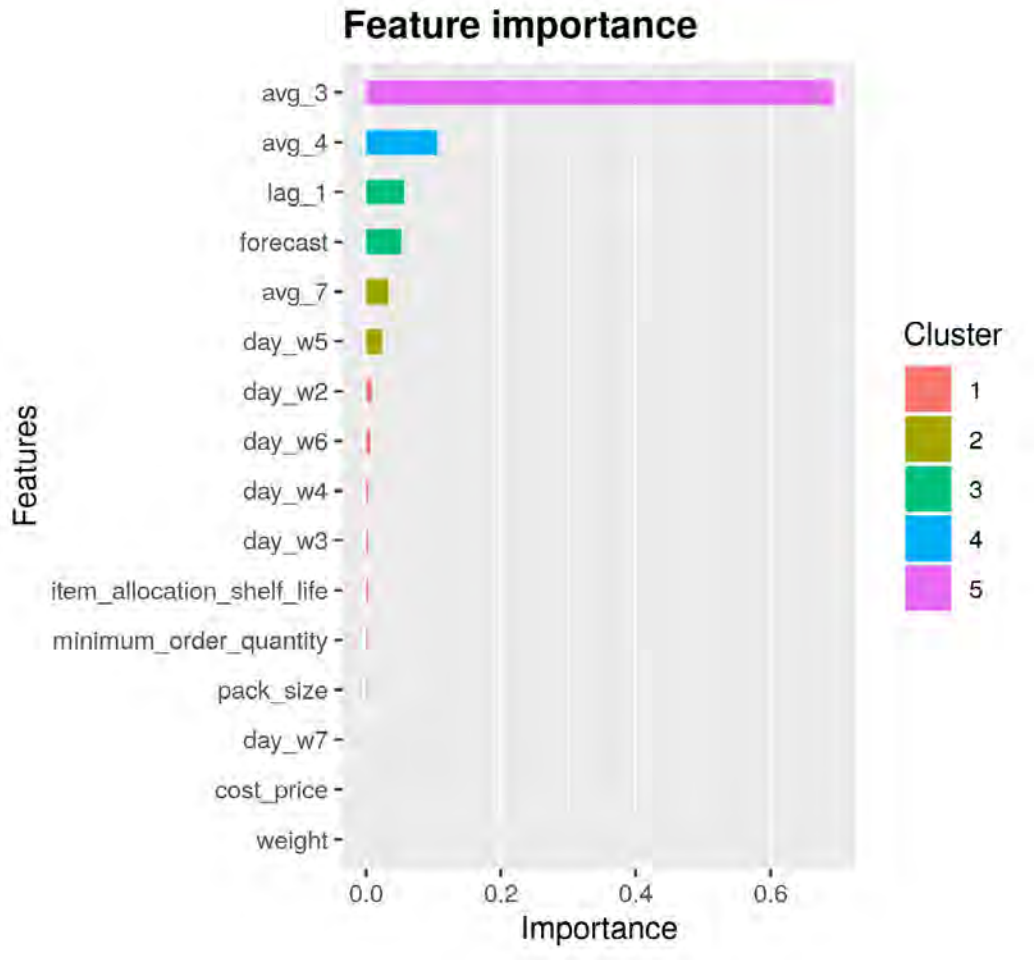


Fig. 20: Importance of each feature in making predictions (measured in gain) for the model using moving averages over the next n days as predictors. This model was trained Store 3 data. Methods are clustered by their importance.

Now that we have a model based on Store 3, we can use it to predict deliveries for Stores 1 and 2 over the 53 days of historical data (63 minus

the first 7 days). Once we have these deliveries we can calculate the waste Stores 1 and 2 would have produced and compare them to their actual waste. To make this comparison we need to make four assumptions:

i If, for a given day, the actual sales were higher than deliveries from the depot and the stock in store predicted by the model, then we would cap the sales at sum of deliveries from depot and stock on the day.

ii We assume that all products arrive to stores completely fresh, i.e. that they do not spend any time at the depot.

iii The actual deliveries were the same to model deliveries for the first 7 days.

iv The products that are closest to expiry are sold first.

We assume (i) to simplify the scenario. Assumption (ii) is a simplification of the challenge that assumes that all products can stay at the depot on any day. However, the products can only stay up to 20% of their shelf life in the depot, and given that the median shelf life of the products is 8 days, that would mean that most products would need to leave the depot after 1 or 2 days at a maximum. Furthermore, if we recalculate waste given the actual sales and actual deliveries assuming (ii), the total waste for Stores 1 and 2 is 28,652 units, which is only 5% less than than the total waste actually produced at these two stores (30,269 units). Thus, although assumption (ii) is a simplification, it does not make the waste comparisons completely uninformative. We assume (iii) because our model does not give any predictions for the first 7 days.

Fig. 21 shows actual and predicted waste for Stores 1 and 2 for the first 63 days. We can see the waste predicted by the model is noticeably lower than the actual waste after the first 7 days. This suggest that the model learnt from Store 3 data would be able to reduce waste when applied to the other two stores.

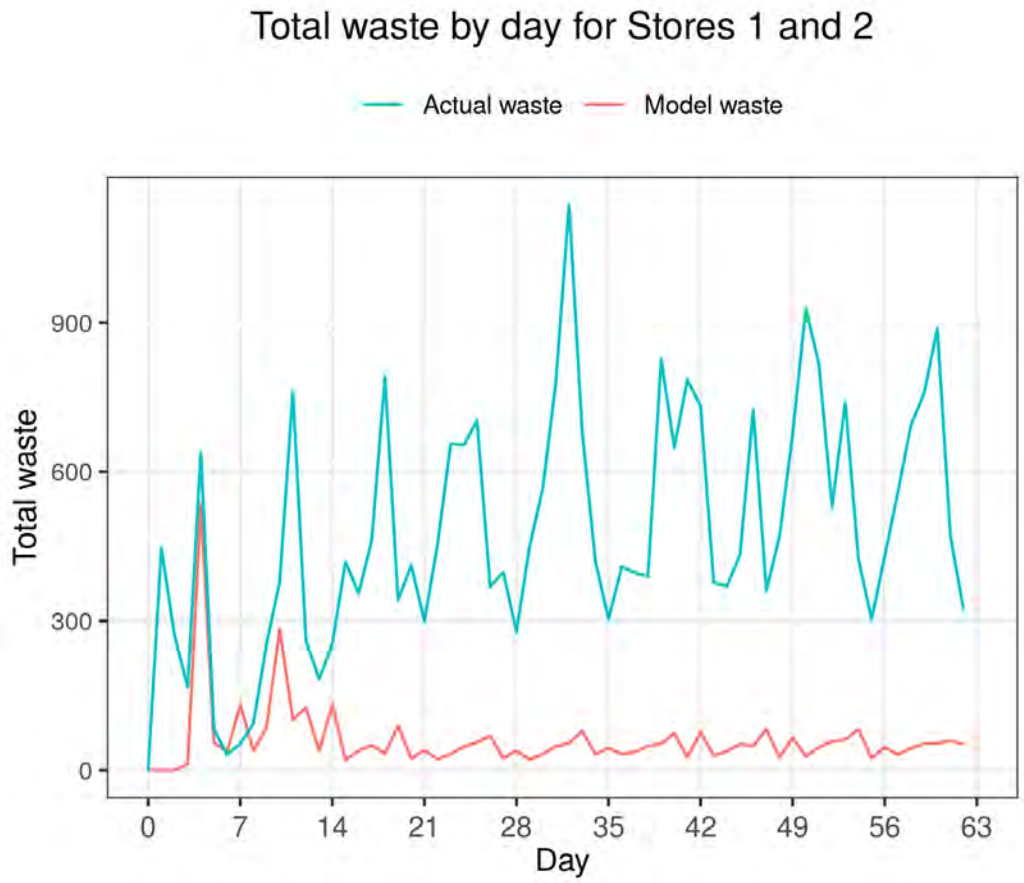


Fig. 21: Actual and model waste for Stores 1 and 2 for the first 63 days of data.

Because we have capped the actual sales (as per assumption (i)), we expected the sale to be lower than the actual sales. Fig. 22 shows actual and predicted sales. We can see the model has fewer sales than in the store data.

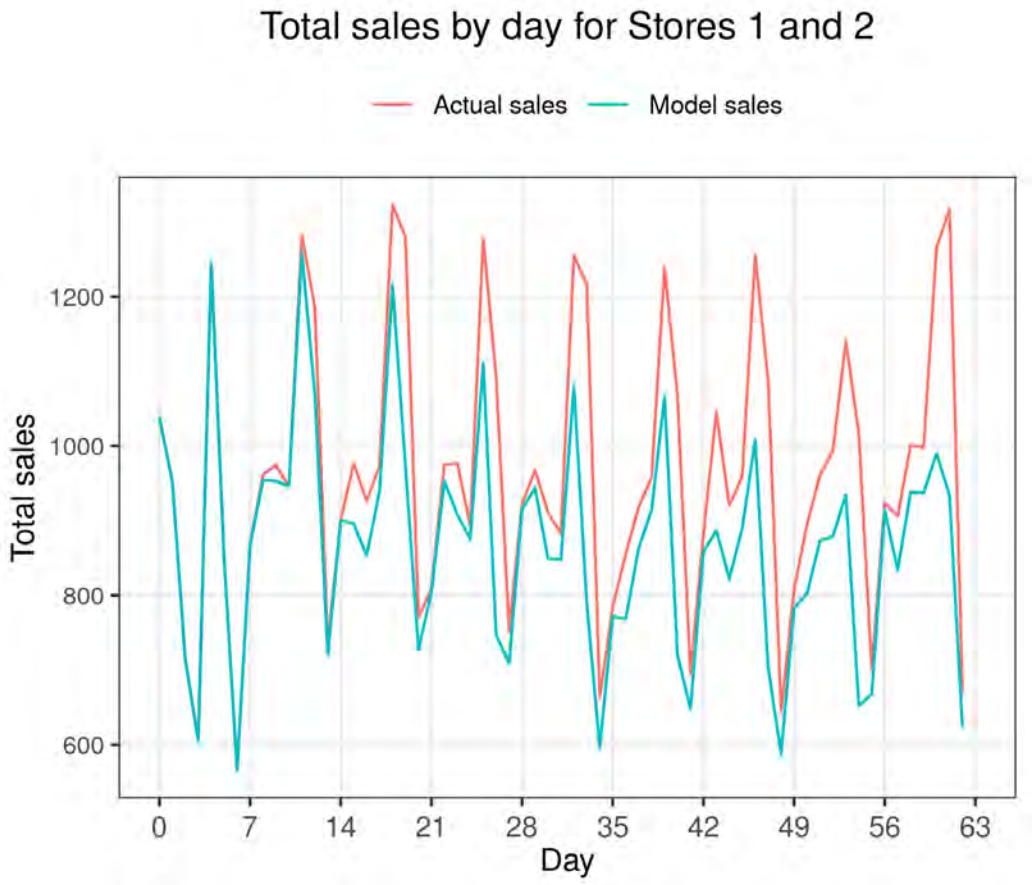


Fig. 22: Actual and model sales for Stores 1 and 2 for the first 63 days of data.

Fig. 23 shows the total model waste and sales as a percentage of total actual waste and sales for Stores 1 and 2. This shows the reduction in sales was not as great as the reduction in waste. Namely, model sales were only 10% lower than the actual sales. Waste, on the other hand, was more than 85% lower than the actual waste. The model thus shows significant improvement in waste management for a relatively small cost in sales.

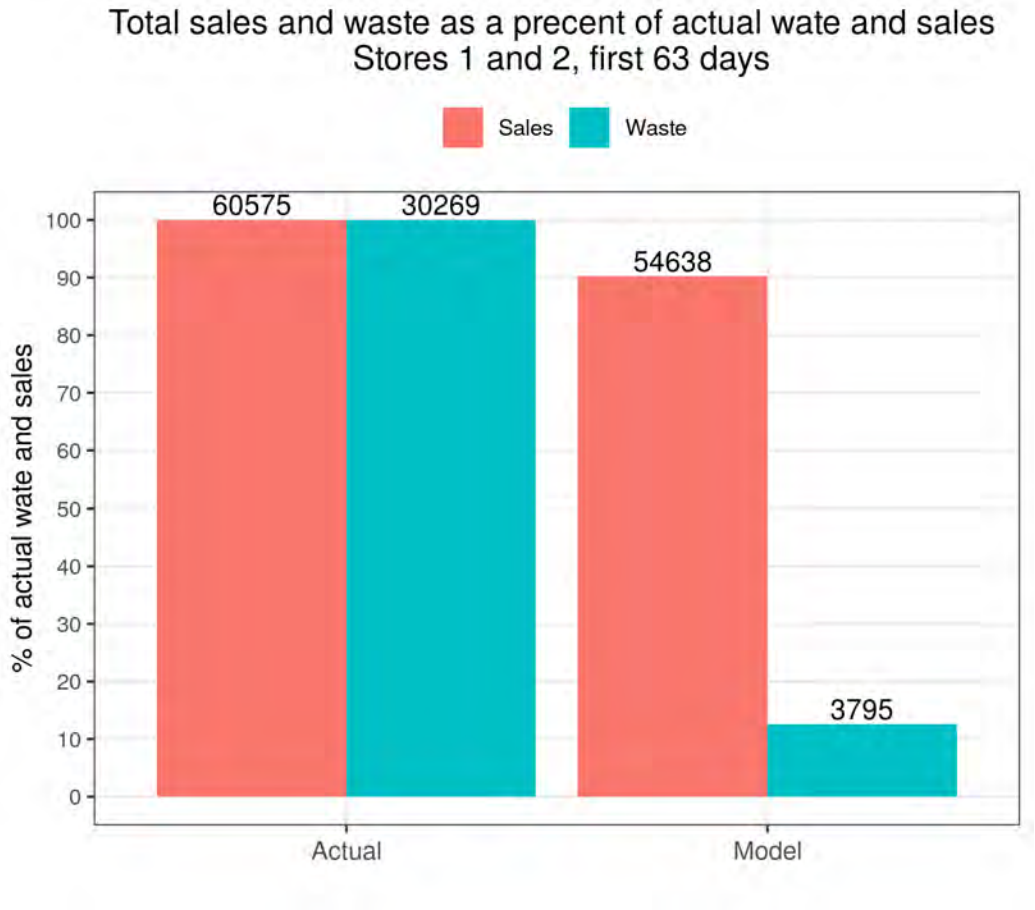


Fig. 23: Total model waste and sales as a percentage of total actual waste and sales for Stores 1 and 2. The numbers above the bars are total waste and sales in units.

There are, however, a few limitations with this model and its waste calculations. First, the waste and sale calculations assume (i) to (iv). Although (i) seems like a natural assumption, further research should aim to relax (ii). We hypothesise that relaxing (ii) would lead to higher waste production, although we do not expect it to lead to a large increase (above we have seen that recalculating waste using (ii) lead to only a 5% difference in total waste). Relaxing (ii) would also lead to considering products staying at the depot as well as other constraints, such as minimum order quantity. We expect assumption (iii) does not make a big

impact on actual waste calculations given the data stretches long enough into the past. Assumption (iv), however, could be further relaxed in a manner that is informed by data. In general, however, we expect that relaxing (iv) would lead to more waste, but the exact amounts will depend on how it is relaxed.

We have noted that as Store 3 produces less waste without lower deliveries and sales, it could be used as a *role model* for other stores. We thus trained a model on Store 3 data in the hope of learning delivery schedules that would reduce waste in other stores as well. However, constraints such as minimum order quantity could imply that a higher amount of product would need to be ordered to satisfy (a perhaps low) demand. This could further mean that products from the depot have to be shipped to the stores (as they cannot stay at the depot longer that 20% of their shelf life) and therefore that Stores 1 and 2 receive more than they require. Although our model does not separate between this situation and the situation where Stores 1 and 2 order more due to a sub-optimal supply chain, we can use this model to help distinguish between these scenarios.

4 Mixed-Integer Linear Programming (MILP)

The problem specified by Morrisons fits well into the MILP framework. Specifically, we are interested in *minimising* the product *wastage* and *delivery mileage* subject to the following constraints:

1. For the 28-day period, the accuracy of the sales forecast is in line with the historical data provided.
2. The supply of products to stores should be at minimum equal to expected demand.
3. Products have a minimum order quantity from manufacturing plants and so can only be ordered into the depot in specific quantities. Existing minimum order quantities are provided, but this constraint is encouraged to be relaxed/altered to see if easier or better optimal solutions can be found.
4. The lead time from manufacturing to distribution is assumed to be 3 days, so an order placed on day 1 is received in the depot on day 4.
5. The cost of delivering goods from the manufacturer to distribution depot is assumed to be included in the cost of goods.
6. The cost of processing one case in the depot is 10p. This represents the cost of receiving the case and processing the case for dispatch to the store.
7. One case contains multiple units of the same product. A pallet can contain multiple cases, and does not have to contain all the same product.
8. The cost of shipping one pallet from the distribution depot to a store is 30p per mile.
9. One pallet can hold 1.92 m^3 of volume and can hold no more than 1,500kg. The maximum load a shipment can carry is 26 pallets.
10. Each product has a minimum life code. For example, a product with a minimum life code of 10 days will be wasted if it is not sold within 10 days of it arriving into the depot.

4.1 Variables

We define the following integer variables. Notice that some quantities are measured in units of product, while others are measured in cases, which consists of multiple units of a single product.

- $u_{p,t}$ is the number of cases of product p that arrive at the depot at time t. Note that we assume that all units in these cases are *born* at time $\tau = t$, since we assume that products are born when they arrive at the depot.
- $x_{p,t,\tau}$ is the number of cases of product p that arrived in the depot at time τ that are still in the depot at time t.
- $v_{p,s,t,\tau}$ is the number of cases of product p that arrived in the depot at time τ that arrive at store s at time t. Note that we are implicitly assuming here that the time spent in transit between the depot and any store is zero.
- $y_{p,s,t,\tau}$ is the number of units of product p that arrived in the depot at time τ that are in store s at time t.
- $n_{s,t}$ is the number of pallets (consisting of cases, that can be of the same or different products) that leave the depot and arrive at store s at time t.
- $q^F_{p,s,t,\tau}$ is the number of units of product p that arrived at the depot at time τ that are forecasted to sell at time t in store s.

4.2 Parameters

We next define fixed parameters of the model. These are all given in the dataset.

- $Q^{\mathsf{F}}_{p,s,t}$ is the forecasted number of sales of units of product p at store s at time t. We emphasise that, while we know how many units of each product are sold in each store at each time, such that $Q^{\mathsf{F}}_{p,s,t}$ is given, we do not know the timestamps of those products sold. It is for this reason that we treat $q^{\mathsf{F}}_{p,s,t,\tau}$ as an unknown variable that we solve for, and we constrain it to satisfy $\sum_\tau q^{\mathsf{F}}_{p,s,t,\tau} = Q^{\mathsf{F}}_{p,s,t}$. Note that, one

desirable effect of this formulation is that our algorithm will devise an optimal order in which to sell products in terms of their expiry dates.

- T_p is the number of timesteps that product p lives. That is, the number of timesteps between when the product enters the depot and when it expires, so that product p expires instantaneously at time $t = \tau + T_p$.
- C_p is the production cost of a case of product p.
- C^{pro} is the cost of processing one case (10p) of any product. Note that we assume that this is a cost associated with readying cases for dispatch, and as such it is charged for cases leaving the depot (and not all of those in the depot, or those arriving).
- C^{tra} is the cost for a pallet to travel one mile (30p).
- V^{pal} is a pallet's volume capacity (1.92m^3).
- M^{pal} is a pallet's mass capacity (1,500kg).
- V_p is the volume of a case of product p.
- M_p is the mass of a case of product p.
- N_p is the number of units of product p that fit in a case.
- D_s is the distance from the depot to store s.

We also define one parameter that is not given in the data, which we choose:

- $C^{\text{hol}}_{p,t,\tau}$ is the cost of holding product p that arrived in the depot at time τ at time t. We will see that we require this parameter to impose that the algorithm sells the oldest products first, which models true store-behaviours, such as showing oldest products at the front of shelves, and discounting older products. Sensible examples of this function are those that are undefined in the regions $t < \tau$ and $t > \tau + T_p$, since product p does not exist here. In the region $\tau \leq t \leq \tau + T_p$, we choose a monotonically increasing function of t, whose gradient increases with t, and that is minimised with 0 at $t = \tau$, and is maximised at $t = \tau + T_p$. Note that, although we did not consider this in our implementation, strictly we should enforce that the integral under this function is the cost of the product to the algorithm, since we would like maximum total penalisation for an

unsold unit of product to be that product's cost. In the results, we chose the quadratic function that reaches $0.4C_p$ at $t = \tau + T_p$.

4.3 Objective function

We next define an objective function to be minimised. We define

$$
\begin{aligned}
f(u_{p,t}, x_{p,t,\tau}, v_{p,s,t,\tau}, y_{p,s,t,\tau}, n_{s,t}, q^{\mathsf{F}}_{p,s,t,\tau}) = & \sum_{s}\sum_{p}\sum_{t\geq T_p}\left(\frac{C_p}{N_p}\right) y_{p,s,t,t-T_p} \\
& + \sum_{s}\sum_{p}\sum_{\tau > T-T_p}\sum_{t\geq\tau}\left(\frac{C_p}{N_p}\right) y_{p,s,t,\tau} \\
& + \sum_{s}\sum_{p}\sum_{\tau}\sum_{\tau\leq t\leq\tau+T_p} C^{\mathsf{hol}}_{p,t,\tau} y_{p,s,t,\tau} \\
& + C^{\mathsf{tra}}\sum_{s}\sum_{t} D_s n_{s,t} \\
& + C^{\mathsf{pro}}\sum_{p}\sum_{s}\sum_{t}\sum_{\tau} v_{p,s,t,\tau}
\end{aligned}
\tag{9}
$$

We now briefly discuss each term in this objective function:

- The first term is the cost of waste due to *expiry* in the time period $0 \leq t \leq T$ from all products across all stores. When $\tau = t - T_p$ and $t \geq T_p$, then $y_{p,s,t,\tau}$ is the number of products p at store s that are past their expiry date at time t. Note that C_p/N_p is the production cost of a unit of product p.
- The second term is the cost of waste due to all products that remain across all stores *after* the time period of interest $0 \leq t \leq T$ since. When $\tau > T - T_p$ (or equivalently $\tau + T_p > T$) and $t > T$, then $y_{p,s,t,\tau}$ is the number of products p at store s after time T.
- The third term is the cost of holding stock across all products that are due to expire at time T_p. At times $\tau \leq t \leq \tau + T_p$, the term $y_{p,s,t,\tau}$ is the number of products p at store s.
- The fourth term is the cost of moving all products to all stores since, at time t, $C^{\mathsf{tra}} D_s n_{s,t}$ is the cost of moving the required products to store s.

- The fifth term is the cost of processing all products at the depot before they are moved to all stores. At time t, the term $C_{\text{pro}} v_{p,s,t,\tau}$ is the cost of processing a case of product p that arrived at the depot at time τ before it is shipped to store s.

4.4 Constraints

We will minimise this cost function subject to the following constraints. Note that $\tau \geq 0$ is always true, since we assume that products that are in the depot before the initial time are delivered to the depot at the initial time.

- The variables are integers that are greater than zero, which are

$$y_{p,s,t,\tau}, v_{p,s,t,\tau}, x_{p,t,\tau}, n_{s,t}, u_{p,t,\tau} \in \mathbb{N}, \tag{10}$$

$$q^F_{p,s,t,\tau}, y_{p,s,t,\tau}, v_{p,s,t,\tau}, x_{p,t,\tau}, n_{s,t}, u_{p,t,\tau} \geq 0. \quad \forall p, s, \tau, t \in [\tau, ..., \tau + T_p]. \tag{11}$$

- The sum of products of type p delivered to the depot on all days, is the total number of products of type p, which is

$$\sum_{\tau} q^F_{p,s,t,\tau} = Q^F_{p,s,t} \quad \forall\, p, s, \tau, t \in [\tau, \tau + 1, ..., \tau + T_p]. \tag{12}$$

- Products are conserved at the depot. For each product, for each order date τ, the amount of stock at depot is what was there yesterday, plus what comes in, minus what goes out to all stores. That is,

$$x_{p,t,\tau} = x_{p,t-1,\tau} - \sum_{s} v_{p,s,t,\tau} + \delta_{t,\tau} u_{p,\tau}, \quad t \geq 1. \tag{13}$$

- Products cannot last in store for more than 20% of their lifetime (T_p). So, defining $\delta_p = [0.2T_p]$, we have,

$$x_{p,t,\tau} = 0 \quad \text{for} \quad t > \tau + \delta_p. \tag{14}$$

  Notice that this allows us to reduce the dimension of the variable space.

- We know the number of products in the depot and the store initially, so that

$$x_{p,0,0}, \quad y_{p,s,0,0} \tag{15}$$

are known quantities for every p and s. Notice that, combining these equations allows us to write

$$x_{p,t,\tau} = x_{p,t-1,\tau} - \sum_s v_{p,s,t,\tau} + \delta_{t,\tau} u_{p,\tau} \quad \max\{1,\tau\} \leq t \leq \tau + \delta_p \tag{16}$$

$$0 = x_{p,t-1,\tau} - \sum_s v_{p,s,t,\tau} + \delta_{t,\tau} u_{p,\tau} \quad t = \tau + \delta_p + 1. \tag{17}$$

- Products are conserved at each store. That is, for each product, for each order date τ, the amount of stock at each store is what was there yesterday, plus what comes in, minus what goes out due to forecasted sales, so that

$$y_{p,s,t,\tau} = y_{p,s,t-1,\tau} + \delta_{t \leq \tau + \delta_p} N_p v_{p,s,t,\tau} - q^F_{p,s,t,\tau} \quad \tau + 1 \leq t \leq \tau + T_p. \tag{18}$$

- The volume of the stock being delivered must be less than the maximum volume that the whole number of pallets that are required can hold, such that

$$n_{s,t} V^{\text{pal}} \geq \sum_p \sum_{t - \delta_p \leq \tau \leq t} V_p v_{p,t,\tau}. \tag{19}$$

- The mass of the stock being delivered must be less than the maximum mass that the whole number of pallets that are required can hold, such that

$$n_{s,t} M^{\text{pal}} \geq \sum_p \sum_{t - \delta_p \leq \tau \leq t} M_p v_{p,t,\tau}. \tag{20}$$

4.5 Complexity

We consider the computational size of the problem that we have developed. We have four variables that are indexed by p, s, t, τ namely

$x_{p,s,t,\tau}, y_{p,s,t,\tau}, v_{p,s,t,\tau}$ and $q^F_{p,s,t,\tau}$. If the number of days considered is T, the number of products is P, and the number of stores is S, with δ_p and T_p as before, then the total number of variables in the problem is:

$$2(\delta_p + 2T_p)PST + PS + ST = 253,134 \tag{21}$$

This is very large for an MILP problem, and may result in a large computational expense that the problem takes an infeasible amount of time to solve. With the limitations imposed on computational resources and time during the challenge, the MILP was subject to the following simplifications:

- We solved for each product separately. Notice that this means that we will not achieve a globally (across all products) optimal solution. Our data analysis, however, suggests that products are largely decoupled. Notice that one obvious extension is to solve for product groups. One might consider product groups that arise from our clustering analysis.
- We do not solve each particular product problem to optimality. Instead, we allow our algorithm to exit its search when our solution is *good enough*. We note that, with more time, each product problem is feasible.
- We note that solution time is as little as a second when we relax our problem to a linear problem, by which we mean that we no longer constrain our variables as integers. We note that this type of relaxation should be avoided though, since much of the complexity of the problem lies in the fact that products must be shipped and stocked whole (as integers), and relaxing this constraint solves a rather different problem.
- Furthermore, we supplied the searching algorithm with artificial but sensible constraints, in order to limit the space over which it searches for an optimal solution. In particular, we choose

  $$Y_p \leq \alpha \max_{s,t}(Q^F_{p,s,t}), \tag{22}$$

  where $\alpha > 1$ is some real number, so that the stock is upper bounded linearly by the forecast, and never exceeds the forecast dramatically.

Other sensible bounds emerge naturally from this store stock bound. In particular, we impose

$$v_p \leq \frac{Y_p}{N_p}, \tag{23}$$

$$q^{\mathsf{F}}_{p,s,t,\tau} \leq Q^{\mathsf{F}}_{p,s,t}, \tag{24}$$

$$x_{p,t,\tau} \leq \delta_p S \frac{Y_p}{N_p}, \tag{25}$$

$$u_{p,t} \leq \delta_p S \frac{Y_p}{N_p}, \tag{26}$$

$$n_{s,t} \leq \max_p \left\{ \frac{Y_p}{N_p} \frac{V_p}{V^{\mathsf{pal}}}, \frac{Y_p}{N_p} \frac{M_p}{M^{\mathsf{pal}}} \right\}. \tag{27}$$

We note that the bound on $n_{s,t}$ makes sense since, for example, $Y_p V_p / N_p V^{\mathsf{pal}}$ is the upper bound on the volume of a case of product divided by the pallet's volumetric capacity, which indeed gives the maximum required number of pallets.

4.6 Results

To obtain results in reasonable computational time, we solve the problem for each product individually, over all the stores and also considering the depot. We stop the algorithm early, after a fixed time of 30 minutes. Note that it would be more appropriate to stop when the optimality gap drops below a specific threshold, but unfortunately this is not supported by the *mip* package in Python, which is used as a solver. Nevertheless, the time criterion generally gives sufficiently small optimality gaps.

From Fig. 24 we can observe that it is generally optimal to deliver products to stores in quantities to be sold over one lifetime period. However, this does not translate as easily to the depot, since a depot has to deliver products to multiple stores. The minimum order quantity at the depot will strongly influence this. In this model, we have not included the minimum order quantity for simplicity. Furthermore, note that the model also tells us the optimal order of selling the products based on their expiration dates, and this information is found in $q^F_{s,p,t,\tau}$.

Fig. 25 gives a comparison between the waste in the given historical data versus the waste given by the solution to our model. It is clear that the

proposed solution reduces waste, albeit with an exception in the case of product 10 for Store 2. This is because we could not solve the corresponding sub-problem sufficiently close to optimality in the allocated time. Note that due to time constraints and shared computational resources we could not compute the MILP solution for all the products. An alternative approach to MILP that can be less computationally expensive is Reinforcement Learning. Assuming the action-value function is learned, reinforcement learning should be much faster since optimal decisions are made on the fly at each time step. However, the big challenge is in learning the action-value function. Reinforcement learning is further discussed in Section 6. We next use ideas derived from this model to formulate a simpler and more computationally tractable optimisation model in Section 5.

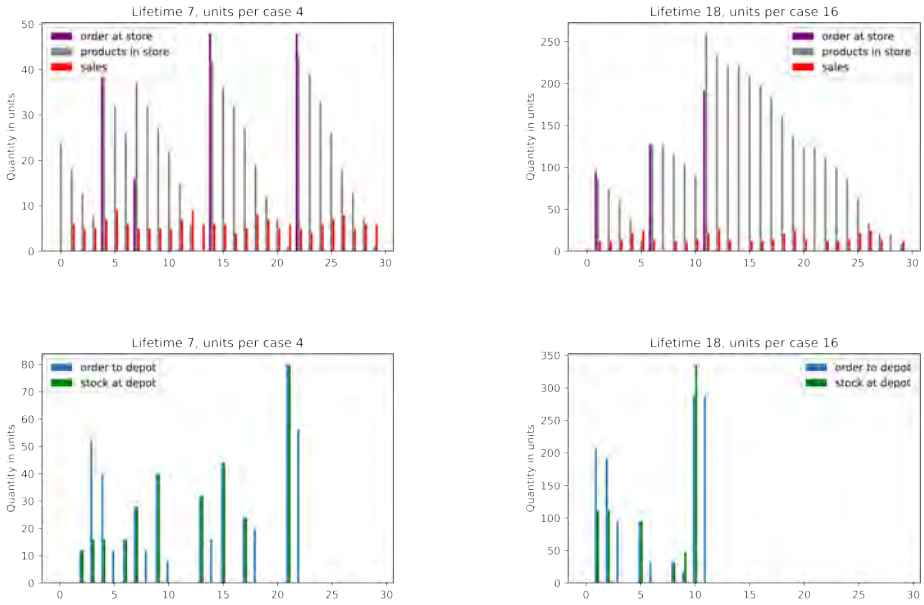


Fig. 24: Delivery, sales and stock levels at end of day at store 1 for two representative products (top), and the orders and stock at the depot (bottom)

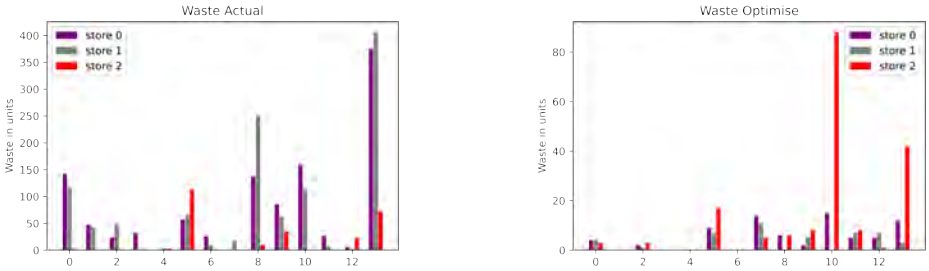


Fig. 25: Waste at stores based on actual decisions (left) vs waste at store based on MILP solution (right) for 13 products

5 One period optimisation

The integer programming approach, described in Section 4, suggests that often a near-optimal solution is to make a large delivery that will last roughly one shelf-life period. This is similar to the *news vendor* problem [1, 9]. In the news vendor problem, a total of v newspapers are ordered at cost price c and sold at price p. Each day the demand d is a random variable with a cumulative distribution function F. All newspapers remaining at the end of the day are wasted. The expected profit is therefore

$$\bar{P} = E\left[p\min(v,d)\right] - cv, \tag{28}$$

where E is the expectation operator. $\bar{P}$ compares the expected sales income with the cost of ordering v papers. The optimal solution is to order:

$$v^* = F^{-1}\left(\frac{p-c}{p}\right) \tag{29}$$

Note that the optimal solution depends on the cumulative distribution of the stochastic demand, rather than the expected demand, since at most v papers can be sold.

Let us now formulate our problem in a similar way to the news vendor problem. Consider a single product in a single store. Thus we are relaxing the minimum order constraints required at the depot and we are decoupling different products by assuming they will be packed in different pallets. Consequently, we expect this approach to over estimate transport costs. Our strategy is to deliver enough stock to last until the end of the

product's shelf life (one period), after which any remaining stock is wasted and a new delivery is made for the next period. Thus the problem is similar to a series of single period news vendor problems. However, in our case we do not know the distribution of demand, rather we know the forecast demand. We also do not know the product prices and there are additional costs due to processing and transportation. We define the following variables:

- v, the number of items delivered in a period, which we assume is a multiple of cases;
- q, the forecast number of items that will be sold;
- d, the actual number of items sold (demand).

Next, we define the following parameters:

- C, the cost of producing one case;
- C^{p}, the cost of processing one case;
- C^{t}, the cost of transporting one pallet per mile;
- D_{s}, the distance from the store to the depot in miles;
- N_{i}, the number of items per case;
- N_{n}, the number of pallets per case.

Following the approach of the news vendor problem, we might arrive at the *cost* function

$$c(v,d) = \frac{C}{N_{\mathrm{i}}}\left[v - \min(v,d)\right] + C^{\mathrm{p}}\frac{v}{N_{\mathrm{i}}} + C^{\mathrm{t}}D_{\mathrm{s}}\mathrm{ceil}\left(\frac{v}{N_{\mathrm{i}}N_{\mathrm{n}}}\right). \tag{30}$$

However, the optimal order quantity that minimises this cost function is always $v = 0$ (i.e. no stock should be ordered), so we need to reconsider how we approach this problem. Instead, we consider penalising unmet demand with the cost function

$$c(v,d) = \frac{C}{N_{\mathrm{i}}}\left|v - q\right| + C^{\mathrm{p}}\frac{v}{N_{\mathrm{i}}} + C^{\mathrm{t}}D_{\mathrm{s}}\mathrm{ceil}\left(\frac{v}{N_{\mathrm{i}}N_{\mathrm{n}}}\right). \tag{31}$$

The first term in the cost function ($\frac{C}{N_{\mathrm{i}}}\left|v - Q\right|$) is the cost of producing stock above or below the forecast demand. Here we are assuming that the cost

of producing one unit is the same as the store price, since we were only given production costs. Thus ordering one too many cases costs the same as ordering one too few, which would result in lost potential revenue. The second term ($C^{\mathrm{p}}\frac{v}{N_{\mathrm{i}}}$) describes the processing cost, where $\frac{v}{N_{\mathrm{i}}}$ is the number of cases processed. The third term ($C^{\mathrm{t}}D_{\mathrm{s}}\mathrm{ceil}\left(\frac{v}{N_{\mathrm{i}}N_{\mathrm{n}}}\right)$) is the transport cost, where $\mathrm{ceil}\left(\frac{v}{N_{\mathrm{i}}N_{\mathrm{n}}}\right)$ is the number of pallets transported.

The optimal solution to the above problem is

$$v^* = N_{\mathrm{i}}\mathrm{round}\left(\frac{v}{N_{\mathrm{i}}}\right), \tag{32}$$

provided that

$$C > C^{\mathrm{p}} + C^{\mathrm{t}}D_{\mathrm{s}}. \tag{33}$$

Thus the optimal solution is to deliver the closest whole number of cases to the forecast. The cost condition (eq. (33)) means that the production cost (which in this case is the same as the price of the item) should be larger than the combined processing and transportation costs. If this is not the case, then it may be possible to reduce the cost by reducing the number of cases processed (e.g. if the processing cost is high) or the number of pallets transported (if the transportation cost is high). Since the condition in Eq. (33) is not always met in the data, we compare three costs: rounding up the number of cases, rounding down the number of cases, and rounding down the number of pallets. We then choose the order corresponding to the minimum cost.

For each product in each store in the data, we identified its period, identified or computed the parameters from the data given, and summed up the store sales, store forecast, waste at store and delivery from depot for each period. We set the periods to start from the first day in the data. In reality, stock that hasn't expired is transferred from one day to the next, so when we aggregate the data over each period, some stock can be passed from one period to the next. Thus the total amount of stock delivered during the period does not correspond to what is used. From conservation of stock, the amount of stock used in a single period is equal to the sum of the total items sold and total items wasted.

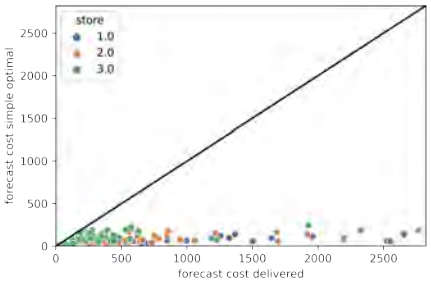
With the data given and the cost function in Eq. (31) there are two choices for v and two for d. The number of items delivered to the store (denoted

v) can be our *optimal one-period estimate* v^* (see eq. (32)) based on the forecast demand. However, we can also compare this to the *actual stock used* in the store, i.e. sales and waste. Note that the latter quantity will not necessarily be a whole number of cases. Eq. (31) allows for this, and since in reality the deliveries are a whole number of cases, penalising the actual stock used because of this would be unfair. For the number of items sold (d), we can use the actual sales figure, but we can also use the forecast sales, since this gives us a forecast cost. Once we have computed the cost for each product in each period, we compute the total cost (i.e. the sum over periods) for each stock in each store. Fig. 26 illustrates these results. In the figure, black diagonal lines correspond to equal costs on the horizontal and vertical axes, thus points above the line mean that the *vertical* axis costs more, and points below mean the *horizontal* axis costs more.

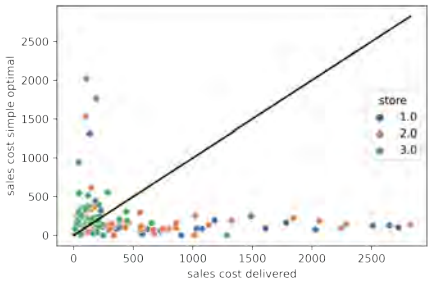
In fig. 26, the black diagonal lines correspond to equal costs on the horizontal and vertical axes. Fig. 26a illustrates the comparison of the total forecast cost between the actual stock used and the optimal one-period estimate . This suggests that the forecast total cost based on the actual stock used is always higher than that of the optimal one-period estimate.

Fig. 26b compares the total sales cost between the actual stock used and the optimal one-period estimate. This shows there are a number of products (those spread out horizontally) where the total cost of the actual stock used is higher than that of the optimal one-period estimate. However, there are also a number of products (those spread out vertically) where the reverse is true. The balance however is in favour of the optimal one-period estimate: there are 89 product/store combinations where the total cost of the optimal one-period estimate is lower than the total cost of the actual stock used, and 76 where the reverse is true.

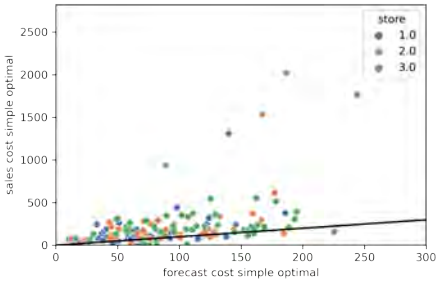
Fig. 26c compares the total forecast and sales costs for the optimal one-period estimate. The optimal one-period estimate based on the forecast cost underestimates the true total cost that results from sales. Fig. 26d compares the total forecast and sales costs for the actual stock used. In this case, while the forecast over-estimates the sales for the majority of product/store pairs (85 over-estimates vs 80 under-estimates), the over-estimation of products is generally relatively small.



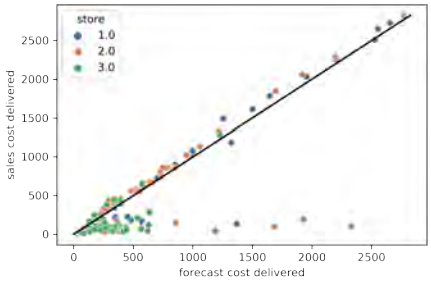
(a) Comparison of the total forecast cost between the actual stock used and the optimal one-period estimate.



(b) Comparison of the total actual sales cost between the actual stock used and the optimal one-period estimate.



(c) Comparison between the total forecast and sales costs for the optimal one-period estimate.



(d) Comparison between the total forecast and sales costs for the actual stock used.

Fig. 26: Comparisons of costs where black diagonal lines correspond to equal costs on the horizontal and vertical axes.

It is interesting to note that the optimal one-period estimate outperforms the actual stock used. One possible reason is that we have not considered the order size constraints at the depot, which may force deliveries that are too large resulting in large amounts of waste. We might also consider the effect of treating products independently; i.e. we have not allowed for pallets to contain more than one type of stock. Fig. 27 shows a histogram of the number of pallets used per period by each of the product/store pairs.

This shows the majority of products use less than one pallet per period. Consequently, if the transport costs are large compared to the production and processing costs, it is possible that not delivering any stock is optimal; e.g. if the lost revenue of not delivering stock is less than the transport cost of a whole pallet. This issue could potentially be resolved by including information about product price.

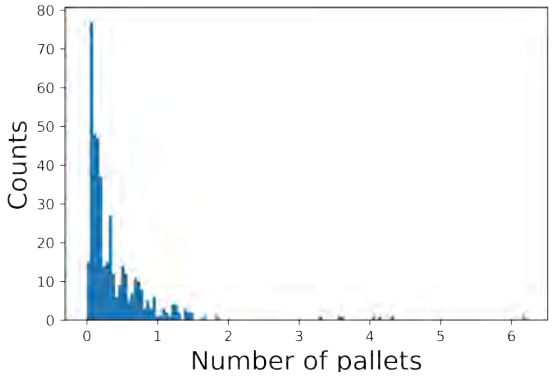


Fig. 27: Histogram of the number of pallets used per period across product/store pairs.

6 Reinforcement learning

Due to time constraints, the Reinforcement Learning (RL) solution was not computed for the supply chain optimisation challenge. The following provides the RL formulation which could be subject to future work, and is likely to scale much better computationally with more products/stores than other solutions. Note that further consideration on the correct

implementation would probably be necessary in addition to the following formulation provided.

6.1 Overview of the model

In a general Reinforcement Learning (RL) setting, at time t, an agent observes the state of an environment. It then chooses an action at time t to try to maximise the reward that it is given at time $t+1$. The action chosen at time t dictates the new state of the environment at $t+1$. The process then repeats. The agent gradually learns which action to choose to maximise its reward.

The agent can observe the depot, store, and demand forecast in the environment. The agent can take actions to control the order of an amount of stock from the manufacturer to the depot, and the order of an amount of stock from the depot to the stores. The agent will execute orders to attempt to maximise the reward, which we interpret as monetary profit. Next, we make this idea exact by defining the necessary notation.

6.2 Parameters and indices

First define the following parameters:

- T is the timestamp of the final day, so that $T+1$ is the number of times.
- S is the number of stores, $S = 3$.
- P is the number of products, $P = 55$.
- T_p is the lifetime of product p.

The following model parameters will be important when defining the reward:

- C_p^{pro} is the cost of processing a case of p that arrives at the depot. This can be thought of as the cost at which the agent buys a case of p from the manufacturer.

- C_p^{tra} is the cost per mile of transporting a case of p from depot to store.
- C_p^{cus} is the cost at which a customer buys a unit of p.
- D_s is the distance from the depot to store s.

In what follows, we will define variables indexed by the following indices:

- p is the product index, $1 \leq p \leq P$.
- t is the time index, $0 \leq t \leq T$.
- τ is the time-stamp index, $0 \leq \tau \leq T$. This is the time at which the product entered the depot.
- s is the store index, $1 \leq s \leq 3$.

6.3 State

We define the following variables:

- $x_{p,t}$ is the number of cases of product p in the depot at time t.
- $y_{s,p,t}$ is the number of units of p in store s at time t.
- $F_{s,p,t}$ is the forecasted demand, which is the number of units of product p that are forecasted to sell in store s at time t, measured at the end of the day.

We assume that $F_{p,t,s}$ is drawn from some known underlying Gaussian distribution $\mathcal{N}(\mu_F(s,p,T), \sigma_F)$ with known parameters μ_F and σ_F. Note that $\mu_F(s,p,T)$ is simply the forecast of sales in store s for product p at time t (assuming zero bias), and σ_F can be computed as the standard deviation of the time series from the residuals of forecast with respect to the actual sales. The distribution $\mathcal{N}(\mu_F(s,p,T), \sigma_F)$ is used to resolve the actual state transition, but is not part of the state observed by the agent. At the simplest level, one can take $\sigma_F = 0$. At each time t we consider the agent to have the future forecast values ΔT steps ahead.

To restrict the state and action space, unlike in the MILP model, we do not incorporate an explicit index τ, to resolve the expiration dates. Rather,

since the probabilistic environment is naturally incorporated via the Markov Decision Process framework in RL, we assume a probabilistic waste model of the form

$$w_{s,p,t} = U_{s,p,t} y_{s,p,t}, \tag{34}$$

where $U_{s,p,t} \sim \mathcal{U}(0, f)$ are random variables drawn from a uniform distribution in the range $[0, f]$ with $f < 1$, that can be chosen to reflect store data. The waste $w_{s,p,t}$ is then subtracted from $y_{s,p,t}$ to obtain the stock at the beginning of next day.

6.4 Action

We define the following variables:

- $u_{p,t}$ is the number of cases of product p that arrive at the depot from the manufacturer at time t.
- $v_{s,p,t,\tau}$ is the number of cases of p that arrived at the depot at τ that arrive at store s from the depot at t. Notice that travel is instantaneous, so that these cases also leave the depot at t.

The difficulty is that the action space is discrete yet infinite. Relaxing to a continuous action space would result in solving a problem equivalent to the LP relaxation of the MILP approach (a non-integer solution). This is clearly undesirable. Thus, we propose to retain the discrete and infinite nature of action space as follows. There are three types of actions:

1. Order at *depot* action: a_p^D. For each product p, order a case at the current time at the depot. This will have the form $(0, 0, 0, 0..., N_p, 0, 0, 0)$ for a product p. This vector is to be added to the depot stock levels. This action can be performed as many times as desired before going to the next time step. This action can always be performed, and thus is unconstrained.
2. Order at *store* action: $a_{s,p}^S$. For each product p, order an extra case at the current time at store s from the depot. We later describe the mechanism to ensure that stock is available in store when this action is performed. This action can be performed as many times as desired before going to the next time step.

3. Finish action decisions and jump to the next time step a_{next}. Once this action is performed, product transportation to the depot and from the depot to the stores, as well as the sales and waste generation, are resolved and thus their associated costs also. All actions between two consecutive a_{next} are therefore performed at the same time step.

The idea is that the agent has the flexibility to perform as many a^D (depot) and a^S (store) actions between two consecutive a_{next}, whilst also holding the action space fixed and limited, and guaranteeing integrality constraints.

6.5 Reward

We now build a reward function, which will calculate the reward that the agent will receive at time t. We define the following variables:

- $b_{s,p,t}$ is the number of units of product p that are bought by customers at store s at time t. We model this as

$$b_{s,p,t} = \min \left\{ F_{s,p,t}, y_{s,p,t} \right\}. \tag{35}$$

  One should interpret the second term in the set as the total number of units of p that are still sellable (not waste) across all stores at time t. This definition of $b_{s,p,t}$ is intuitive then, since if stock levels are less than demand then all stock is bought, while if demand is less than stock then the demand is exactly the quantity bought.

- $\rho_{p,t}$ is the profit due to p at t. We model this as the revenue from sale of p at t across all stores, minus the cost of transport of p from the depot to all stores at t, minus the cost of p that arrive at the depot at t. This is given as:

$$r_t = \sum_{s,p} C_p^{\text{cus}} b_{s,p,t} - C^{\text{tra}} \sum_s D_s n_{s,t} - \sum_p C_p^{\text{pro}} u_{p,t}. \tag{36}$$

  where r_t is the total reward at the end of physical time t, and $n_{s,t}$ obeys the same equations as in the MILP model, marginalised over τ. It is important to note that we do not explicitly negatively reward the agent for wastage. Rather, we reward it for making a profit and therefore ask it to learn to minimise waste to maximise profit.

Note that in the implementation we resolve each cost on-the-fly for each extra action performed if possible. For this we introduce the free volume available for each store s, that was already paid for (amount $C^{tra}D_s$). If not enough volume is available, a new one is purchased.

6.6 Constraint consideration

The main constraints to be enforced are:

- The integrality and non-negativity constraints of solution variables are enforced through the definition of the action space. All actions ensure order and delivery of a positive integer number of products.
- Enforcing that the depot stock is non-negative is more challenging. As the model stands, one could (i) penalise the agent with a large negative reward and stop the episode if negative levels are reached, or (ii) automatically perform an order to the depot to compensate for the shipment to the stores. The problem with (ii) is that the agent might use this mechanism to stop the episode early at a smaller loss, since it might not see far into the future that positive rewards are possible. Thus a large negative reward is required, but this might influence the learned value function at the previous state. On the other hand, (ii) does not have these issues but it might be more difficult for the agent to learn the meaning of shipping actions, since these now sometimes incorporate orders at depots, making them a nonlinear (piecewise linear) action.
- We enforce the minimum sales constraint via the mechanism described above in (i).

More exploration of possibilities is required to ensure the best choices and modelling of constraints, which relate state and action spaces.

6.7 Any other considerations

In principle, actions are integers $u_{p,t}$ and $v_{s,p,t}$. With the current action space form, we have $55 \times 3 + 55 + 1 = 221$ actions, which is very small. However, it is worth noting that the agent may take many actions before taking the action *next*, to get to the next time instant, and this might

complicate learning. We find this to be the main difficulty in training the agent.

Note that we could decrease the complexity by defining real variables on $[0, 1]$ that underlie $v_{p,t,\tau,s}$. In particular, note that the total possible amount of (p, τ) that can be shipped from the depot at t is $x_{p,t,\tau} + u_{p,t}$. Now, define $g_{p,t,\tau} \in [0, 1]$ as the proportion of this total stock that the agent will choose to ship. Since we need to know how much stock will be shipped to each store, define $f_{p,t,\tau,s}$ to be the proportion g that will be shipped to s, so that $\sum_s f_{p,t,\tau,s} = g_{p,t,\tau}$. Since we want variables on $[0, 1]$, define $F_{p,t,d,s}$ to be the normalisation of $f_{p,t,d,s}$, namely $F_{p,t,d,s} = f_{p,t,d,s}/g_{p,t,d}$, so that $\sum_s F_{p,t,d,s} = 1$. In this way, the agent will choose F, which will determine f and g, and so determine the stock to be shipped, without choosing an integer v. A similar normalisation of u is more tricky, since shipping from the manufacturer is unlimited, and so the agent chooses an integer for $u_{p,t}$. However, the problem with this idea is that we are essentially solving the equivalent of an LP instead of MILP, and thus neglect integrality constraints.

6.8 The Learning algorithm

We parameterise the state space by a vector (v_s) that contains all the store and depot stock levels, future forecasts up to ΔT periods ahead, the current time, and any free volume that was paid for. The action space is parameterised by a one-hot encoding vector (v_a) representing the action, with the value appended to the last entry, and 0 for the next action.

We aim to learn an ω-parametrised action-value function $Q(v_s, v_a; \omega)$. We would recommend an artificial neural network for this task, and the Q-learning algorithm for updating the network parameters ω. Further work is necessary to implement this correctly and provide RL solutions for the problem.

7 Team members

Alphabetical order by first name

Alexandru Puiu is a DPhil student in Applied Mathematics at the University of Oxford. He contributed to this by formulating the Mixed

Integer Linear Program, and wrote the implementation. Proposed required problem specific adjustments for the Reinforcement Learning perspective: state and action space and constraints. Formulated the learning algorithm and wrote the implementation for the RL approach.

Arkady Wey is a DPhil student in Applied Mathematics at University of Oxford. He is a member of the Industrially Focused Mathemtical Modelling (InFoMM) group, and the Oxford Centre for Industrial and Applied Mathematics (OCIAM) group. His research interests focus around the development of dynamical systems and agent-based models on networks. His contributions to this project include refinement of the Mixed-Integer Linear Programming model, its implementation, formulation of the Reinforcement Learning model, and implementation of its environment.

Festus Adedoyin is a lecturer at the Department of Computing and Informatics, Bournemouth University, UK, and a visiting scholar at the University of Alicante, Spain. He contributed to this project by serving as the facilitator for this challenge.

Idris Zakariyya is a PhD student in IoT Cybersecurity at Robert Gordon University. His research interest relate to Machine Learning (ML) for IoT cybersecurity, ML for data analysis, IoT intrusion detection, with focus on developing robust and efficient IoT security system. His contribution to this project include exploratory data analysis (examining actual and forecasted sales across products), products that derive store waste and statistical summary of store waste, actual and forecasted sales.

Jonathan Ward is a lecturer in applied mathematics at the University of Leeds. His research interests relate to dynamical processes on networks, agent-based modelling, and more broadly dynamical systems, data science and industrial applied mathematics. His contributions to this project included the literature review, formulation of the one period optimisation and associated data analysis.

Josie McCulloch is a postdoctoral Research Fellow in the School of Geography at University of Leeds. Her research interests are in understanding and quantifying uncertainty in agent-based models and subjective judgements. She contributed to this challenge by serving as a

co-principal investigator.

Kingsuk Jana is Teaching Assistant at Ahmedabad University in India. His research interests are Extreme Values Analysis, and Public Health Data Analysis. He is working on a project to predict extreme wildfires and has been working on predicting district-wise Covid hospitalization prediction using the Auto Regressive Integrated Moving Average Method.

Marko Tešić is a Royal Academy of Engineering UK IC postdoctoral research fellow at Birkbeck, University of London. He was a research intern at BlackRock where he applied causal Bayesian networks to model the relationships between financial factors and ESG criteria. He currently focuses on explainable AI. His contributions to this project include clustering and PCA analysis, and training and interpreting results of the gradient boosting tree model.

Maxwell Barton is a PhD student in Process data analytics at the University of Manchester. His research interests are in applying machine learning applications to industrial datasets to optimise and better automate processes. He contributed to this challenge by serving as the co-principal investigator for the challenge.

Sophie Ayling is a PhD student at the Centre for Advanced Spatial Analysis (CASA), University College London. She is interested in the application of data science and analytics to inform policy and planning. Sector wise, her work relates to water, sanitation and health policy challenges in developing countries. She is completing her PhD in Agent Based Modeling, alongside consulting for the World Bank.

Xingna Zhang is a postdoctoral researcher at Department of Public Health and Policy, University of Liverpool. She has thematic interests in many aspects of computational social science, data science and statistical methods and modelling, particularly spatial measures of mobility and population dynamics, and the geographies of social and health inequality under the state system; and methodological interests in spatial statistics and spatial econometrics. Her contributions to this project include unpicking the correlation of sales and forecasts between products, understanding the forecast error, and the training and interpreting results of the five machine learning algorithms to predict waste at store based on sale forecast.

References

[1] Kenneth J Arrow, Theodore Harris, and Jacob Marschak. “Optimal inventory policy”. In: *Econometrica: Journal of the Econometric Society* (1951), pp. 250–272.

[2] MA Bourlakis and PWH Weightman. “Introduction to the UK food supply chain”. In: *Food supply chain management* (2004), pp. 1–10.

[3] Tianqi Chen et al. “Package ‘xgboost’”. In: *R version* 90 (2019).

[4] Food Department for Environment and Rural Affairs. https://www.gov.uk/government/statistics/food-statistics-pocketbook/food-statistics-in-your-pocket-food-chain. 2018.

[5] Qinglin Duan and T Warren Liao. “A new age-based replenishment policy for supply chain inventory optimization of highly perishable products”. In: *International journal of production economics* 145.2 (2013), pp. 658–671.

[6] Mark Ferguson and Michael E Ketzenberg. “Information sharing to improve retail product freshness of perishables”. In: *Production and Operations Management* 15.1 (2006), pp. 57–73.

[7] Suresh Kumar Goyal and Bibhas Chandra Giri. “Recent trends in modeling of deteriorating inventory”. In: *European Journal of operational research* 134.1 (2001), pp. 1–16.

[8] Itir Z Karaesmen, Alan Scheller–Wolf, and Borga Deniz. “Managing perishable and aging inventories: review and future research directions”. In: *Planning production and inventories in the extended enterprise* (2011), pp. 393–436.

[9] Steven Nahmias. “Perishable inventory theory: A review”. In: *Operations research* 30.4 (1982), pp. 680–708.

[10] Yan Qin et al. “The newsvendor problem: Review and directions for future research”. In: *European Journal of Operational Research* 213.2 (2011), pp. 361–374.

[11] Fred Raafat. “Survey of literature on continuously deteriorating inventory models”. In: *Journal of the Operational Research society* 42.1 (1991), pp. 27–37.

8 Appendix

8.1 Statistical Summary

	store_sales						
	count	sum	mean	median	std	min	max
store_id							
1	3465	29310.0	8.458874	4.0	13.550930	0.0	125.0
2	3465	31265.0	9.023088	4.0	14.255263	0.0	138.0
3	3465	29096.0	8.397114	4.0	14.721271	0.0	153.0

Fig. 28: Statistical summary for actual sales across stores

	store_forecast						
	count	sum	mean	median	std	min	max
store_id							
1	3465	35546.128240	10.258623	5.424328	14.229219	0.0	120.929414
2	3465	38623.342092	11.146708	5.828317	15.565913	0.0	138.113957
3	3465	39710.743845	11.460532	5.994919	16.256821	0.0	164.053482

Fig. 29: Statistical summary for forecasted sales across stores

	waste_at_store						
	count	sum	mean	median	std	min	max
store_id							
1	3465	15699.0	4.530736	0.0	10.659123	0.0	123.0
2	3465	14570.0	4.204906	0.0	9.614187	0.0	73.0
3	3465	2397.0	0.691775	0.0	3.132321	0.0	61.0

Fig. 30: Statistical summary for waste across stores

8.2 Graphs on store 1 waste vs. sales for individual products

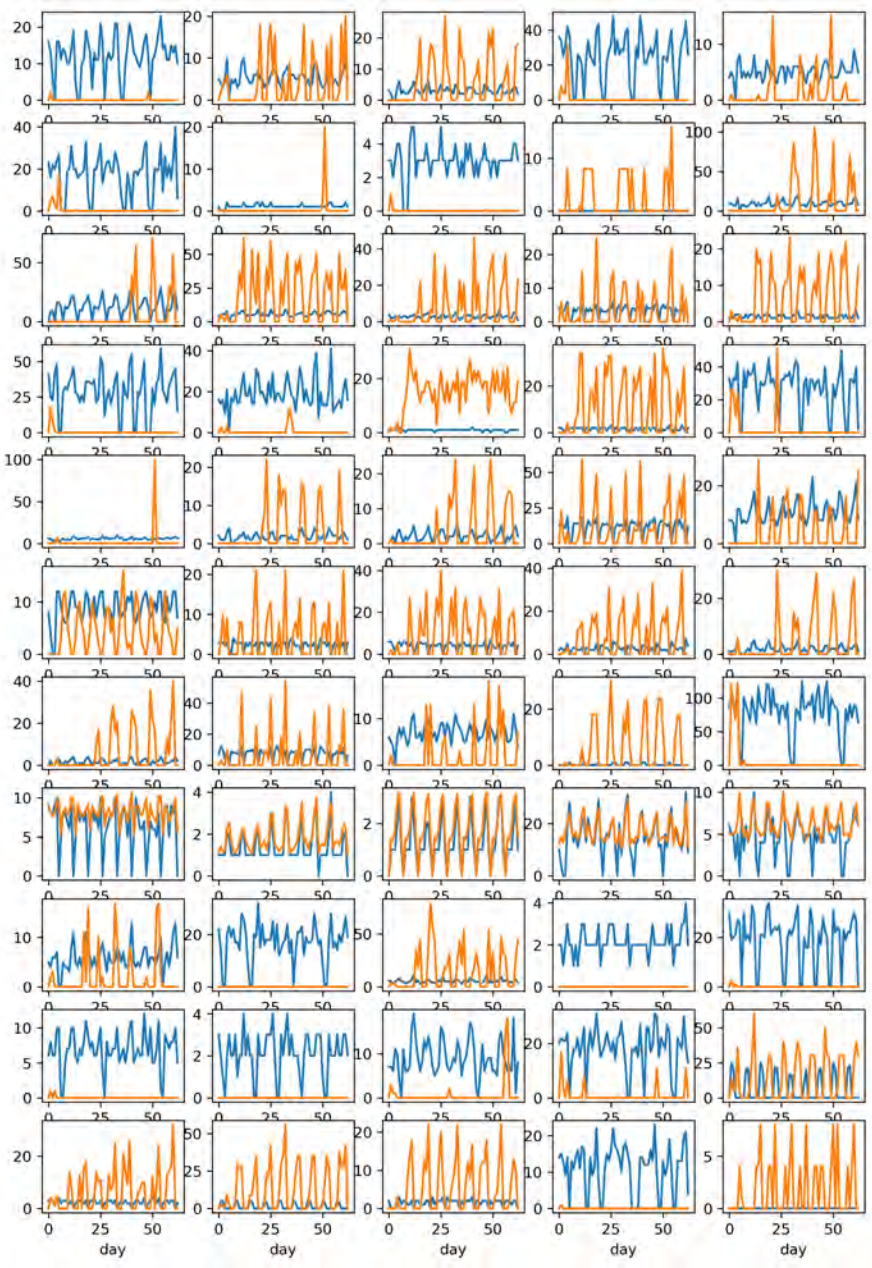


Fig. 31: Actual sales (blue) and waste (orange) distribution by all products

8.3 Full graphs of product sale covariance and residuals

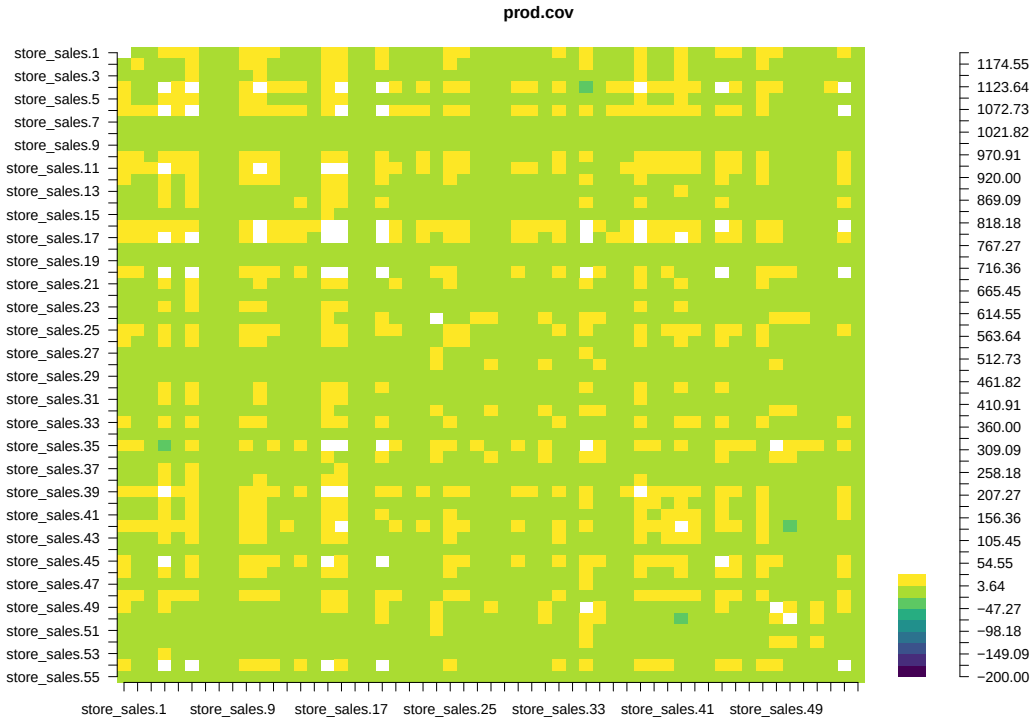


Fig. 32: Covariance of sales by all products.

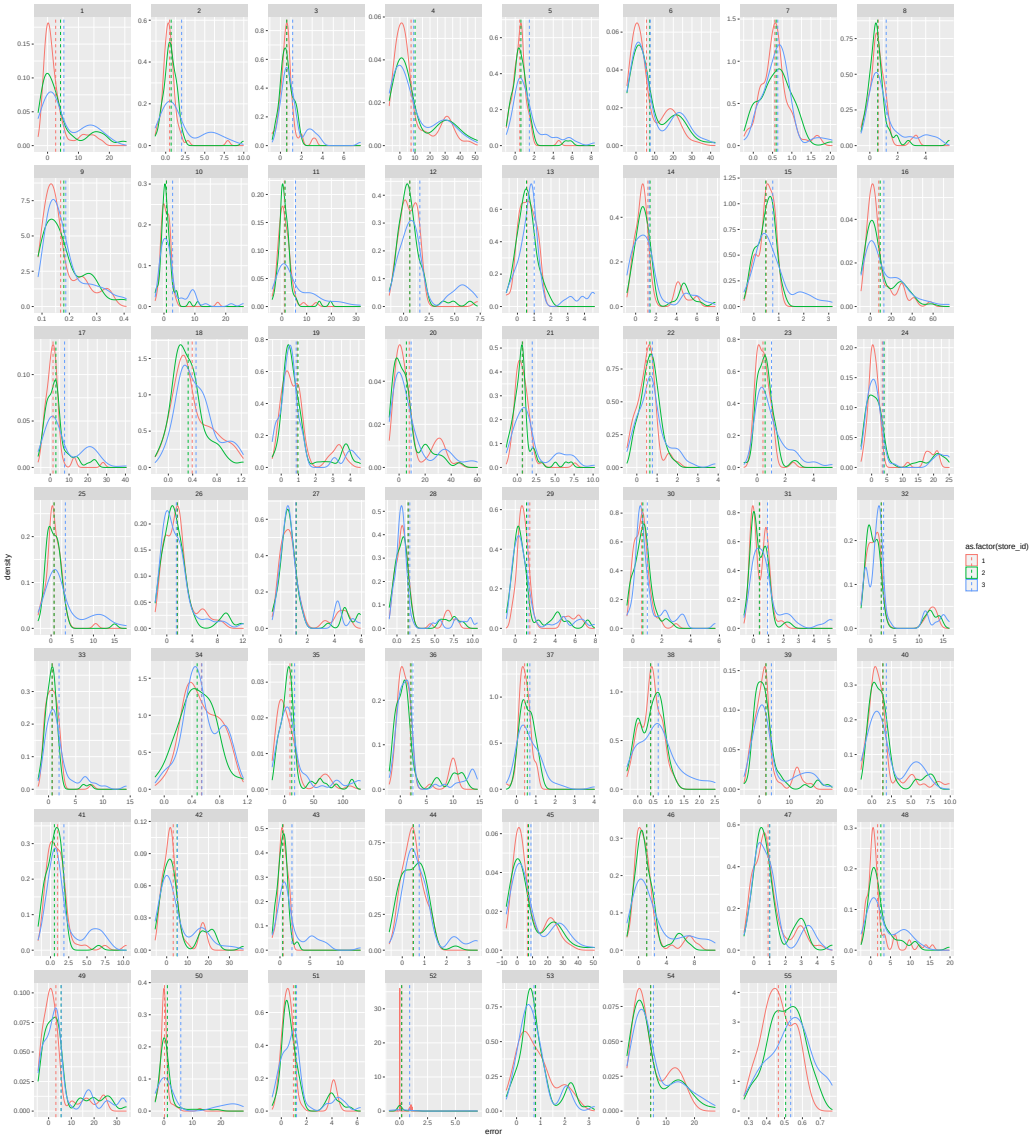


Fig. 33: Forecast error distribution by all products.

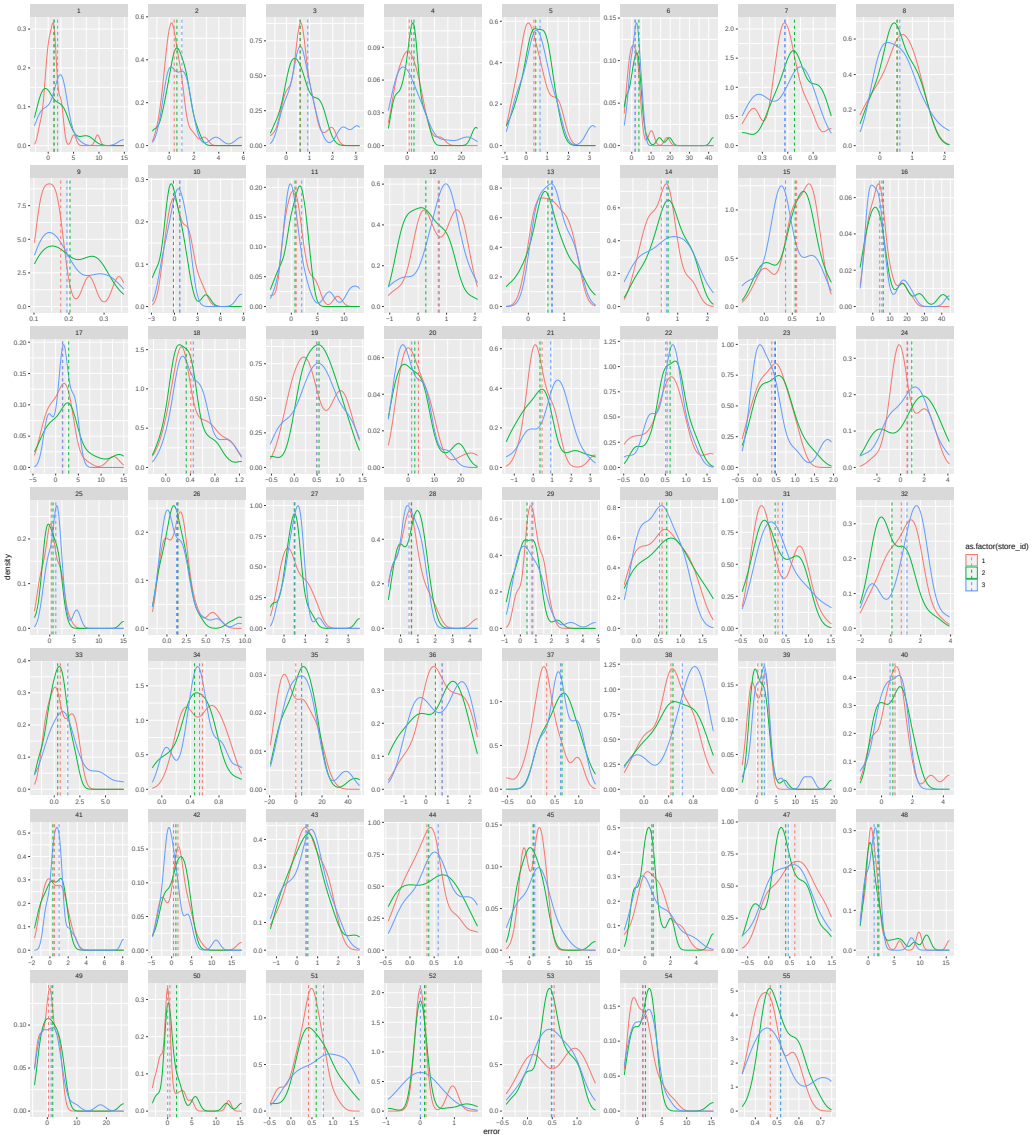


Fig. 34: Forecast error distribution by all products after excluding zero start of day stock and delivery from depot data points.

turing.ac.uk
@turinginst