UPPSALA UNIVERSITET

UPTEC STS 22019

Examensarbete 30 hp

Juni 2022

Improving sales forecasting

A study about the usefulness of geo-positioning and sales correlation data in forecasting of grocery sales

Henrik Wahlström

Civilingenjörsprogrammet i System i Teknik och Samhälle

UPPSALA UNIVERSITET

Improving sales forecasting

Henrik Wahlström

Abstract

The topic of this master thesis is to determine whether product positioning and sales correlation can improve sales forecasting of groceries. Previous studies have stated that the sales of groceries are related to their in-store placement. If this holds, it might be possible to use that relation to perform forecasting of sales. A machine learning framework is applied to perform the forecasting to fulfill this purpose. The machine learning framework consists of several supervised regression models and a neural network. The models are used to forecast sales by first considering product positioning and sales correlation and then not doing so.

One obstacle in forecasting is the need for comprehensive time series. A possible solution is to use augmented data, which was the decision in this project. However, using augmented data requires reasoning about this choice's effect on the forecasts' outcome. Other than data augmentation, re-sampling and data-cleaning are topics of this thesis.

The thesis concludes that using product positioning and sales correlation as features in machine learning models does not necessarily improve sales forecasting. Nevertheless, it is found that there are circumstances when the inclusion does improve the forecast. Those circumstances are when there are many products placed in one section and when the turnover in a section is high. More extensive studies will be needed to fully determine whether product positioning and sales correlation, in general, improve sales forecasting.

Teknisk-naturvetenskapliga fakulteten

Uppsala universitet, Utgivningsort Uppsala

Handledare: Serge de Gosson de Varennes Ämnesgranskare: Prashant Singh

Examinator: Elísabet Andrésdóttir

Populärvetenskaplig Sammanfattning

De senaste åren har dagligvaruhandeln präglats av en ständigt pågående digitalisering. Digitaliseringen har blivit ett avgörande steg för butikers förmåga att optimera sin verksamhet, och därigenom anpassa sig efter en resultatdriven omvärld. En del i att optimera verksamheten är att lyckas förhålla sig till lagermängder och tider. Risken vid stora lagermängder är dels att varje produkt spenderar stor tid på lagret, vilket bidrar till kostnader, dels att produkter med korta förbrukningstider riskerar att behöva kasseras. Motsatt finns risker vid små lagermängder. Alldeles uppenbart är det faktum att för små lagermängder riskerar resultera i utebliven försäljning, då efterfrågan överstiger utbudet.

För att på bästa sätt balansera lagernivåer för olika produkter krävs en uppskattning av hur många produkter som kommer säljas under en given tidsperiod. Det finns flera olika sätt att genomföra en sådan prediktion. Ett enkelt sätt är att anta att en produkt säljer lika mycket varje dag året om, vilket förmodligen stämmer någorlunda väl för några produkter. Många produkter är däremot mer säsongsberoende än så, för dessa krävs en mer precis uppskattning utifrån kunskap om tidigare försäljning. Det har de senaste åren blivit mer populärt att ta hjälp av artificiell intelligens för att genomföra nämnda uppskattning. Artificiell intelligens är ett begrepp som beskriver maskinell förmåga att tänka som människor (PK 1984). Det kan exempelvis handla om att förstå vad som skapar trender och mönster. Förhoppningen är att artificiell intelligens kan ta hänsyn till säsongsberoenden genom att lära sig av historiska data, och därigenom förutse framtida händelser.

Det finns forskning som tyder på att produkters försäljning är beroende av dess placering i en butik (Li, Wu, and Chen 2020). Om så är fallet finns det även relationer mellan försäljningen av olika produkter. Med hjälp av information som beskriver var i en butik produkter är placerade, samt vetskap om varje enskilds produkts försäljning, är det eventuellt möjligt att på ett mer framgångsrikt sätt predicera försäljning. Följande studie ämnar undersöka om så är fallet. Genom att titta på vilka produkter som befinner sig inom samma sektion i en butik, och hur deras försäljningsmönster förhåller sig till varandra, ska studien besvara om denna kunskap är av nytta för att predicera framtida försäljning.

Studien kommer genomföras i samverkan med företaget Pricer AB som arbetar med att utveckla och tillverka produkter för att digitalisera dagligvaruhandeln. Deras produkter innefattar bland annat digitala prisetiketter. Utifrån de digitala prisetiketter som Pricer AB tillhandahåller följer en mängd data. Bland nämnda data finns bland annat information om etiketternas (och därigenom produkternas) placering i en butik.

Utifrån studien kan ett antal slutsatser dras. Först och främst bör det sägas att vetskap om placering och beroenden mellan försäljningsmönster inte nödvändigtvis bidrar till bättre prediktioner. Det finns däremot tendenser som tyder på att vetskap om ovanstående kan vara till nytta. Ett exempel som indikerar att vetskapen resulterar i bättre prediktioner är när det finns många produkter i en och samma sektion. Anledningen till detta är att det på ett tydligare sätt är möjligt att hitta relationer mellan försäljningsmönstret, eftersom urvalet är större.

Ett annat konstaterande som är möjligt att göra utifrån studien är att långa tidsserier av data är en nödvändighet för att kunna dra nytta av placering och relationer mellan försäljningsmönster. I genomförd studie var mängden data inte tillräcklig för att få en fullständig inblick i hur vetskap om ovanstående påverkar förmågan att predicera försäljning. Det finns förhoppningar om att fördelarna skulle vara större när säsongstendenser måste tas till väga.

List of Concepts

Geo-position – Position of an item in a two-dimensional space looking from above. In this project, geo-positioning describes product positioning in a store.

Correlation – Dependency between variables. High correlation implies that the patterns of two variables are similar.

Gondola – Section of a store.

Table of Contents

1. Introduction

Nothing has influenced the sales of consumable goods as much as the modern analytical tools developed to increase cost efficiency. The contributions have resulted in a continuous process of lowering prices and optimizing other parts of the store operations, such as inventories. A prerequisite for conducting effective sales operations is the ability to anticipate future sales. Successfully anticipating future sales provides the opportunity to reduce both the inventory time and the inventory balance. Efficiently managing inventory is a critical factor for stores to be successful (Erjavec, Gradisar & Trkman 2012). High inventory levels where products spend long time result in increased costs—the opposite, with sold-out products and empty stocks, renders the risks of absence in sales. In recent years, numerous attempts have been made to find the most accurate methods to predict future sales. The development and implementation of machine learning models have taken the precision level of predictions to a previously unreached level. Intelligent algorithms now characterize the field, and the work to further improve predictions of time-series data is widespread (Arunraj & Ahrens 2015).

Almost 70 years ago, Ferber (1954) suggested that correlation could be a helpful factor in sales forecasting. Explaining how serial variables are dependent on each other, Ferber (1954) concluded that knowledge of one variable might not only indicate the future of this variable. More recently, Li, Wu, and Chen (2020) performed a cluster analysis on sales correlation. They concluded that the information could be highly beneficial for commodity promotion and store layout decisions (Li, Wu, & Chen 2020). The suggestion that sales correlation is interesting for store layout decisions does imply that geo-positioning and sales correlation could affect each other. It reasons that two products that customers would be interested in at the same time should be accessible to customers within a short period. If this holds, it might be possible to use correlations from sales data for products placed close to each other to see tendencies in the sales pattern.

The following work intends to investigate the possibility of predicting future in-store sales of consumables. More precisely, the purpose is to investigate whether the knowledge of product placement and correlation of sales can improve sales forecasting. For mentioned forecasting, supervised machine learning models were utilized. The performance of the algorithms; Linear Regression, K-Nearest Neighbor, Random Forests, Extreme Gradient Boosting, and Neural Network were investigated and evaluated. Deciding what is successful forecasting is a challenging task. Predicting future sales is too dependent on the available data, and therefore it is impossible to determine a generalized threshold separating good predictions from bad ones. Data might be missing, fluctuating, or unreliable, to mention some problems. As a result, the baseline of a good result will change from one case to another. There are different approaches to handling this problem. This project investigates whether using geo-positioning combined with correlating sales can improve sales forecasting. Since all forecasting problems are dependent on the available data, it is impossible to compare different cases to each other. Therefore, there is no need to decide on an absolute value for successful predictions. Instead, the task is to determine whether or not the forecasting is more precise when using geo-positioning and correlating sales.

1.1 Scientific questions

- Do knowledge of product placement and sales of nearby products improve sales forecasting of groceries using machine learning models?

1.2 Delimitations

The time series available for this project does not cover a full calendar year. Thereby, the data is not considering seasonality entirely. As this was the case, data augmentation was performed. The limitation of augmenting data from a time series that does not fully consider seasonality is that the augmented data will have to explain the same season as the existing data does. The extent of data augmentation is thereby limited. Further elaboration about the available data will be given in section *5 Method.*

For this thesis, only one store is considered. Therefore, it is a topic of speculation whether the results represent other stores other than the one considered.

1.3 Disposition

This thesis consists of 8 chapters. The next chapter will describe the necessary background about Pricer AB and previous scientific publications within the field. Then, in chapter 3, the theoretical foundation of supervised regression models, along with evaluation metrics for the same models, is presented. After that, a chapter about Neural Networks is provided. Chapter 5 contains a methodology presentation, mainly about the data processing needed for this project. Following the presentation of data, chapter 6 contains the results of the study. In the next chapter, a discussion of those results follows. Finally, chapter 8 contains the conclusions from the study, along with suggestions for future work related to the topic.

2. Background

2.1 Pricer AB

Pricer AB manufactures and develops digital products to retail. The company specializes in developing Electronic Shelf Labels (ESL). As the demand in retail for digital and automated solutions has increased, Pricer's product portfolio has grown. For instance, cameras are used to detect products, ESLs and gaps on shelves in order to alert retailers that products are out of shelf and need replenishment. There is a link between each item in the store and one ESL. Along with the cameras, the relationships mentioned above can help gather information about which products have run out of stock. The technology used to detect products missing from the shelves has the effect that data containing information about gaps is available over time. The same data creates the opportunity to follow the sales cycle of a particular product by seeing when there has been a gap in the shelf. Pricer is also collecting information about the position of each active ESL in a store. One use-case where knowing the exact position is helpful is when refilling stocks. It also allows for analyzing data from a positional view. That is, wherein the store sells the highest amount of products. There will be reasons to revisit this later in the report. Beyond the aforementioned position and gap data that Pricer's products produce, the company also has access to ongoing sales and in some cases, even margin data from one of the retailers it supplies with ESLs and cameras.

2.2 Related Research

Yang and Chen (1999) discuss the opportunity of applying complex algorithms to solve problems of shelf space allocation. One of the main presented objectives is how to balance inventory at an optimal level. Yang and Chen (1999) describe that, by the time of their study, the typical way of handling such problems is by using heuristic rules. That would, for instance, be to use the rolling average when projecting inventory needs. However, it is suggested that inventory balance, among other things, can be optimized by using more complex solutions. The study, which aims to solve problems related to shelf space allocation, considers other aspects than just inventories (Yang and Chen 1999). One such aspect is in-store product placement. It is suggested that the product placement affects the store's cost efficiency. Further, Yang and Chen (1999) conclude that implementing "computerized systems" to solve such problems is beneficial.

Chen and Lin (2007) provide a new dimension to optimizing shelf space allocation by implementing a data mining approach. Previous solutions have mainly involved linear and integer programming algorithms (Yang 2001). Other than conclusions about the importance of efficient shelf space management, Chen and Lin (2007) suggests the importance of managing product and transaction data. Data mining models depend on sufficient data, and therefore data availability is vital to improve those models.

Mentzer and Moon (2005) present fluctuating sales as a reoccurring problem with forecasting sales. The example given is for a product selling four units the first week, three the next, five the next, and then spiking to selling 10 000 units (Mentzer and Moon 2005 pp. 1). How should a retailer be able to predict the extent and exact time of these

spikes? Mentzer and Moon (2005 pp 75-76) discuss that the knowledge of seasonal trends is an indicator that might help retailers predict sales fluctuations. However, long time series of data is required to determine seasonal trends. Else, what looks like a pattern might be nothing else than a coincidence. The longer the time series is, the more significant conclusions about seasonal trends will be. One common forecasting technique is rolling average (Mentzer and Moon 2005 pp 77). Unfortunately, using the rolling average will not account for fluctuating demands. To capture some of these tendencies, it is possible to use the moving average instead. However, neither this method will consider historical trends needed to capture fluctuations (Mentzer and Moon 2005 pp 80-84). Adaptive and Exponential smoothing are two other methods suggested to solve forecasting problems. These techniques make it possible to involve historical data to capture seasonal trends (Mentzer and Moon 2005 pp 92-106).

Chase (2021) discusses the implementation of machine learning in forecasting sales within retail. One use case is trying to utilize big data to predict customer demand. However, Chase (2021) explains that machine learning provides some problems. Using machine learning algorithms requires proficient insight into how they should be implemented. Chase (2021) explains a different complexity compared to previously used methods. Using Machine Learning models also requires amounts of data that most companies do not have. Chase (2021), however, concludes that recent development within data processing, such as storage, provides an opportunity for machine learning algorithms. In addition, implementing cloud-based software enables higher complexity in the models, which makes implementing machine learning within sales forecasting successful in theory. In reality, the complexity of implementing machine learning algorithms and the knowledge about data required might hinder many companies from taking it to production (Chase 2021).

As touched on in the introduction of this report, Li, Wu, and Chen (2020) performed a cluster analysis on sales correlation and concluded that the information could be highly beneficial for commodity promotion and store layout decisions. The study contained clustering using the K-nearest neighbors model. Though one of the findings from the study was that in-store layout could affect sales of several products. However, the suggestion is that sales correlation can be used for other purposes as well. For example, such would include cross-marketing between stores and fields, knowing that products sold by different retailers are correlating (Li, Wu, and Chen 2020).

3. Supervised Learning – Regression

This section will present the theoretical framework of supervised regression models. The models chosen for this project are Linear Regression, K-Nearest Neighbor, Random Forests, and Extreme Gradient Boosting. Following the presentation of each model, tuning hyperparameters and improving the model's performance will be discussed. Lastly, a presentation of evaluation metrics will be provided.

3.1 Linear Regression

Linear Regression is one of the most common regression models, and it has been around for quite some time. The approach is considered easy to implement. The model is helpful for problems involving predicting a quantitative response. From a use-case point of view, Linear Regression is only applicable when the data is linear. However, that does not make Linear Regression a non-factor in non-linear cases. For complex non-linear problems, Linear Regression might not be the preferred solution any longer. However, Linear Regression is still the foundation for more complex methods (James et al. 2013. p.59-61). Linear Regression is also a preferred solution method in terms of computation time. The time complexity for Linear Regression is significantly lesser than for other regression models. The lesser time complexity is because of the expected linearity between the predictors and the response. If the actual relationship between the predictors and the response turns out to be nonlinear, the method will perform poorly (James et al. 2013. p.92). For a two-dimensional set of data, the Linear Regression model is given by equation 1.

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \mathbf{x} \tag{1}$$

$\hat{\beta}_0$ and $\hat{\beta}_1$are unknown constant coefficients. They represent the *slope* and the *intercept* of the regression. $\hat{y}$ is the predicted value (James et al. 2013. p.61). To find the optimal values of $\hat{\beta}_0$ and $\hat{\beta}_1$, the Linear Regression model computes the residual corresponding to each point. The residual is described as e and given by equation 2.

$$e_i = |y_i - \hat{y}_i| \tag{2}$$

From the residual of each datapoint i, the *residual sum of squares* (RSS) is computed. The Linear Regression model searches for $\hat{\beta}_0$ and $\hat{\beta}_1$to minimize the RSS (James et al. 2013. p.61). With the RSS being given by equation 3.

$$\begin{aligned} RSS &= {e_1}^2 + {e_2}^2 + \cdots + {e_n}^2 \\ &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \left(y_1 - \hat{\beta}_0 + \hat{\beta}_1 x_1\right)^2 + \left(y_2 - \hat{\beta}_0 + \hat{\beta}_1 x_2\right)^2 + \cdots \\ &\quad + \left(y_n - \hat{\beta}_0 + \hat{\beta}_1 x_n\right)^2 \end{aligned} \tag{3}$$

When the dimensionality of the predictors is higher than one, that is, the prediction is not only based on one input value. As a result, the formula differs some, and it is no longer possible to define $\hat{\beta}_0$ and $\hat{\beta}_1$ as slope and intercept. The formula for higher dimensionality is found in equation 4.

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p + \varepsilon \tag{4}$$

With the formulation for multidimensional Linear Regression, the RSS can again be computed from the actual and predicted value difference. The approach is the same by considering least squares (James et al. 2013. p.72-73). This time, it is considering a higher degree of residuals, shown in equation 5.

$$\begin{aligned} RSS &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n} (y_i - \hat{\beta}_0 + \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - \cdots - \hat{\beta}_p x_{ip})^2 \end{aligned} \tag{5}$$

3.2 K-nearest neighbors

K-nearest neighbors is also considered a simple approach to solving regression or classification problems. As the name indicates, the approach involves predicting through finding the most alike predictor. Using only one predictor, this becomes an easy task. However, when the number of predictors increases, defining the "nearest" neighbor becomes harder. In this case, a voting mechanism is performed, considering all the means of the neighbors (Müller & Guido 2016). For a training set X, with M number of datapoints and N features, $X = \{x_1, x_2, \ldots, x_m\}$, the regression estimate for k-nearest neighbors is given by equation 6 (Hu et al. 2014).

$$d(x_t, x_i) = \sqrt{\sum_{n=1}^{N} w_n (x_{t,n} - x_{i,n})^2} \tag{6}$$

Where x_i represents each training point in the dataset and x_t represents the response value. For each training data point, the weighted Euclidian distance to the response value is calculated. The data is then reordered based on the weighted Euclidian distance between each training point and the response value as shown in equation 7 (Hu et al. 2014).

$$0 \leq d(x_t, x_{(1)}) \leq d(x_t, x_{(2)}) \leq \cdots \leq d(x_t, x_{(n)}) \tag{7}$$

The reordered data is increasing with n. Meaning that the nearest neighbor is where the value for n equals one (Hu et al. 2014). One difficulty in this sortation is how to break ties. For low dimensionality, it does not make a considerable difference. In this case, the most common way is to select the value corresponding to the lowest index (n) value. It is preferred to prioritize between features for higher dimensionality, having some features be the tiebreaker, would the distance be equal (Müller & Guido 2016).

3.3 Random Forests

Random forests consist of trees that act as decision-makers to find the optimal split for each feature. Each tree is built by nodes, where all nodes represent the split for one feature (Ali et al. 2012). Random forests create a combination of trees such that each tree predicts the outcome based on the values of a random vector. The random vector uses the same distribution and is sampled independently for each tree. When the number of trees increases, the generalization error will converge to a limit for which the prediction can be received. One advantage of the generalization error always converging is that the model will not overfit the training data. Thereby the concern of overfitting is reduced (Breiman 2001). When applying Random Forests to regression problems, the random vector allows trees to grow such that each predictor is expressed by h. For regression problems, h takes on numerical values.

$$h(x, \theta) \tag{8}$$

for any numerical predictor $h(x)$, the mean square generalization error can be expressed as equation 9.

$$E_{X,Y}(Y - h(\mathbf{x}))^2 \tag{9}$$

The final predictor can be found by extracting the average over k of the trees.

$$\{h(\mathbf{x}, \theta_k)\} \tag{10}$$

Breiman (2001) did prove that as the trees in the forest goes to infinity, the following holds.

$$E_{X,Y}(Y - av_k h(X, \theta_k))^2 \rightarrow E_{X,Y}(Y - E_\theta h(X, \theta))^2 \tag{11}$$

The expression found in equation 11 represents the generalization error of the entire forest (Breiman 2001).

The Random forests algorithm was created as a competitor to classical boosting. The appeal of random forests is partly that the algorithm is computationally cheap. Thereby, training and testing are relatively fast. Further appeals are that the model only has two tuning parameters, making the tuning non-complex (Cutler, Cutler & Stevens 2012).

3.4 Extreme Gradient Boosting

Extreme gradient boosting (XGBoost) has its roots in tree ensemble methods. Ensemble trees are decision trees with an iterative behavior and a scorekeeping system. By creating decision trees repeatedly, the decision boundary for each feature is optimized. The difference between ordinary gradient tree boosting and XGBoost is the latter's scalability. Several factors enable the scalability of XGBoost. First, XGBoost includes a novel tree learning algorithm for handling sparse data. Secondly, it implements a weighted quantile sketch procedure, which enables handling instance weights in each iteration of the tree boosting. Lastly, and perhaps most importantly, XGBoost allows for parallel computing, making the model computation much quicker (Chen & Guestrin 2016).

To understand the functionality of XGBoost, one should first consider the framework of gradient tree boosting. This framework is based on training a model in an additive manner. To do so, finding the optimal solution in euclidian space is required. Therefore, $L^{(t)}$ found in equation 14 is minimized.

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{\,t-1} + f_t(x_i)\right) + \Omega(f_t) \tag{14}$$

Where the term Ω is given by equation 15.

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda\|w\|^2 \tag{15}$$

The term Ω is applied to penalize the complexity of the model. To optimize the objective quickly, a second-order approximation is necessary.

$$\tilde{L}^{(t)} = \sum_{i=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t) \tag{16}$$

$$g_i = \partial_{\tilde{y}^{(t-1)}} l(y_i, \hat{y}_i^{\,t-1}) \tag{17}$$

$$h_i = \partial^2{}_{\tilde{y}^{(t-1)}} l(y_i, \hat{y}_i^{\,t-1}) \tag{18}$$

Equations 17 and 18 the gradient statistics on the loss function for the first and second order. h_i given by equation 18 is also the weight of the XGBoost function (Chen & Guestrin 2016).

3.5 Optimizing hyperparameters

There is no one-fits-all solution to tuning regression models for available data. For each case of data, a different set of parameters might be optimal. The process of finding these optimal parameters is called hyperparameter tuning. Tuning hyperparameters means that the parameters that allow for the highest model performance are found. For each regression model to perform as well as possible, this project will elaborate on two kinds of tuning methods, Bayesian optimization and Grid Search. To perform Bayesian optimization, the python package Hyperopt was utilized. Hyperopt performs optimization to find the optimal combination of hyperparameters (Bergstra et al. 2015). Hyperopt's method is less computationally expensive in scanning the solution set of hyperparameter values than, for example, Grid Search. Given that large amounts of data were used, it was beneficial to use Hyperopt for the models for which the package is enabled. The disadvantage of using Hyperopt is that the method does not always converge towards the global optimum as not all parts of the solution set are explored. For this project, that risk is considered manageable as the alternative, Grid Search for all models, was not possible because of the cost of calculation time.

3.5.1 Bayesian optimization and Grid Search

In this thesis, Bayesian optimization and Grid Search has been performed to find optimal values for hyperparameters. Another well-known approach to finding optimal hyperparameters in the development of machine learning models is Grid Search. However, Grid Search inspects all combinations of specified hyperparameters. Thus, Grid Search does not take into account the trend of how the values of hyperparameters affect the outcome. Although increasing values for a specific hyperparameter produces worse and worse predictions, Grid Search will perform all tests. The problem with Grid Search is thus the cost involved in calculating all combinations (Alibrahim & Ludwig 2021).

Bayesian optimization is a probabilistic model based on estimating results for different hyperparameter values. Then, based on previous observations, the algorithm guesses which hyperparameters gives the best results (Alibrahim & Ludwig 2021). The process is performed through a certain number of iterations. Then the value that generated the best hyperparameters for the model is returned. Not considering the entire solution set for all hyperparameters makes Bayesian optimization a considerably less costly process than Grid Search. On the other hand, Bayesian optimization still allows the complete set of hyperparameters to be considered. However, the algorithm risk converging towards a local optimum instead of the global one.

3.6 Evaluation metric

3.6.1 Mean absolute error

The metric used to compare the performance of each regression model is the Mean Absolute Error (MAE). Willmott and Matsuura (2005) suggest that MAE is more suitable

than other metrics when considering average model-performance error. MAE represents how far from the actual value each prediction is in absolute numbers. The metric does not penalize large errors more than small ones in a way that Mean Squared Error would. That is considered helpful when comparing predictions of amounts. Each prediction and its error can be quantified based on the amount it differs from the actual value. Since the error equals the potential loss, it would be misleading to compare squared numbers.

$$MAE = \frac{1}{N}\sum_{i}^{N}\left|y_{i,pred} - y_{i,true}\right| \tag{19}$$

4. Neural Network

A neural network is a decentralized computation method where many simple units work in parallel. The storing of information is based on the weights between all units. The way of improving a neural network is to redistribute the weights, which will cause a chain reaction. Each neural network behaves differently depending on the architecture. Usually, the architecture includes the combination of the number of neurons, the number of layers, and the Types of connections between layers (Patterson & Gibson 2017). The most common sort of neural network contains a feed-forward multi-layered structure. One example of such neural networks is displayed in figure 2.

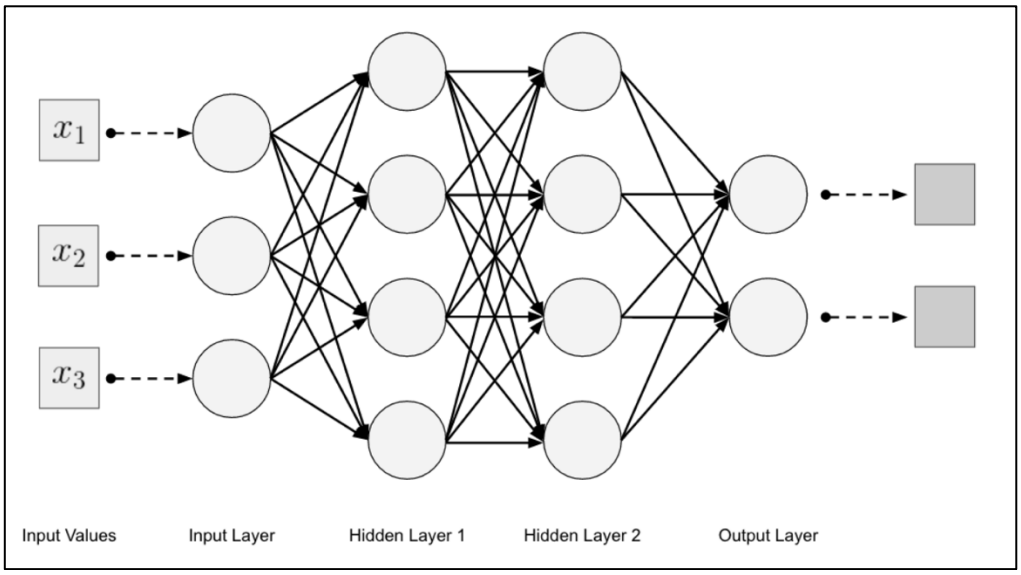

Figure 2: Feed-forward multi-layered Neural Network (Patterson & Gibson 2017)

Feed-forward multi-layered neural networks consist of one input layer, one or more hidden layers, and one output layer. As implied above, the connection between layers can vary. Most commonly, all neurons are connected to every neuron in the next layer in a fully connected neural network. All connections between neurons in different layers contain weights that update iteratively as the model trains. These weights can be understood mathematically as optimizing a parameter vector to find the minimal error. The weights in the neural network aim to amplify the signal and reduce the noise. The bigger the weight is, the more the input signal correlates with the outcome of the network. If the weight is high, the input data will be more affected than with lower weights. When training the network, the weights try to find out how each feature relates to the network's output. Meaning it is trying to find the dependency between features and outcomes (Patterson & Gibson 2017).

Each layer has at least one artificial neuron. The neurons consist of an activation function that is set dependent on the layers' purpose (Patterson & Gibson 2017). The different activation functions that have been used for this project will be explained later in this section. In figure 3 below, the activation functions' role for each layer can be visualized.

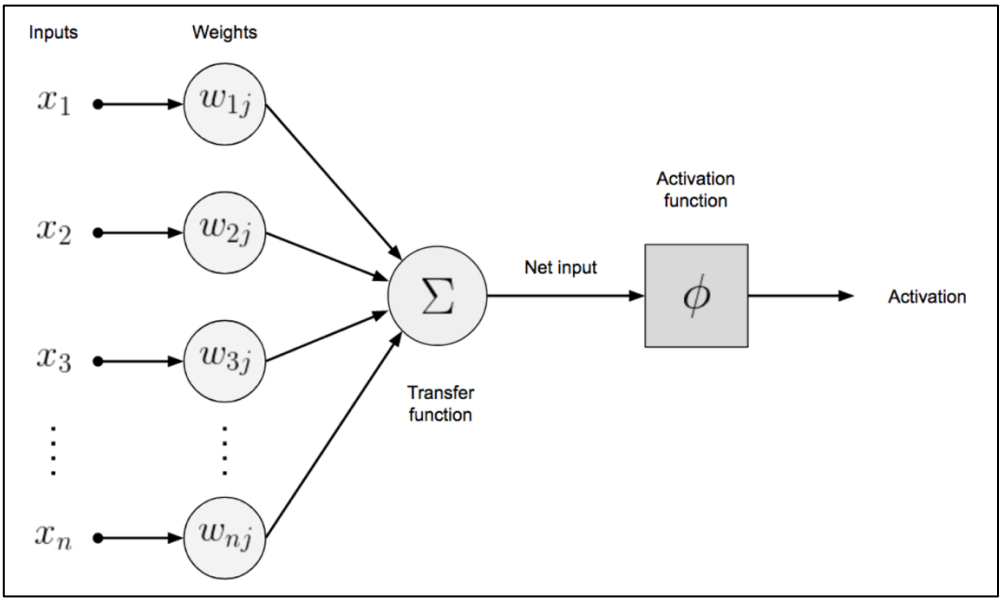

Figure 3: Example of artificial neuron (Patterson & Gibson 2017)

The purpose of activation functions is to infuse non-linearity in the connections between layers. They are scalar to scalar functions, meaning that both input and output are scalar numbers. Below, figure 4 displays the Rectified Linear Unit (ReLU) function that has been applied in this project.

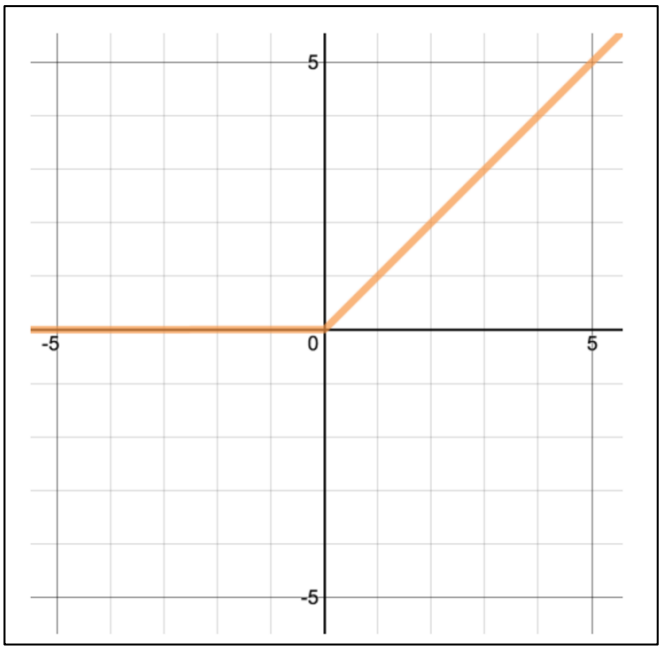

Figure 4: Rectified Linear Unit function (Patterson & Gibson 2017)

ReLU only activates a node if the input value is greater than zero. If the input is equal to or less than zero, the output will be zero. The following function describes the dependency between input and output for the ReLU activation function (Patterson & Gibson 2017).

$$f(x) = \max(0, x) \tag{20}$$

5. Method

This section will present the methodology of this project. The section starts with a presentation of all the utilized tools, followed by a display of the data process. Finally, the section contains a presentation of the predictive model.

5.1 Tools

Google Cloud Platform

Google Cloud Platform (GCP) is Googles cloud-based computing service that allows for a series of services. Among the services utilized on GCP are data storage, data analytics, and machine learning. Furthermore, GCP allows the user to run services on Googles hardware. One documented advantage of using GCP is the opportunity to keep processes in the same place (Techtarget, Google Cloud Platform).

BigQuery

The data handled for this project was collected from Google Cloud Platforms database tool BigQuery. BigQuery is a serverless solution that allows for SQL queries. BiqQuery is described to be built to serve storage and management of data analytics problems. It provides built in features such as machine learning, geospatial analysis, and business intelligence (Google cloud, introduction). For this project, the analytical tasks have not been performed directly through BigQuery. That means BigQuery has only been handled as a "classic" SQL-database with columnar storage.

Jupyter Notebook

Jupyter Notebook is a web-based development environment for notebooks, code, and data. The primary purpose is to allow users to configure and arrange workflows to solve, for instance, machine learning problems (jupyter, Jupyter). The AI platform at GCP can deploy Jupyter Notebooks.

Python Packages

Several python packages have been used during this project. Pandas have been utilized to manipulate and format the data. In addition, pandas allow reading and writing data from one data structure to another—for instance, SQL databases. It also provides data frames with automatic indexing, which is useful when handling large amounts of data (Pandas). The Numpy package was used to allow mathematical computations and random numeric generators (Numpy). To perform visualizations, plots, and graphs, Matplotlib and Seaborn were applied. Matplotlib is a library that allows for creating static, animated, and interactive visualizations in Python (Matplotlib). Seaborn is based on Matplotlib and is used to perform visualization through statistical graphs (Seaborn). The Python package Tsaug was used for data augmentation. With the help of Tsaug, calibration of the augmentation could take place. Tsaug includes several parameters that can be set depending on the scope of the augmentation (Tsaug). Geopandas was applied in the cases of geospatial data. In this project, Geopandas allowed for the smoothly handling of

geometric data. Geopandas is built from Shapely and enables geometric operations in pandas (Geopandas). For the regression models, the Scikit-learn package was utilized. Scikit-learn includes tools to perform data analysis in Python. It is built on NumPy, SciPy, and Matplotlib. In this project, Scikit-learn allowed the usage of regression models and the evaluation metrics for the same models (Scikit-learn). The Extreme Gradient Boosting model is not included in the Scikit-learn package, and instead, the XGBoost package could be utilized. Therefore, within this project, XGBoost was implemented the same way as Scikit-learn (XGBoost). Hyperopt and HPSklearn were used to tune the parameters for each regression model. The combination of the two allows for automated tuning of regression models (Hyperopt). At last, Tensorflow with Keras allowed for the implementation of neural networks. Tensorflow is an easy-to-use machine learning package that can be deployed both on-premise and, as in the case of this project, in the cloud (Tensorflow).

5.2 Data

The data handling in this project consisted of several steps, with data importation being the first. Secondly, the data had to be pre-processed to match the required format. Following the data pre-processing, some data cleaning was needed. When the data cleaning was performed, it was possible to complete the augmentation of the data. However, the data augmentation rendered some need for further data cleaning, which was the last step before the data could be considered usable as input for the regression model. The following section will dive deeper into the different steps of data handling.

5.2.1 Data pre-processing

Pricer holds large amounts of data regarding the products to which ESLs are linked. After the data had been imported, pre-processing was needed to enable using the data in predictions. The data is spread through different BigQuery tables; however, not all data was relevant. Through this report, every step of the pre-processing will not be discussed. Instead, the final dataset following the pre-processing will be considered. For each product, the information found in table 1 was usable in this project.

Table 1: Available and utilized information

Name	Description
Item Id	Unique id for each product
Barcode	Unique id for each ESL
Store Id	Unique id for each store
Date (timestamp)	Time of observation. Varies in intensity
Sales	Avarage sales over a week
ESL position x	Estimated ESL-position x in two dimensions
ESL position y	Estimated ESL-position y in two dimensions

5.2.2 Data Cleaning

Following the data preprocessing, some data cleaning had to be performed. While the data had the correct format at this stage, some irregularities in the data required some cleaning. First, we had to consider outliers in the data. The second part was to decide a threshold for how much data was needed for each product. Lastly, it was necessary to remove inconsistent data. Inconsistent is the data that does not hold updates for a significant period.

Outlier detection

There were cases where the sales was very different for one or a couple of weeks. The reason for this phenomenon is worth discussing. One reason that applied to certain products was that the product had a strong seasonal dependence. Thus, it is expected that the product's sales will deviate at a particular time each year. One such example could be the sale of pumpkins around Halloween. Sales the same week that Halloween occurs are not representative for the remainder of the year. Therefore, when augmenting data for a different time of year than the one in the existing data, it is of great importance to pay attention to seasonal trends. Data augmentation is often accurate to a higher degree when the existing data contains complete cycles, preferably an entire calendar year. It was essential to detect weeks where sales differ significantly from other weeks. Therefore, outlier detection was performed. The method used to perform outlier detection is called z-score.

Z-score

Z-score is an approach to determine how many standard deviations a data point is from the mean. The value is given by equation 21 (DeVore 2017).

$$z = \frac{(t_i - m)}{s} \tag{21}$$

where t_i represent the value of each observation, m is the mean of the dataset containing all observations, and s is the standard deviation of the dataset containing all observations.

Z-score measures how far from the average value of a data set a value is. More specifically, a measure is given of how many standard deviations a value is from the mean. Hence, the method assigns each value in a dataset a "z-score" bigger than zero. A z-score between zero and one indicates that the value is within one standard deviation, a z-score between one and two indicates that the value is within two standard deviations, and so on. The threshold for this project was decided to be three standard deviations. Thereby, all values within three standard deviations were considered to represent the sales pattern of a specific product. 99.7 percent of all values are within three standard deviations in a normally distributed dataset (DeVore 2017).

Z-score was considered a suitable method for this project as the data was not of great complexity. Only sales data were taken into consideration when searching for outliers. Since z-score is a low-demanding method from a calculation perspective, it was also a suitable choice given that it was applied to large amounts of data.

Sufficient amount of data

Data for all products were not available for the entire timespan. As a result, a minimum threshold for the amount of data had to be decided. Different regression models require different amounts of data to perform well. For instance, a more basic model like linear regression might perform exceptionally well with only some data points (provided low complexity in the data). While other models, such as deep learning methods, will require significantly more data to perform well. When considering the entire dataset, it was detected that most of the data appeared for at least seven weeks. Seven weeks compares to 49 days of data, the lowest amount for one product to appear in the dataset. It is worth noting that it is hard to determine whether 49 observations are sufficient. What is sufficient is a relative discussion that should be considered when evaluating the performance of different models. The fact that 49 observations were deemed enough for this project was mainly because choosing a higher threshold would significantly reduce the overall amount of data.

Continuous data

After the threshold for the required amount of data was had been set, it was time to find inconsistent data. The data might be inconsistent in different ways. For some products, the data appeared for the entire time series, and this data is, of course, considered consistent. On the other hand, some data showed inconsistency by missing one week, while some were missing as often as it was not. As a result of the irregularity, some rules had to be decided for what would be considered continuous data. The decision was made to consider data missing for consecutive weeks as inconsistent. That means, if two weeks were missing following each other, that product would not be kept in the dataset.

Setting the threshold of inconsistent data at consecutive missing weeks was partly based on the resampling done in later stages. The resampling will be further described in the section *Resampling of data.* What can be touched on briefly is that resampling converts weekly data to daily. In the case of missing two consecutive weeks of data, the resampling will create daily data from observations at least three weeks apart. Doing this would mean too much tampering with the data, as it might lead to untruthful tendencies.

5.2.3 Data Augmentation

One advantage of data augmentation is that it can improve forecasting accuracy (Bandara et al. 2021). Lack of data is a recurring problem in computer science. The ability to create models to determine future events is highly dependent on the data available. Bandera et al. (2021) discuss the usefulness of data augmentation when developing predictive models. Further, Bandera et al. (2021) describe two approaches to augmenting time series. The first alternative is based on creating data as consistent as possible compared to the existing time series. For instance, suppose the changes over time can be expected to be cyclical, and the time series will thus repeat itself. In that case, it is appropriate that augmented time series follow the same pattern. The risk with the approach mentioned above is if the existing data is not representative for a more extended period (Bandera et al. 2021). Thus, the augmentation results will only represent part of the sales cycle and cannot extend the existing time series. The alternate approach that Bandera et al. (2021) present is based on the fact that the augmentation should contribute to a more significant variation in the data. If the properties in the data should be expected to vary more than they do in the existing dataset, it is appropriate to create the mentioned variety. By augmenting data from these guidelines, randomness is inserted into the data. Thus, providing this kind of randomness, there is a risk that cyclic trends and patterns will be lost (Bandera et al. 2021). Inserting randomness into the data is also valuable for learning the model to handle outliers. If the existing dataset is continuous and without a high degree of fluctuation, the model will have problems determining differentiating values. Applying the randomness will enable the model to truthfully handle a greater data span.

Despite Pricer's comprehensive supply of time series, the existing dataset had to be supplemented with created data. Since the existing data had been reduced to seven weeks in the data cleaning part of this project, this became clear. However, augmenting loads of data from short time series is not trivial. As discussed in the paragraph above, the occurrence of seasonal trends requires consideration. It would be unreasonable to augment data for an entire year from the seven weeks of data. Distinguishing how far data can be augmented without interfering with seasonality differs from one case to another. In the case of this project, a period considered to remain within the same season was the following seven weeks. Therefore, the augmentation of sales data was made for the seven weeks following the existing data.

Visualization

Data augmentation requires extensive knowledge of the existing dataset on which the augmentation is based. Therefore, the first step was to investigate whether cyclic patterns could be detected by simply looking at the data distribution. It was possible to draw several conclusions about the sales pattern through visualization. The data for this project were distributed weekly. Meaning the knowledge about sales was weekly averages. As already been stated, the data should be up-sampled from weeks to days in later stages of the data processing. As this was the case, discovering daily trends and tendencies were of great interest. Finding these daily sales patterns was also crucial because noise (randomness) had to be infused into the time series following up-sampling. Why infusion of noise is dependent on sales patterns will be further discussed in a later section. Luckily enough, there were daily data available for other retailers. The relative sales between weekdays are seen in figure 5 below for one of those retailers.

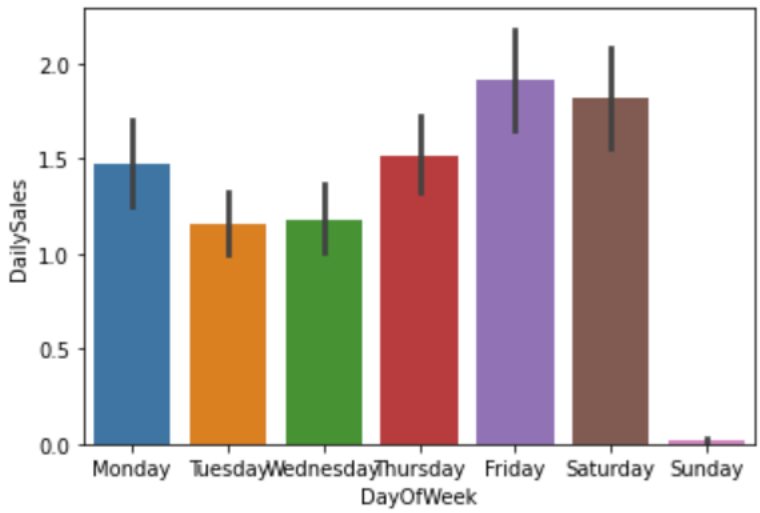

Figure 5: Distribution of daily sales. The Y-axis represents the average number of products sold per day for all products in a store.

From figure 5, it is possible to visualize a significant difference in sales between different days of the week. The store represented in figure 5 is closed on Sundays, of which daily sales are equal to zero. Figure 5 shows high peaks in products sold on Fridays and Saturdays, while sales are lowest on Tuesdays and Wednesdays. The pattern mentioned above appeared for all stores for the examined retailer. Thereby, it was possible to conclude that the same kind of pattern applied for several stores.

In the same way as for the daily data, weekly visualization was performed. Seeing cyclical patterns by just looking at the data proved more difficult for weekly sales. Figure 6 shows the weekly sales for a specific product. The x-axis represents weeks and the y-axis represents Swedish crowns (SEK).

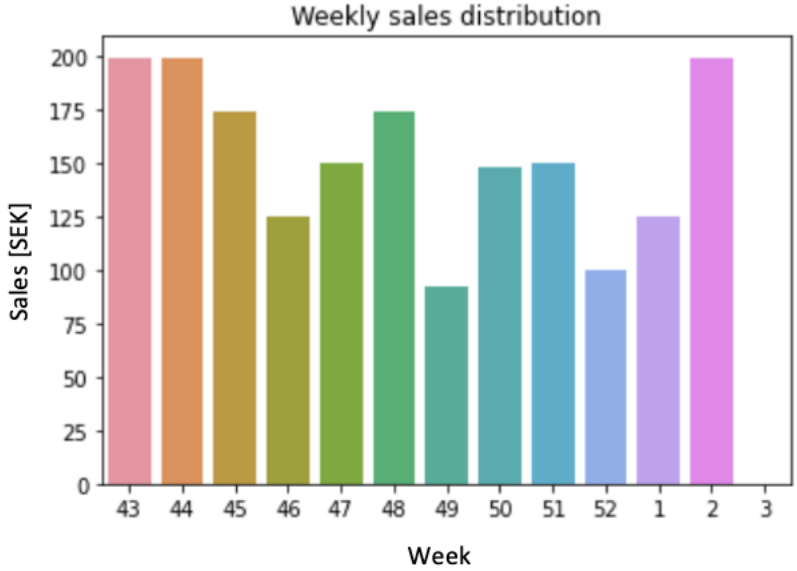

Figure 6: Weekly sales for one product.

What could be distinguished when visualizing weekly sales for different products was the tendency that sales increase in the weeks following Sweden's standardized salary day, the 25th. Furthermore, the week following the 25th of each month often contained a recurring increase. The increase mentioned above in sales did not occur for every product. However, this trend was present for a majority of the products in the store.

Another trend observed through visualization was the impact of Christmas and New Year. The sales and margin data for the last weeks of December varied differently from other weeks. The problem that presented itself was that the deviation varied between a significant increase and an equally significant decrease, depending on which product was studied. The behavior proved to be challenging to unify. As the weeks for which data were to be augmented do not include Christmas and New Year, the decision was not to use these weeks as a basis for the augmentation. Instead, the mean values from surrounding weeks were used.

Validation

Investigating how well the augmentation of data represents the reality required validation of the generated dataset. Measuring accuracy for an augmented dataset can be done in different ways. One way is to perform data augmentation on a subset of the data, allowing the other part to validate the augmentation. This technique was applied for this project. Through this procedure, various parameters in Tsaug could be adjusted to determine what gave the most satisfactory result.

A problem that arose in the validation phase was that the other half of the existing dataset, the validation data, contained the weeks before Christmas and New Year. Therefore, when comparing the augmented data and the validation data, no account was taken of the precision over the weeks when Christmas and New Year occurred.

Outcome of augmentation

It was desirable to ensure that the augmented data followed the same pattern as the existing data when augmenting. The desire to make the augmented data as interchangeable as possible with the existing one was based on knowledge of the distinct sales patterns. Since each week did not vary to a large degree, it was probable that future data would follow the same pattern. Furthermore, it was also possible to state that the data showed cyclical patterns every month. The goal was to maintain the reoccurring cycles for the augmentation. Hence the decision was made that the augmented data should not deviate extensively from the existing data.

The best results were obtained when augmentation was performed only by letting a data point vary between a percentage span of the current value. For the first week of a month, the augmented data is based on the corresponding value in the existing dataset. Figure 7 shows the weekly sales in SEK for a specific product. Combining both the real and augmented dataset.

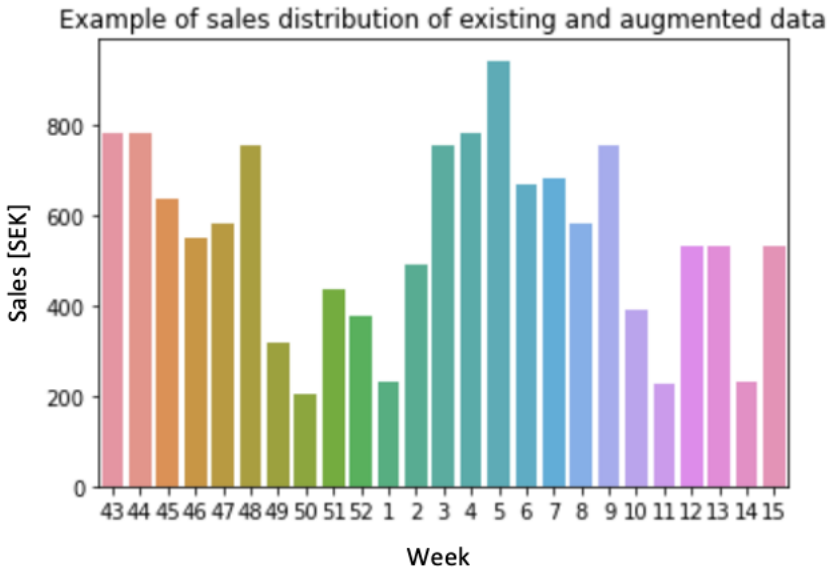

Figure 7: Weekly sales for one product. Combining both the real and augmented dataset.

Considering the distribution of weekly sales from the existing dataset and the augmented one might not indicate how well the augmentation performed. Therefore, augmentation was done for daily sales as well to visualize this. An example of augmentation for an individual product is shown in Figure 8 below.

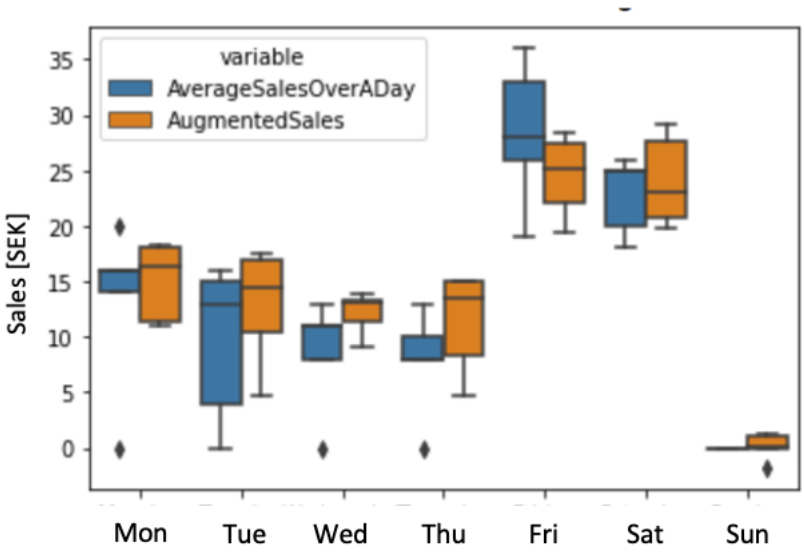

Figure 8: Distribution of existing data compared to augmented data for a specific product.

Figure 8 shows how the distribution of the augmented data differs from the actual data. It shows that the augmented data does not choose to predict outlier values. In the design of the augmentation model, the decision that deviations would be minor from the original data was made. Therefore, the augmentation should not predict which days the sales will stand out, neither a deviating increase nor a decrease. Nevertheless, the result of the daily augmentation was considered satisfactory as the created data essentially showed the same tendencies as the existing one.

5.2.4 Resampling of data

The distribution of existing sales was weekly. Therefore, the data were formatted as weekly averages. Though, the prospect was to perform daily forecasting. Thus, it was necessary to perform resampling to enable daily prediction of the sales data. Resampling is an approach to adjusting the length of time for which data exists. Going from a more frequent unit of time to a less frequent one (for example, days to weeks) is called down-sampling. The equivalent, up-sampling, is used to go from a less frequent unit of time to a more frequent one (for example, weeks to days). For this project, up-sampling was performed to obtain daily data.

For this project, up-sampling has been performed. Up-sampling becomes the same as interpolating values from the weekly averages. There are several ways to handle this problem. One way would be to use imputation theory, either using the mean or a regression model (Shahzad 2020). For this project, a combination of those was performed. From the weekly average, a daily average was found. Those daily averages were set to be Sunday sales. The rest of the days were assigned values that created a smooth line from one Sunday to the next for each week. The resulting sales pattern can be found in figure 9.

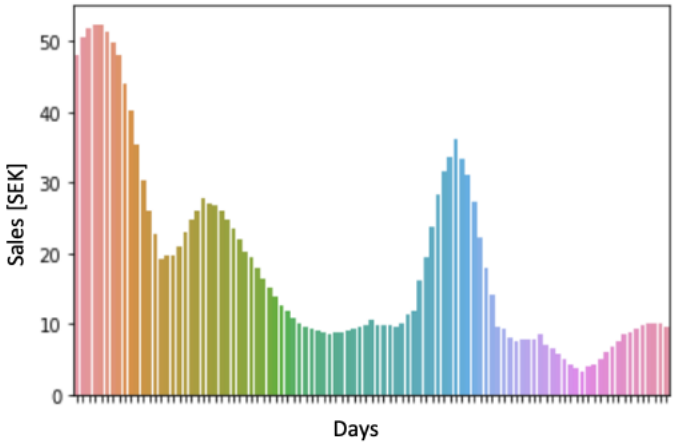

Figure 9: Example of up-sampled sales for one product over time

The up-sampled data becomes too linear in its day-to-day pattern. A consequence is a need to infuse noise into the time series. The infusion of noise will be the topic of the next section.

5.2.5 Infusion of noise

Adding noise to a time series is more complex than it might seem. However, there are suggested solutions to adding noise when there is previous knowledge about noise for a time series. Suggested solutions to adding noise to time series are recognizing dynamic or observational noise and infusing those into the time series (Nakamura & Small 2006). The problem in the case of this project is that we know nothing about the noise in daily sales. The only knowledge about weekly sales cycles is found from other retailers. As the alternative would be only to consider linear time series like the one shown in figure 9, it was decided to infuse noise based on sales tendencies for other retailers. The value will be drifting between percentual values depending on which weekday it is. The daily sales pattern shown in figure 5 was used as a point of reference when infusing noise into the sales data. A factor randomly changed the sales data based on the weekday. The randomness ensured that even though the sales on average were lower on Tuesdays than on Mondays, all Tuesdays should not contain lower sales data than all Mondays. Following the infusion of noise, the sales pattern of the product shown in figure 9 now takes the shape of figure 10.

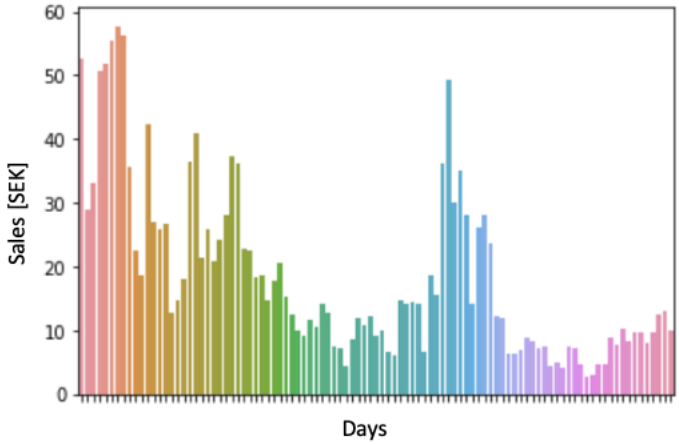

Figure 10: Example of the sales pattern after infusion of noise for the same product as in figure 5

Infusing noise in the above-described way might lead to some misrepresenting data. However, this project aims not to create an optimal algorithm but to compare performance when geo-positioning and correlating sales are considered. As the same dataset will be used for forecasting both with and without considering geo-positioning and correlating sales, this should not affect the study in either direction. Also, the implemented sales pattern has been studied for other stores, making it reasonable to represent the truth to some extent.

5.3 Predictive model

The process of finding the most optimal machine learning model is often time-consuming. The first part of creating the regression model was to understand which solution method was most suitable for predicting sales. Six different models were used to get knowledge of what method would be most accurate in its predictions. The methods tested were: Linear Regression, K-Nearest Neighbor, Random Forests, Extreme Gradient Boosting, and Neural Network. The methods were considered appropriate as they solve regression problems in different ways. They are also widely used in applications both in industry and science.

Using several models was based on the risk of models handling the feature of sales correlation differently. For instance, one model might be unsuited to solve sales forecasting overall but still improved by considering sales correlation and geo-positioning data. In this case, stating that sales correlation and geo-positioning data are helpful might be untruthful in a grand scheme since it is not ascertained that the data is helpful in cases of suitable models. Therefore, the actual usefulness of sales correlation and geo-positioning data in sales forecasting would not have been examined.

5.3.1 Features

A regression model was created to perform sales forecasting. Since this project aims to see whether geo-positioning and sales correlation can improve forecasting, the input data will be designed in two different ways. The input for the model considering geo-position and sales correlation consisted of the data found in table 2.

Table 2: Input for the model considering geo-position and sales correlation

Item ID
Sales previous day
Sales two days ago
Weekday
Highest covariance for nearby products
Lowest covariance for nearby products
Sales tendency for product with highest correlation
Sales tendency for product with lowest correlation

The input for the model **not** considering geo-position and sales correlation consisted of the data found in table 3.

*Table 3: Input for the model **not** considering geo-position and sales correlation*

Item ID
Sales previous day
Sales two days ago
Weekday

5.3.2 Training

The entire dataset consisted of information on 1323 products over 98 days. The data was divided into a training and test set for training the models. The distribution between test and training sets was random, which did not guarantee a particular product to be present in both data sets. The training data consisted of 80%, while the test data consisted of 20% of the total amount of data. Using Hyperopt, the hyperparameters of each regression model were then tuned before completing the forecasting.

5.3.3 Forecasting

For each model presented in earlier parts of this report, forecasting was performed for the 14 upcoming days. The last two weeks of the dataset were removed before feeding the

model the data. Then, through an iterating process, the model projected the sales for each product one day at a time. The forecasted data could then be compared to, and validated by, the actual, removed data. The two-week forecast was done in two different ways for each regression model. First, forecasting was done without considering correlating sales of nearby products. Secondly, forecasting was done considering correlating sales of nearby products.

As a foundation of discussion, forecasting was performed for selected parts of the store. First, the gondolas with the highest number of products were localized. Performing forecasting for these was decided to provide some material for discussion. This project aims to investigate whether correlating sales of nearby products can be utilized as a feature when predicting future sales. The reason finding the gondolas with the most products was interesting was since the more products there are, the likelier the sales patterns are to correlate and not correlate. The probability that the sales of two products are correlating gets greater when there are more products to compare. It became evident that the number of products placed in a gondola dropped off after considering the five gondolas with most products. As a result, forecasting was performed for these five gondolas.

Secondly, forecasting of the gondolas with the highest sales was performed. The reasoning behind considering the best-selling gondolas was that it was evident that these gondolas had been exposed to customers. High-selling products are harder to predict as they tend to fluctuate more. However, when these products correlate in sales, it is unlikely a coincidence. The same cannot be said about low-selling products. Those might share sales patterns due to low demand for the products. Therefore, considering sales tendencies for these products is probably not going to indicate any trends for other low-selling products, even though their correlations over time can be much alike. Forecasting was done for the ten highest-selling gondolas. The decision to use the ten highest-selling gondolas was made because it did not take away too much data. Contrary to the gondolas with most products, there was no apparent drop off in this case.

Forecasting of selected parts of the store was only done with one of the best-performing models from the “full-store forecast”. Therefore, deciding which model to use for this part had to consider performance in absolute numbers and computation time. The main reason for considering several models was to ensure that some models were unsuitable for solving the forecasting problem of this project. Therefore, considering all models for selected parts of the store will not add any further aspects.

5.4 Method discussion

This section discusses the method selected for this project. First, the selection of models will be touched on. Secondly, the data process and decisions connected to it will be discussed.

One of the purposes of this project is to find what model would be most helpful in forecasting sales data. As described earlier in the report, the decision was to select five regression models and create a neural network for regression and evaluate each result. However, one problem with this methodology is that the optimal model might not be one of the selected ones. Therefore, this project will not be able to determine which model is

the best for forecasting sales. Neither will it be possible to conclude an absolute number with the best possible forecast. However, deciding the best performance in absolute numbers is not much of a limitation. Each forecasting case depends on the available data, meaning what would be performed in this project cannot be compared to other performances with different data. Regarding selecting six different forecasting methods, the aim is to suggest what kind of model is most beneficial. It would not be possible to try each regression method available. This part should be well considered when concluding this project.

Several decisions have been elaborated on in earlier parts of this section regarding the data process. Those include the amount of data used, the threshold for both a sufficient amount of data, and the outlier detection. However, the decisions are most often a result of conditions from the initial dataset. For example, the amount of data, as well as missing data, required a reduction. The use of augmented data should be touched on as well. For instance, only using low complexity regression models would not have required an extension of the dataset. However, augmentation was needed to enable the implementation of more complex regression models. As all models should be trained and evaluated on the same set of data, an extended dataset used for all models became the preferred option. The advantages of implementing not only low-complexity models were too evident.

6. Geo-positioning

Pricer AB holds geo-positioning data for some of the retailers. The position of each ESL is located daily, with various accuracy. The geo-position is a two-dimensional position looking from above. Figure 11 is illustrating a store in two dimensions. Figure 12 is illustrating the same store, also including positions for ESLs.

The precision of the ESL geo-position varies depending on the transceivers, ESLs and other store obstacles. To receive information from each ESL, the transceivers send out signals with reoccurring intervals. If the ESL is within reach of the signal, it will respond to the transceiver. For each store there are several transceivers positioned in the roof. The above-mentioned signal between the transceivers and ESLs are affected by objects in the store. For instance, shelfs and products might be in the way of the signal, hindering it from going straight from the transceiver to the ESL. In these cases, the signal will "bounce" between objects before reaching the ESL, an example of this is displayed in Figure 13.

Figure 11: Store in two dimensions, showing position of gondolas. The values for the x- and y-axis represent length in meters.

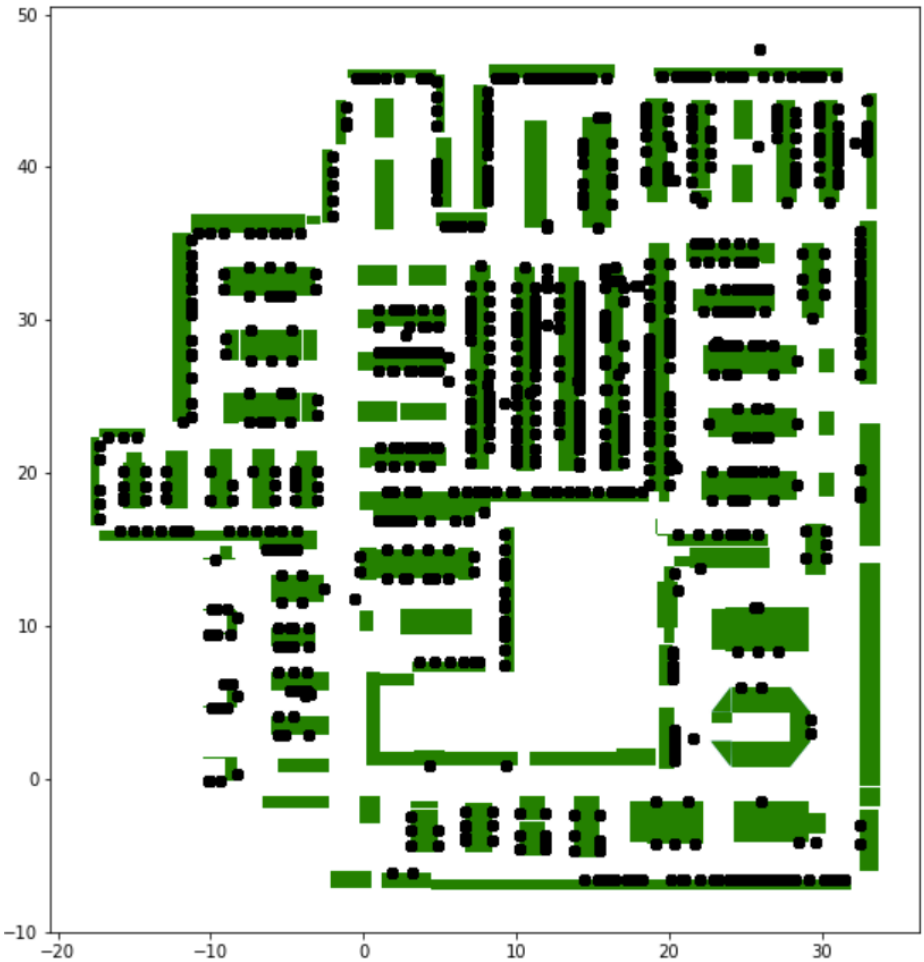

Figure 12: Store in two dimensions with gondolas and products. The values for the x- and y-axis represent length in meters.

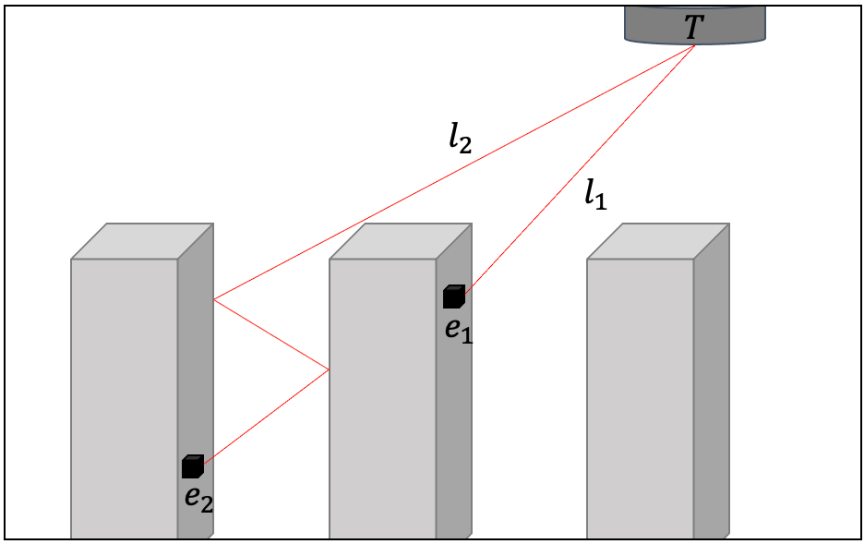

Figure 13: Display of signals between the transceiver and ESLs

Figure 13 displays one of the problems for reliably computing the geo-position of ESLs. What shows is the occurrence of a signal being disrupted by an obstacle on its path between the transceiver and the ESL. In one of the cases, the signal l_1 between the transceiver T and the ESL e_1 is not hindered. In this case, it is possible to conclude that the length of the signal l_1 is the actual length between the transceiver and the ESL. It gets more complicated when studying the signal l_2 in the figure. l_2 bounces against the shelf-gondolas twice before reaching the ESL e_2. Computing the actual length between T and e_2 becomes much more difficult. The signal does not give any information about whether it has bounced between obstacles or taken a straight path.

In the case of Figure 13, the obstacle is in the form of a gondola. Unfortunately, these occurrences do not have a straightforward fix. To some degree, it might make sense to place gondolas to minimize the disruption of signals between the transceiver and the ESL. The counterargument to that might be that the placing of gondolas is vital for the store's actual sale performance (Shaw et al. 2020). However, the same argument cannot be proven true for increasing the precision of geo-positioning.

One other option would be to implement some logic based on the placement of the gondolas. The fact that they are static would theoretically make it feasible to align the computation of each ESL position by considering the expected signal path. However, this would require the computation to be unique for each store where geo-positioning is implemented. It would also require remodeling the algorithm every time gondolas are moved around. Nevertheless, we have not discussed the case of non-static obstacles, which will make the above-suggested solution unfeasible.

Gondolas are not the only obstacle disturbing the signal between the transceiver and the ESL. There are also more challenging to predict obstacles that come in the way. For instance, when signals are sent during opening hours, customers and staff might unintentionally be covering the signal's path. The same goes for groceries that are about to be replenished. Unfortunately, those obstacles cannot be accounted for in advance, which makes it impossible to align some logic to correct for obstacles.

So far, it has been determined that the computation of geo-positioning is affected by obstacles in the store. Which, of course, will affect how precisely it is possible to decide the position of each ESL. One sub-track of this thesis has been to see whether it is possible to improve the accuracy of the geo-positioning algorithm using Neural Networks.

The question is whether improved accuracy of the geo-positioning model is essential to predicting future sales. Several factors should be considered in this reasoning. First, how well does the current geo-positioning work? The answer is that it varies. For some stores, the ability to decide the positions of ESLs works well. For others, it does not. However, predictions are most often in the ballpark of what should be expected. In most instances, the predicted value is in the right aisle of the actual ESL. That leads to the second factor, do minor improvements matter for this project? This question is two-folded since the answer is both yes and no. Here is why: It does matter to confirm the significance of correlating sales involvement. Due to the fact that the available data is relatively limited, sales of products might correlate by coincidence. The risk of correlating by coincidence is significantly greater for products placed in different parts of the store. One of the premises of this thesis is that the customer considers products placed in the same section at approximately the same time. With this aspect in mind, one would say that it is vital to

determine the exact position of products. Counter-wise, we already know that almost all predictions are placed in the same aisle as their actual position. There is no proof that the distance between products affects their sales relation (in fact, things are pointing towards this not being the case). With this in mind, it is probably just as fair to consider correlating sales of an entire gondola as considering only one section. The timespan for which customers are considering the entire gondola vs. the section is, in this case, considered negligible.

6.1 Correlation

Pearson correlation was applied to determine correlation between sales of products (Houle 2008).

$$Corelation = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2 \sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{21}$$

Where X_i represent values of the x dataset Y_i represent values of the y dataset, $\bar{X}$ is the mean of the x dataset, and $\bar{Y}$ is the mean of the y dataset.

6.2 Correlating sales

The correlation of sales has been considered for products placed on the same gondola. The hypothesis to test is whether products on the same shelf affect the customers' decision to buy other products on the same gondola. Since these products are exposed to the customer within a short period, the customer might also be influenced to consider other products. That would intend that correlation of sales patterns for products placed on the same gondola is not as likely to be a coincidence.

The correlation of sales patterns could be found by considering all products placed on the same gondola; figure 7 displays a geometric view of all gondolas. Each product's sales pattern was compared to all other products on the same gondola. Again, the highest and lowest correlation was stored. With the highest and lowest correlating sales patterns decided for each product, it was of interest to find the day-to-day sales tendency. Let us consider product x and the highest and lowest match in correlation. For example, if for day_i, the sales were 10, and for day_{i+1}, the sales were 5, the stored tendency would be 0.5, as the sales for day_{i+1} is 50 percent of the sales for the day_i.

7. Results

The following section contains a presentation of the result of this project. The first part of the section will be about the results for each model when forecasting one random day from the train- and test-split. Furthermore, it will be presented what results are found when using augmented data and the same split. The second part of the results section will contain a presentation of the results for each model when forecasting two unknown weeks. For this second part, which is more time-consuming due to the amount of data, computation times will be considered for each model.

7.1 Projecting one day

Each regression model used in this project holds several hyperparameters that can be set in different combinations. Tuning those hyperparameters have been done using the python package Hyperopt. The forecasting results will be presented in the following sections for each model. The neural network setup and tuning contain a different kind of work process. There is no automatic way to find the best layer design and parameters for a deep learning model. Therefore, the activation functions and optimizer were set manually to find the optimal model. Below, results will first be presented for the regression models before considering the results from the neural network.

7.1.1 Regression models

Figure 14 below shows the mean absolute error for all regression models. The mean absolute error represents the average error per product in SEK.

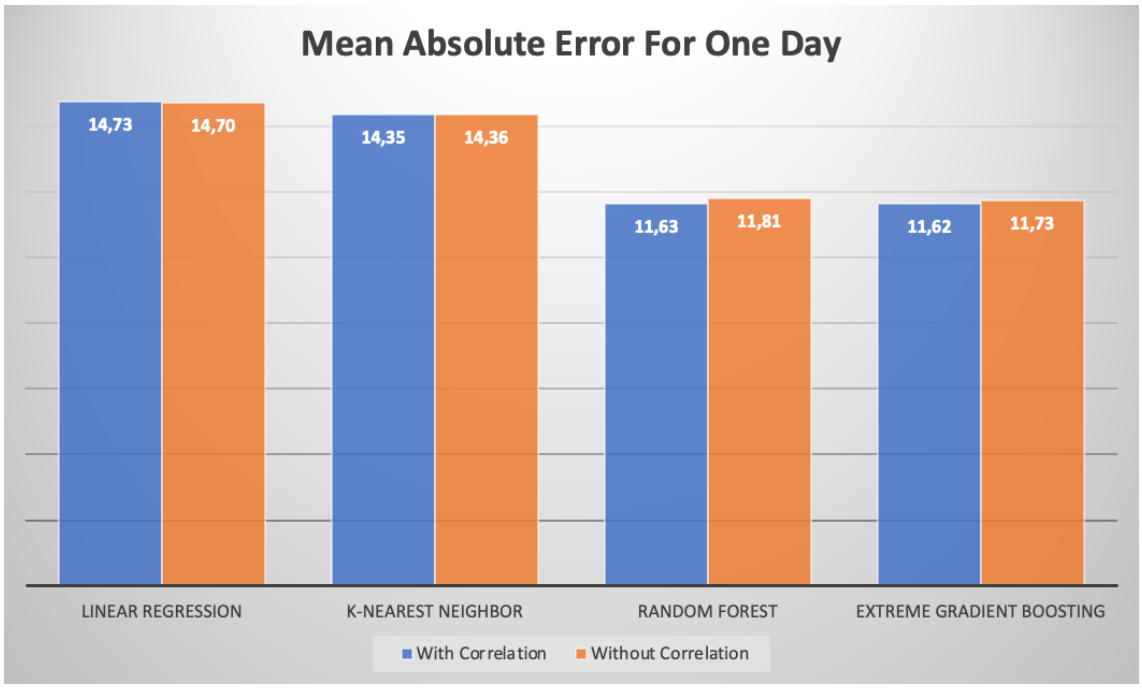

Figure 14: Mean absolute error for each model in SEK. With and without correlation.

It can be seen that the inclusion of correlation does not have a significant impact on any of the models. However, the best-performing models, Random Forest and Extreme Gradient Boosting showed slight improvement when correlation was considered. It can also be seen that the models vary in performance. Further discussion about the results can be found in the discussion section of this report.

7.1.2 Neural Network

When designing the neural network model, several aspects had to be considered. The most common type of network, when applying it to regression problems, is sequential models. A sequential model was considered the most applicable for this problem as well. Deciding the number of layers and their activation function is somewhat a trial-and-error process. However, considering the data, one could decide on a framework for the model architecture. The data in this problem is not of great complexity. Applying many layers would, most reasonably, not improve the model significantly. Some different kinds of layer architecture and activation functions were tried ahead of finding the best-performing model. For which the design can be found in figure 14. For each layer, “ReLu” was used as the activation function.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_2 (Conv1D)            (None, 1, 32)             288
_________________________________________________________________
dense_20 (Dense)             (None, 1, 128)            4224
_________________________________________________________________
dropout_4 (Dropout)          (None, 1, 128)            0
_________________________________________________________________
dense_21 (Dense)             (None, 1, 64)             8256
_________________________________________________________________
dense_22 (Dense)             (None, 1, 32)             2080
_________________________________________________________________
dense_23 (Dense)             (None, 1, 1)              33
=================================================================
Total params: 14,881
Trainable params: 14,881
Non-trainable params: 0
_________________________________________________________________
```

Figure 15: Design of deep learning model

The model was trained with the Adam optimizer for 200 epochs with early stopping with the patience of 20. Below is Figure 15, displaying the mean absolute error in SEK for training and testing for each iteration when sales correlation is used as a feature.

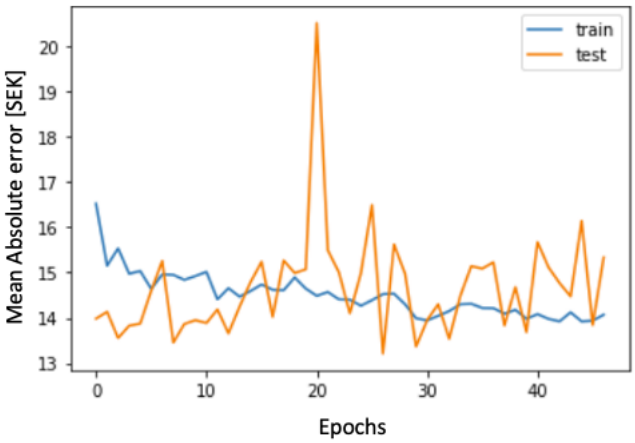

*Figure 16: Mean absolute error for training and testing **with** correlation*

From Figure 15, it is shown that the test error is volatile to a high degree. However, the test error becomes somewhat consistent between the values 13 and 16, with the best mean absolute error for testing being 13.21 when using sales correlation as a feature. Applying the same model to the data, this time not considering correlating sales of nearby products rendered in the train- and test mean absolute error shown in Figure 16.

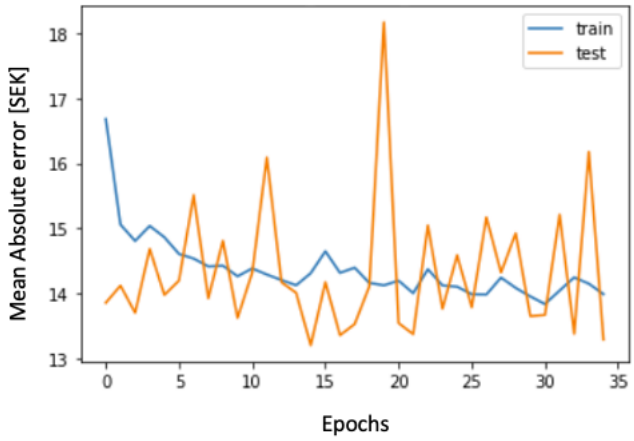

*Figure 17: Mean absolute error for training and testing **without** correlation*

As shown in Figure 16, the mean absolute error for testing is acting similar as when correlating sales were used as a feature. However, the best performance of the test error is not as good in this case as when correlating sales were used. The epoch with the lowest mean absolute error for the test data was 13.19. The difference is close to none. What that means will be elaborated on in the discussion section.

7.1.3 Summary

In a way, projecting one day might be the most indicative part of how well each model can forecast sales data. This is because it is the only case when it is reassured that all models are considering the exact data for training and testing. Though, it might not be as helpful as telling what can be implemented in the industry. The ability to predict one day alone is not enough of a time span to be able to set stock levels accordingly. When predicting sales using regression models, the hyperparameters of each model must be tuned. Below, in Table 4, is a summary of the results for each model with tuned hyperparameters when using the real data. All values in the Table are represented by mean absolute errors in SEK.

Table 4: Results for predicting one random day

Model	Linear Regression	K-nearest neighbor	Random Forest	Extreme Gradient Boosting	Neural Network
Mean Absolute Error	14.73	14.35	11.63	11.62	13.21
Mean Absolute Error without Correlation	14.70	14.36	11.81	11.73	13.19

7.2 Projecting two weeks

Forecasting for two weeks might be more useful in the industry. However, during evaluation it is essential to consider what implications might cause the results. This topic will be further elaborated on in the discussion section. The statistics of each model's two-week forecasting can be found in Table 5. The average sales value for one day is 49.5 SEK per product. The errors found in the table represent mispredictions in SEK.

Table 5: Results for two-week forecasting

Model	Linear Regression	K-nearest neighbor	Random Forest	Extreme Gradient Boosting	Neural Network
Average Daily Error with Correlation	30.77	25.26	22.09	22.86	36.06
Average Daily Error without Correlation	30.28	25.25	23.04	23.80	32.13
Mean absolute error with Correlation	430	354	309	320	505
Mean absolute error without Correlation	424	353	323	333	450

Table 5 shows that the difference between using sales correlation and not doing so becomes more evident when forecasting two weeks than when forecasting one day. At least when studying the most appropriate models, Extreme Gradient Boosting and Random Forest. The other models either show no difference in their results or see a slight decrease in the performance.

Seeing a performance difference close to one might seem matterless. However, the context should explain that with more than 1300 products, the cumulative wrongful prediction will approach 20 000 SEK over two weeks.

7.3 Forecasting part of store

7.3.1 Part of store with most products

As a foundation for further discussion, forecasting was done for the most product-populated part of the store. Only one of the best-performing models from the previous section was utilized for this exercise, the Extreme Gradient Boosting model. The five most product-populated gondolas in a store were selected to perform this task. The mean absolute error with sales correlation was 9.22 SEK. Correspondingly the mean absolute error without sales correlation was 9.32 SEK.

Table 6: Results when forecasting part of store

Model	Extreme Gradient Boosting
Mean Absolute Error	9.22
Mean Absolute Error without Correlation	9.32

The results from forecasting 14 days can be found in table 7 with errors given in SEK. The average sales value for one day is 36.7 SEK per product.

Table 7: Results when forecasting part of store for two weeks

Model	Extreme Gradient Boosting
Average Daily Error with Correlation	12.80
Average Daily Error without Correlation	13.94
Mean absolute error with Correlation	179
Mean absolute error without Correlation	195

It is shown that correlating sales of nearby products provide a more significant difference when only considering the highest product density gondolas. The difference is still relatively small but still obviously present. For example, the average sales value is lower when only considering the gondolas with most products. However, the difference in absolute prediction error does become even more significant in this case. It suggests that the percentual difference in using sales correlation will become significantly larger than when considering the forecasting for the entire store. The discussion section below will elaborate on how this relates to the forecasting of an entire store.

7.3.2 Part of store with highest sales

Considering the most product intense shelves is one way of indicating whether sales correlation is a helpful feature in forecasting sales. Another way of considering the effect of including sales correlation as a feature is to look at the gondolas with the most frequently bought products. Those products are exposed to customers frequently, and their sales patterns should be accurate over time. The ten gondolas with the highest total sales were selected to perform this task. The results are shown in Table 8. The mean absolute error with sales correlation was 27.69 SEK. Correspondingly the mean absolute error without sales correlation was 28.44 SEK.

Table 8: Results when forecasting shelves with highest sales

Model	Extreme Gradient Boosting
Mean Absolute Error	27.69
Mean Absolute Error without Correlation	28.44

The results from forecasting 14 days can be found in the table 9. Once again, the errors in Table 14 represent mispredictions in SEK. The average sales value for one day is 113.3 SEK per product.

Table 9: Results when forecasting shelves with highest sales for two weeks

Model	Extreme Gradient Boosting
Average Daily Error with Correlation	42.78
Average Daily Error without Correlation	45.54
Mean absolute error with Correlation	599
Mean absolute error without Correlation	638

When the gondolas with the highest sales are considered, sales correlation provides an advantage in forecasting. Since the average sales of products are higher than in earlier instances, the forecasting error becomes quite large. However, sales forecasts perform better when sales correlation is taken into consideration.

8. Discussion

As can be seen, the differences between using correlating sales as a feature do, in some cases, provide a difference in the models' ability to predict sales. There is an interesting tendency for the best-performing models, Extreme Gradient Boosting and Random Forest, to perform better with the knowledge of sales correlation of nearby products. The differences get more evident when considering the prediction of the entire two-week span than for one day only. Interestingly, the two best-performing models perform better when geo-position and correlations are considered. It should be elaborated on why these minor improvements occur. It does seem like sales correlation of nearby products is a helpful feature, though a small one. The question arises regarding if there are situations and opportunities where sales correlation is better to utilize.

Something to notice is that when forecasting more extended periods (in this case, two weeks), the inclusion of correlation of nearby products provides better performance. While the one-day projection is not significantly affected by the inclusion, more extended periods seem to be. Why that is the case should be discussed. One hypothesis would be that sales correlation acts as an "insurance" against outliers. While not including correlations based on positioning, wrongful forecasting will only worsen each day. Accounting for sales correlation of nearby products might help by considering the trend of correlated and uncorrelated products. When sales correlation is considered, the risk of worsening an already wrongful prediction is probably smaller.

Using sales correlation in areas of a store where products are placed with high density is interesting. Performances are improved when utilizing sales correlation of nearby products in these areas. Suggesting that using correlation as a feature in forecasting improves the more products are considered "nearby". This should not be too much of a surprise. The more products placed on a shelf, the greater the chances of two products on that shelf sharing similar sales patterns. In this project, "nearby" meant that products were placed on the same gondola. That might be too sharp of a distinguishment, as gondolas and shelves vary in size. This means some gondolas will only contain a few products. Therefore, the sales patterns between these products might not correlate extensively. However, considering the entire store, the model built in this project will not consider how high the highest and low the lowest correlation is. It will only find the highest and lowest sales correlation for each product on a shelf and use that as the feature. It has been shown that the performance can be significantly improved by using sales correlation of nearby products in high-density regions of the store. In some ways, this indicates that sales correlation given by geo-position can be utilized in forecasting, though it requires the correct prerequisites.

It is hard to distinguish if the prediction for one day or two weeks is most indicative of the usefulness of sales correlation. Comparing models to each other might be most fair by only looking at one day. The reasoning is that the train and test set is the same for all models. However, this is not aligned with the actual scope of forecasting. Only being able to forecast one day does not provide much usage. Instead, the interest lies in forecasting sales over more extended periods, for instance, two weeks. However, the predicted data is added and used to predict the coming days for the two-week prediction. Therefore, there might be some flukes involved. Imagine the following case, one day of low sales is followed by three days of unexpectedly high sales. A model that predicts the low sale accurately will be much less likely to predict the top in sales the following days. On the

contrary, a worse prediction in the early part of the period might be better than a good one, for some instances. To fairly judge each model and its performance, the predictions for both one day and two weeks should be considered. The two best-performing models, Extreme Gradient Boosting and Random Forest, indicate that the inclusion of sales correlation of nearby products is a valuable feature for one-day and two-week forecasting.

Another topic to elaborate upon is the amount of data in terms of time series length. First off, how much time is required to decide that the sales pattern of two products correlate? This question does, of course, not have a distinct answer. However, with the amount of data used in this project, it should not be neglected that the sales correlation of nearby products is a coincidence as much as a dependency. As this project is only considering seven weeks, it is essential to raise the question of correlation by coincidence. However, this is probably not an argument against sales correlation being a helpful feature in forecasting. As there are slight improvements to be seen for several models, the performance might be able to increase even more if the correlations could be determined to a greater degree. If one could use a lengthier time series, the correlation between products would be more truthful. That would reasonably be a good thing in terms of improving forecasting performance.

It should, once again, be mentioned that this project considered data from one specific store. Therefore, whether or not the results represent other stores is not possible to conclude. The case could be that environmental circumstances affect sales patterns of products. For instance, a store located in a densely built-up area might differ from one in the countryside. However, for this project, there is no knowledge of sales of groceries varying unpredictably between stores, which implies that the use of sales correlation in forecasting also should be helpful in other stores than the one observed. In addition, circumstances such as store design and the number of products per gondola might affect the usefulness. To give a clear answer about the usefulness, outside and inside store circumstances would have to be considered.

9. Conclusion

The conclusions regarding the usefulness of geo-positioning and correlating sales when forecasting future sales are somewhat two folded. First, there is a tendency that implies that the inclusion of correlating sales in forecasting would improve the performance. This is the case for the best-performing regression models. However, for some models, the inclusion of sales correlation decreases the performance. More extensive studies are needed to determine the usefulness of sales correlation in forecasting. Such studies should include several stores located in different environments and more extended time series of data. Nevertheless, it should be stated that the tendency is that the inclusion of sales correlation has a positive effect on forecasting, even more so when circumstances such as the number of products and high-sales gondolas are favorable. The actual appliance of those findings will depend on which model and what features are currently used to perform forecasting.

Acknowledgement

I want to thank my subject reader Prashant Singh for his guidance throughout this project. Furthermore, thank you to Sabina Westergren Ahlin for a much-appreciated opposition. I also would like to thank the data team at Pricer for providing me with insights during the project. At last, the greatest appreciation to my supervisor Serge de Gosson de Varennes. This project would, in no way, have been the same without your help and knowledge.

References

Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012), Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI)*, *9*(5), 272.

Alibrahim, H. & Ludwig, S.A. (2021), "Hyperparameter Optimization: Comparing Genetic Algorithm against Grid Search and Bayesian Optimization", IEEE, pp. 1551.

Arunraj, N.S. & Ahrens, D. (2015), "A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting", International journal of production economics, vol. 170, pp. 321-335.

Bandara, K., Hewamalage, H., Liu, Y., Kang, Y. & Bergmeir, C. (2021), "Improving the accuracy of global forecasting models using time series data augmentation", Pattern recognition, vol. 120, pp. 108-148.

Bergstra, J., Komer, B., Eliasmith, C., Yamins, D. & Cox, D.D. (2015), "Hyperopt: a Python library for model selection and hyperparameter optimization", Computational science & discovery, vol. 8, no. 1, pp. 14008

Breiman, L. (2001), "Random Forests", Machine learning, vol. 45, no. 1, pp. 5-32

Chen, M. & Lin, C. (2007), "A data mining approach to product assortment and shelf space allocation", Expert systems with applications, vol. 32, no. 4, pp. 976-986.

Chen, T & Guestrin, C. (2016), “XGBoost: A Scalable Tree Boosting System”. Association for Computing Machinery. New York

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012), Random forests. In Ensemble machine learning (pp. 157-175). Springer, Boston, MA.

DeVore, G.R. (2017), "Computing the Z Score and Centiles for Cross-sectional Analysis: A Practical Approach", Journal of ultrasound in medicine, vol. 36, no. 3, pp. 459-473.

Erjavec, J., Gradisar, M. & Trkman, P. (2012), "Assessment of stock size to minimize cutting stock production costs", International journal of production economics, vol. 135, no. 1, pp. 170-176.

Ferber, Robert. (1954), "Sales forecasting by correlation techniques." Journal of Marketing 18.3. 219-232.

Houle, M.E. (2008), "The Relevant-Set Correlation Model for Data Clustering", Statistical analysis and data mining, vol. 1, no. 3, pp. 157-176.

Hu, C., Jain, G., Zhang, P., Schmidt, C., Gomadam, P. & Gorka, T. (2014), "Data-driven method based on particle swarm optimization and k-nearest neighbor regression for estimating capacity of lithium-ion battery", Applied energy, vol. 129, pp. 49-55

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2017), An introduction to statistical learning: with applications in R, Springer, New York.

Li, Hailin, Yenchun, Wu, and Chen. (2020), "Time is money: Dynamic-model-based time series data-mining for correlation analysis of commodity sales." Journal of Computational and Applied Mathematics 370

Mentzer, J.T. & Moon, M.A. (2005), Sales forecasting management: a demand management approach, 2nd;2; edn, Sage Publications, Thousand Oaks, Calif.

Müller, A.C. & Guido, S. (2016), Introduction to machine learning with Python: a guide for data scientists, O'Reilly Media, Sebastopol, California.

Nakamura, T. & Small, M. (2006), "Nonlinear dynamical system identification with dynamic noise and observational noise", Physica. D, vol. 223, no. 1, pp. 54-68.

Patterson, J. & Gibson, A. (2017), Deep learning: a practitioner's approach, O'Reilly Media, Sebastopol, California.

PK, F. A. (1984), What is Artificial Intelligence?. “Success is no accident. It is hard work, perseverance, learning, studying, sacrifice and most of all, love of what you are doing or learning to do”., 65.

Shahzad, U., Al-Noor, N.H., Hanif, M., Sajjad, I. & Muhammad Anas, M. (2020), "Imputation based mean estimators in case of missing data utilizing robust regression and variance–covariance matrices", Communications in statistics. Simulation and computation, , pp. 1-20.

Shaw, S.C., Ntani, G., Baird, J. & Vogel, C.A. (2020), "A systematic review of the influences of food store product placement on dietary-related outcomes", Nutrition reviews, vol. 78, no. 12, pp. 1030-1045.

Wauters M. & Vanhoucke M. (2014), “Support vector machine regression for project control Forecasting”, Autom. Constr. 47 pp. 92–106

Willmott, C. J., & Matsuura, K. (2005), Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate research, 30(1), 79-82.

Yang, M. (2001), "An efficient algorithm to allocate shelf space", European journal of operational research, vol. 131, no. 1, pp. 107-118.

Yang, M. & Chen, W. (1999), "A study on shelf space allocation and management", International journal of production economics, vol. 60, no. 1, pp. 309-317.

Web References

Cloud.Google.com (2022), What is BigQuery?, Tillgänglig online: https://cloud.google.com/bigquery/docs/introduction (2022-03-29)

Geopandas (2021), Tillgänglig online: https://geopandas.org/en/stable/ (2022-04-07)

Hyperopt (2022), What is Hyperopt-sklearn?, Tillgänglig online: https://hyperopt.github.io/hyperopt-sklearn/ (2022-04-07)

Jupyter.org (2022), Jupyter, Tillgänglig online: https://jupyter.org/ (2022-03-20)

Matplotlib (2021), Matplotlib: Visualization with Python, Tillgänglig online: https://matplotlib.org/ (2022-04-07)

Numpy (2022), Tillgänglig online: https://numpy.org/ (2022-04-07)

Pandas (2022), About Pandas, Tillgänglig online: https://pandas.pydata.org/about/ (2022-04-07)

Scikit-learn (2022), Machine Learning in Python, Tillgänglig online: https://scikit-learn.org/stable/ (2022-04-07)

Seaborn (2021), seaborn: statistical data visualization, Tillgänglig online: https://seaborn.pydata.org/ (2022-04-07)

Techtarget.com (2022), Google Cloud Platform, Tillgänglig online: https://www.techtarget.com/searchcloudcomputing/definition/Google-Cloud-Platform (2022-03-30)

Tensorflow (2022), An end-to-end open source machine learning platform, Tillgänglig online: https://www.tensorflow.org/ (2022-04-07)

XGBoost (2021), XGBoost Documentation, Tillgänglig online: https://xgboost.readthedocs.io/en/stable/ (2022-04-07)