ARCADA

Time-series sales forecasting for an Enterprise Resource Planning system

Toni Malila

Master's Thesis

Master of Engineering - Big Data Analytics

2019

MASTER'S THESIS	
Arcada	
Degree Programme:	Big Data Analytics
Identification number:	7261
Author:	Toni Malila
Title:	Time-series sales forecasting for an Enterprise Resource Planning system
Supervisor (Arcada):	Leonardo Espinosa Leal
Commissioned by:	eCraft Oy Ab

Abstract:
Resale businesses and product suppliers rely on enterprise resource planning (ERP) systems to manage their inventory. One of these systems is Microsoft Business Central (BC). BC can create purchase orders for sold products, in order to keep stock at a constant level. The aim of this thesis is to implement a model that can forecast the sales before they happen and thus create the purchase orders before the product runs out of stock or is sold. The dataset used in the thesis is from a product supplier consisting of product sales data for about a thousand different products on a time period of three years. The thesis presents three different forecasting algorithms: autoregressive integrated moving average (ARIMA), long short-term memory (LSTM) and neural networks. The different models were compared using root mean squared error (RMSE) values based on the datasets values against the predicted values. The results show that the best suited model for most of the product sub-datasets is ARIMA. Many of the products however had too little data to be reliably modeled. Further study and development is proposed to be done in testing the presented architecture with another dataset as well as optimizing the LSTM and neural network models.

Keywords:	time-series, ARIMA, eCraft Oy Ab, Enterprise resource planning, Long short-term memory, neural network, machine learning
Number of pages:	36
Language:	English
Date of acceptance:	31.5.2019

CONTENTS

Figures

Tables

Definitions

ERP	Enterprise Resource Planning
BC	Microsoft Business Central
GDPR	General Data Protection Regulation
ML	Machine learning
NN	Neural network
PLS	Partial Least Squares
PCA	Principle Component Analysis
GA	Genetic algorithm
RBF	Radial Basis Function
ARMA	Autoregressive Moving Average
ARIMA	Autoregressive Integrated Moving Average
TAR	Threshold Autoregressive
ARCH	Autoregressive conditional heteroscedastic
sMAPE	Symmetric mean absolute percentage error
ANN	Artificial Neural Network
MSE	Mean squared error
MAD	Mean absolute deviation
MAPE	Mean absolute percentage error
LSTM	Long short-term memory
RNN	Recurrent neural network
PC	Personal computer
RAM	Random access memory
CPU	Central processing unit
GPU	Graphics processing unit
RMSE	Root mean squared error
MAE	Mean absolute error

1 INTRODUCTION

1.1 Background

Many resale businesses and product suppliers today process and handle products in very large quantities. This processing and handling requires a lot of both manual and computational resources. That is why many companies rely on different software solutions such as Enterprise Resource Planning (ERP)-applications. These applications manage many of the core business processes by storing data of these processes in databases and offering a graphical user interface to handle these operations. As such these applications also manage the records of sales and purchase orders and the related processes to these. One example of such a software is Microsoft Business Central, hereafter referred to as "BC". BC manages many aspects of the sales cycle, and one of those is the ordering of sold products to a warehouse in order for the warehouse to be able to ship it to the customer. BC does this by recommending an order list comprised of products and quantities of products already sold by the warehouse in question. This list is then inspected and ordered or cancelled by a user of the software.

This process of ordering products after they have been sold leaves room for inventory shortages and the potential risk of not being able to deliver products to the end customer at all if the supplier is out of stock or has problems with delivering the product. This might also mean that the makes the same but separate order to the supplier with only a short time in between, as the software does not take into consideration future sales at all.

1.2 Proposal

We propose to offer a better solution to the sales cycle by predicting the future sales of products based on past sales data. This will be done by training a forecast model on past sales orders in order to get a model that is capable of predicting the demand of products. This demand will then be reflected on the current warehouse inventory and an order list for purchase orders with product labels and quantities will be generated to keep the

stock at and desired level. This solution will offer better preparedness for seasonal spikes of products as well as reduce the delivery time for the customer from the moment of ordering to delivery as the product is in stock at the warehouse instead of being ordered from the supplier. This prediction of sales may also be used in other business processes of the application as well, such as budgeting and resource management. The predictions should get more accurate as more data is gathered. This data and the predictions can also be used for estimating the demand for new products.

1.3 Dataset

The data for the project is gathered from a BC installation for an undisclosed resales company. The databases in BC are very similar between different installations meaning that the proposed solution should be easily applicable for different customers. The databases are also very well defined and consistent. The data itself is however very varied between products and different customers. The chosen customer for this project has data from a three year period with thousands of different products with varying amounts of sales lines as well as quantities. Some products have consistent sales throughout the data period but others might have gaps in data extending for years. This is to be expected as products might be taken off from the product range and brought back at a later time. The big variance in sales between products will have to be accounted for in the model as a generic model would not accurately represent sales for different products and thus the recommended purchase orders.

1.3.1 Ethical considerations

The dataset used in this project is exported from a Microsoft SQL-database that is created by and managed by the BC-system. As such the physical storage location of the data is configured and managed by BC and is not included in the scope of this project.

The exported data consists of one table of raw sales order data from BC. The data itself contains no personal information or other information that could be linked to individual

citizens nor products of the company that provides the data. All of the product data is linked to physical products by identification numbers that cannot be used to derive the actual product nor other information about the company or the buyer of the products. As such the dataset should be in accordance to the General Data Protection Regulation (GDPR) (European Union).

1.4 Limitations

We will not take into consideration deliveries within the target companies own warehouses and logistic centers but only consider sales to a customer and purchases from the supplier. The logistics process between locations and their respective delivery times and optimizations of these are also not a part of this project. Products in the customers inventory might have different variables such as stock life of the product which are not taken into further consideration in this project. The aim of this project is to predict the sales demand of given products which could be characterized as the optimization of the stock inventory. But further analysis of the lifecycle of stock inventory and its lifecycle will not be carried out.

2 LITERATURE REVIEW

Methods for modeling and forecasting time-series data can roughly be divided into machine learning (ML) methods, such as neural networks (NN), and the more traditional statistical methods (Alon et al 2001 p.147). Of these ML methods are the more recent ones where a lot of development and research effort are put into (Alon et al 2001 p.148). In the following section the methods are presented in their own chapters as well as the combination of ML and statistical methods in so-called hybrid-methods.

2.1 Variable selection

Variable selection can be performed both for statistical and ML methods. The purpose of this is to minimize the model size and training time while at the same time optimizing

the forecasting results. In the study done by Doganis et al. (2006) methods for variable selection such as multivariate analysis with the partial least squares (PLS) (Lohmöller, 2013) and the principle component analysis (PCA) (Müller & Guido 2017 p.140) are mentioned (Doganis et al. 2006 p.197). The drawback for these methods are characterized by the fact that the original linkage to the variables are lost in the process (Doganis et al. 2006 p.197). In the study they used a genetic algorithm (GA) to select the appropriate variables for the model and a radial basis function (RBF) neural network for the forecasting model (Doganis et al. 2006 p.198). Out of the original 14 variables the GA chose 5 which were used to train the RBF model. The study did not compare the results of different variable choices.

2.2 Statistical methods

Many of the traditional statistical models such as autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA) expect that the forecasting behavior is linear (Doganis et al. 2006 p.197). In order to model nonlinearity, models such as the threshold autoregressive (TAR) model and the autoregressive conditional heteroscedastic (ARCH) model can be used. The main drawback of these are that the type of linearity has to be known or discovered and then specified by the modeler (Zhang 2001 p.161).

In a study of forecasting daily milk sales for a one year period Doganis et al. (2006) compared a proposed NN model to some of the aforementioned linear models as well as other NN models. They concluded that the proposed NN model outperformed all of the tested statistical methods. On basis of the results they also concluded that the time series dataset is of mostly nonlinear type (Doganis et al. 2006 p.201).

In another study comprising of 3003 time series the model with the lowest symmetric mean absolute percentage error (sMAPE) out of 7 models was the Theta-method which was not outperformed by any ML models in any of the 18 forecasting horizons that were tested (Makridakis et al 2018 p.3).

G. Peter Zhang (2001) presented in a study for a hybrid ARIMA and artificial neural network (ANN) model the differences between an ARIMA and ANN model using three different time series datasets with previously found nonlinear characteristics. The forecast period of 35 and 67 points ahead were considered. In one of the datasets the ANN model outperformed the ARIMA model in the 35-period horizon while in the 67-period horizon the ARIMA model had better mean squared error (MSE) and mean absolute deviation (MAD) values. The study suggested that neither the ARIMA nor the ANN model could accurately enough model the data pattern (Zhang 2001 p.167).

2.3 Machine learning methods

In contrast to the traditional statistical methods, ML methods are capable of flexible nonlinear modeling without any prior knowledge of the data (Alon et al 2001 p.150).

In a study for forecasting monthly US aggregate retail sales, Alon et al (2001) compared Winters exponential smoothing (Alon et al 2001 p.151), ARIMA model and multivariate regression against an ANN. All of these statistical methods are capable of modeling trend and seasonal fluctuations characterized by the dataset (Alon et al 2001 p.147). The dataset was split into two sets where one is characterized by big fluctuations including two recessions while the other set was taken from a more stable period (Alon et al 2001 p.149). The accuracy of the forecasts were compared via mean absolute percentage error (MAPE) instead MSE since MAPE is not affected by the magnitude of the forecasted values (Alon et al 2001 p.151). The study concluded that the ANN model performed the best across the two periods and that the ANN model outperformed the statistical methods in the first dataset where the fluctuations where bigger. The traditional statistical methods performed well when the dataset was more stable and had less fluctuations (Alon et al 2001 p.154).

In a study on forecasting sales for short shelf-life products Doganis et al (2006) proposed a framework consisting of a RBF neural network for forecasting values combined with a GA to select appropriate variables for the RBF (Doganis et al 2006 p.198). The proposed GA-RBF model outperformed traditional statistical methods and was trained only using historical sales data. The study also concluded that by the forecast can be

improved by introducing adaptation capabilities to the model which would take into account recent data that was not included in the originally trained model. (Doganis et al 2006 p.203).

2.4 Hybrid models

Hybrid models are a combination of statistical methods and ML methods. The objective of hybrid methods is to combine the linear modeling qualities of statistical methods with the nonlinear modeling capabilities of ML methods. The need for such models are apparent in situations where the form of the dataset is unknown or the dataset is prone to structural changes. Hybrid models are also suitable for datasets that are not clearly defined as liner or nonlinear, but instead contain both linear and nonlinear patterns (Zhang et al 2003 p.160).

In a study using three datasets with previously studied nonlinear patterns Zhang compared an ARIMA model, ANN and a hybrid of these two. First a ARIMA model was used to model the linear part. On the remaining data that the ARIMA model could not model a neural network was developed to model the nonlinearity (Zhang 2003 p.165). The results of the study show that the hybrid of these two models outperformed the separate ARIMA and ANN models. This suggests that the hybrid model was able take advantage of the linear modeling capabilities of the ARIMA model as well as the nonlinear capabilities of the ANN model (Zhang 2003 p.171).

2.5 Discussion

The dataset in this project consists of thousands of different products with varying periods and amounts of data. The products might also present different patterns in sales data across the products lifetime. The resulting product of this project should also be easily adaptable to different customers with their own product ranges and sales data presenting yet unknown patterns. Some products might have clear linearity and consistent sales performance while others might present nonlinear sales patterns. Some products might display a combination of periods with clearly linear sales and at other times nonlinear fluctuations. The resulting method should be able to accurately model these variances

between products. Based on these observations a hybrid model combining the linearity of statistical methods and nonlinear capabilities of ML methods is to be considered for further examination.

3 RESEARCH METHODOLOGIES

The time-series dataset used in this project has big variations in both linearity and seasonality between different products. This leads to the hypothesis that a statistical method such as ARIMA might model the data well for some products with linear characteristics while other products might benefit more from a NN approach. This chapter is divided into three parts. The first part focuses on the statistical ARIMA model which might work well for linear product data. In the second part the focus is on a neural network approach which might be better suited for nonlinear products. And in the third part the focus is on an ensemble model of ARIMA and NN which might work best for the big variances in this projects dataset.

3.1 Autoregressive integrated moving average (ARIMA)

ARIMA is one of the most widely used models for forecasting time series. It consists of Autoregressive (AR) and Moving Average (MA) processes that together build an integrated model of the time series (Namin & Namin 2018 p.5). ARIMA models are able to model non-stationary time series but the model assumes that the forecasted values are linear functions of past values (Zhang 2003 p.161).

A ARIMA model has three parameters; p,d,q, where p is the autoregressive (AR) order, d is the differencing and q is the moving average (MA) order (Namin & Namin 2018 p.6). In the building of an ARIMA model the values of p and q are essential. The model can be derived using the three staged Box-Jenkins method (Zhang 2003 p.162). The first step is model identification where the time series data is made stationary. Stationarity in the dataset means that the mean and autocorrelation values in the time series stay constant (Zhang 2003 p.162). The second step is choosing the model parameters

which can be done using maximum likelihood estimation or non-linear least-squares estimation. The third step is model checking where the fit of the model is satisfactory. If the model does not meet the expectations and is inadequate, the process has to be repeated from step one until a satisfactory model is found. This three step process is usually repeated a number of times until a good model is found (Zhang 2003 p.162).

3.2 Artificial Neural Network (ANN)

Artificial neural networks are frameworks for modeling a large variety of non-linear datasets. They are modeled similarly to the neurons of the brain where a number of layers, usually at least three, are connected to each other. The layers contain nodes which are interconnected to each other via acyclic links. These nodes apply an activation function on the inputs from previous layers and either pass through or don't. Lastly an output layer generates probability of outputs based on the hidden layers before. To find the model with the smallest error the error values from the output layer are back propagated into the hidden layers and the weights are adjusted accordingly. The back propagation algorithm can be one of many such as the Levenberg-Marquardt algorithm (Alon et al. 2001 p.151). This is then repeated until the error is at a satisfactory value (Namin & Namin 2018 p.7). Figure 1 below shows an illustrated graph for a neural network with one input layer and node, two hidden layers each with 3 nodes and an output layer with one node.

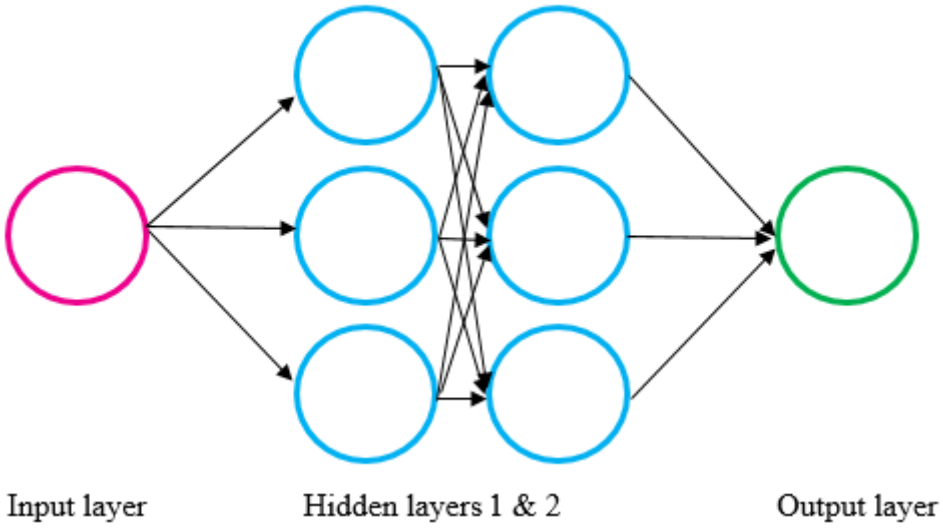

Figure 1. Example of neural network

The model is usually validated with a sample of the data that was not used in the training of the model. In ARIMA on the other hand the whole dataset is usually used for the training and evaluation of the model. This is because of the expectation of linearity in the model which means that the more accurately the historical linearity is modeled the same linearity should be used for forecasting. In an ANN model this approach would most likely overfit the model (Zhang 2003 p.162).

In the training of an ANN model the number of hidden layers are of great importance. The amount of layers to choose is dependent of the data and there is no systematic rule for choosing the parameter. Another important parameter in training a time series ANN is the dimension of the input vector. This parameter determines the nonlinear autocorrelation of the framework. There is no systematic rule for this parameter either which means that the choice of these parameters are often done by experimenting (Zhang 2003 p.164).

3.3 Long short-term memory (LSTM) and recurrent neural network (RNN)

Long short-term memory (LSTM) is a type of recurrent neural network (RNN), which is a type of neural network. Thus it is first needed to explain what an RNN is (Namin & Namin 2018 p.6).

3.3.1 Recurrent neural network (RNN)

A recurrent neural network (RNN) is a type of neural network where the hidden layers act as a type of memory for the following layers. Each element is ran through the same sequence where previously predicted data can be utilized for future data. The memory of layers are restricted and are thus not compatible for long term predictions (Namin & Namin 2018 p.7).

3.3.2 Long short-term memory (LSTM)

Long short-term memory (LSTM) is a type of RNN where the memory of layers are larger and thus better suited for longer term predictions. The model consists of cells where data is stored and processed and after that transported to the next cell. Each cell contains different types of gates that determine if the data is stored, deleted or moved to the next cell (Namin & Namin 2018 p.8).

3.4 Persistent model (naïve)

A persistent model, also called a naïve model, is a model which for value at time t predicts the value t-1. This is used as a baseline model for comparison against the more advanced models.

3.5 Hybrid ARIMA-ANN model

As ARIMA is well suited for linear modeling and ANN for nonlinear modeling and neither are exceptionally suited for the opposite, a hybrid model of these might capture the

linear qualities of the ARIMA model and the nonlinear modeling qualities of the ANN. This might be well suited for real world datasets where the linear characteristics of the data might not be fully known or the data shows characteristics of both linear and non-linear types.

Zhang (2003) proposed a hybrid ARIMA-ANN model by considering a time series dataset to be composed of a linear autocorrelation structure and a nonlinear component. Then by first training an ARIMA model on the dataset it should capture the linear component of the data. While the remaining residuals from the model would then contain the nonlinear component. This nonlinear component would then be modeled using an ANN and thus capturing the nonlinearity of the dataset (Zhang 2003 p.165).

Another proposed ARIMA-ANN hybrid method by Khashei and Bijari in 2010 also assumed that the dataset has a linear and nonlinear component. They first trained an ARIMA model on the dataset and then forecasted one value. The forecasted value together with the original dataset values and the past error sequence were given as inputs to an ANN which then outputted the final forecast. (Babu & Reddy 2014 p.)

3.6 Tools and libraries

The tools and libraries used in this project are common within data science. All modeling and preprocessing was done on a personal computer (PC) using Python and data science related libraries that are explained in the following chapter. The computer used in this project has an Intel i7-7700HQ central processing unit (CPU) and 32GB of random access memory (RAM). The computer also includes a NVIDIA GeForce GTX 1050 graphics processing unit (GPU) that was however not used in this project. The computer runs on a Windows 10 64-bit operating system.

3.6.1 Python

Python is an open-source high-level programming language that is very popular within data science as well as other usages such as web applications. Its popularity within data

science can be seen by the vast amount of different libraries suited for data modeling and processing. Popular libraries for data science such as pandas which provides functionality for data manipulation and analysis, numpy which provides methods for scientific computing and matplotlib that provides plotting of data such as charts. As well as libraries for computing models such as scikit-learn which also includes methods for data manipulation such as scaling and normalizing as well as methods for calculating different error metrics for models and their forecasts. Statsmodels is a library that contains methods for creating different statistical models, such as ARIMA. As well as keras- which is a library for creating neural network models such as LSTM. Training for the neural network and LSTM models where done using the keras library with TensorFlow acting as the backend.

4 IMPLEMENTATION

In this chapter we are going to introduce and explore the dataset used in this project and then go through the process of creating forecasting models for the dataset.

4.1 Dataset

The dataset used in this project is a SQL table consisting of sales rows from an ERP (Enterprise Resource Planning) system for a large company. The dataset date period ranges from 31.12.2014 until 10.10.2018 and consists of 62 table columns and 118558 rows. For the purposes of this project we are going to limit the used columns to “Posting Date” which is a datetime property for the sales row, “Entry Type” which specifies the type of the row i.e. purchase or sales order, “Item No_” which is a integer value specifying the product in question and “Quantity” which is the sales amount for the specific row. The dataset is visualized in table 1 below.

Table 1. Example of dataset

Posting Date	Entry Type	Item No_	Quantity
2018-10-08 00:00:00.000	1	6650	-12

For the purpose of this project we are only using rows where “Entry Type” has a value of 1, indicating that it is a sales order instead of a purchase order. The quantity has a negative value as it is considered as a deduction of stock by the ERP-software, this is regarded in the implementation of the model by simply using the absolute value of the column instead of the actual value.

The distinct sales dates per product vary from 662 to 1 which can be seen in figure 2.

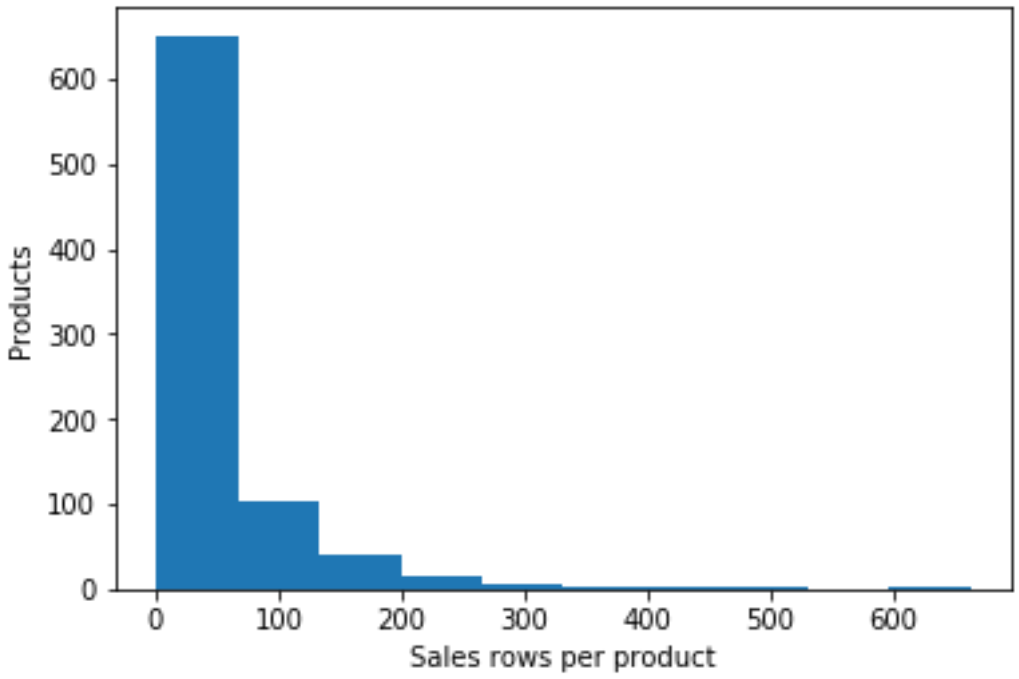

Figure 2. Distinct sales date per product

As can be seen from figure 2, the majority of products have less than 100 sales rows in total. The normalized sales quantities for different products over the span of the datasets time period can be seen in figure 3 below.

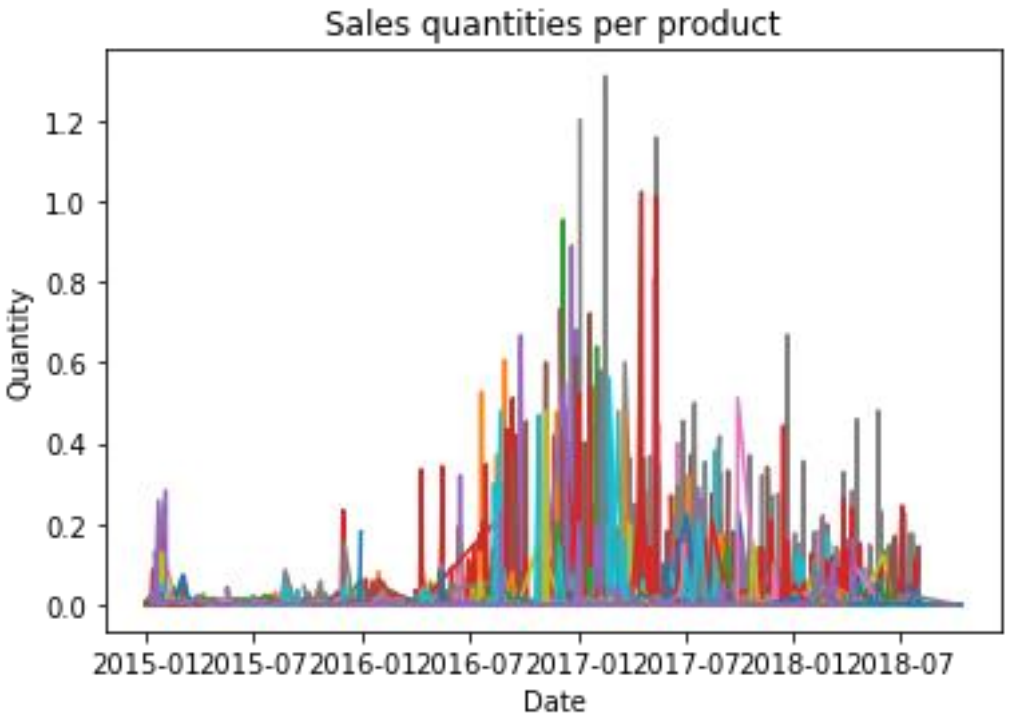

Figure 3. Sales quantities per product during the datasets time period

As can be seen from figure 3 the different products have big variances in the sales quantities as well as in the time span that the product has been sold.

4.2 Implementation

The objective of this project is to create a model that can sufficiently forecast sales amounts for given products.

This is done by grouping the dataset by the product identifier into their own datasets. This results in 825 smaller datasets with varying amounts of data rows. The grouped datasets are then iterated and within each iteration the dataset rows are summed on a weekly time period. This is done in order to get more consistent sales quantity data as well as helping the model learn to forecast on a weekly level instead of daily or time dependent period.

The data is then split into a training and test set where 66% of the data is used for training and 34% is reserved for testing of the trained model. As mentioned in the previous chapter, some products might only have one sales row of historical data. These

products can not be satisfactorly trained into working models as one data point is not enough to provide a good forecasting model. As such datasets with less than 100 rows are exempt from the model creation in order to at least be able to create a training dataset with one row and another test set with the remaining data point. The sales quantities are normalized using the scikit-learn's librarys MinMaxScaler-method (Müller & Guido 2017 p.133) within each product dataset so that the maximum existing value is set to 1 and the smallest value as 0. Each dataset is then iterated and modeled with different models which are explained below.

4.2.1 ARIMA modeling

ARIMA models are trained with all pdq-parameter combinations so that p and d is either 1 or 0 and q is between 0 and 9. A root mean squared error (RMSE) is calculated for each model with the forecasted values and the reserved test set as well as the training set against the models predicted values for the extent of the training set. The models and their respective RMSE for each product dataset are stored in to memory.

4.2.2 Naive modeling

A single persistent model (naive) is modeled and evaluated for each dataset. The modeling is done by shifting the dataset values for every time series point by t+1. The forecast for the model at time t is thus the value at t-1. The RMSE values are calculated for both the test and training set and stored in to memory. The model itself is not stored as it is easily derivable and used only as a baseline to compare the other models performance.

4.2.3 Neural network modeling

A neural network model is trained using the Adam algorithm (Müller & Guido 2017 p.118) with one dense layer with 12 nodes with an rectified linear unit (relu) activation function. The neural network is iteraded for 100 epochs or less if the RMSE loss-function remains constant for more than 2 epochs. The model is evaluated using RMSE

for both the test and train set and are then stored together with the fitted model in to memory.

4.2.4 Long short-term memory modeling

A long short-term memory (LSTM) model with one LSTM-memory cell containing 7 nodes and using the relu activation function is trained for each dataset. The model is iterated for 500 epochs or less if the RMSE loss function stays constant for more than 2 epochs, in the same way as the neural network model. The model is evaluated using RMSE for the test and training dataset and is stored in to memory together with the trained model.

4.2.5 Evaluation metric

When all the models are trained for all of the datasets, a single best model is chosen for each dataset based on the before calculated and stored RMSE which is further explained in chapter 5.
RMSE was chosen as the evaluation metric due to its ability of penalizing large errors and as such it is well suited for the comparison of different models performance. RMSE gives a higher error metric for large outliers, which are undesirable in the case of sales forecasting. Another popular error metric would be the mean absolute error (MAE) which does not penalize large errors in the same way as RMSE, and as such does not reflect possible outliers in the data as well as the RMSE (Chai & Draxler 2014).

This process is well suited for further implementation of different error metrics or implementation of other forecasting models, as the best suited model for each dataset is chosen after all the different methods have been modeled and tested.

5 RESULTS

The dataset was grouped by each different product and the sales rows summed to a weekly period. This means that even datasets that might have two sales rows, one in the beginning time period of the dataset and another in the end, are modeled and tested. As some products have very sparse data such as this, the results for these products might not be very promising. The modeling was only done on datasets which have more than 100 of these weekly sales rows.

Each dataset was modeled with an ARIMA model with pdq-parameters in each combination where p can be a integer between 0 and 9 and d and q variables can be either 0 or 1. Each dataset was also modeled with an simple neural network and long short-term memory(LSTM) algorithms as well as a naive persistence model which works as an baseline. The naive model forecasts each value at time t as the value in time t-1.

From the original 825 grouped datasets 112 datasets were modeled and tested. Each sub dataset was split into an training and test set where 66% of the data was reserved for training and the remaining 33% for validating the models. For each model a root mean squared error was calculated both on the forecasted values the model predicted against the test set as well as the predicted values from the model against the training set. For each dataset a best deemed model is chosen by the smallest rmse on the test set. Figure 4 below illustrates the best deemed algorithm type for all of the models.

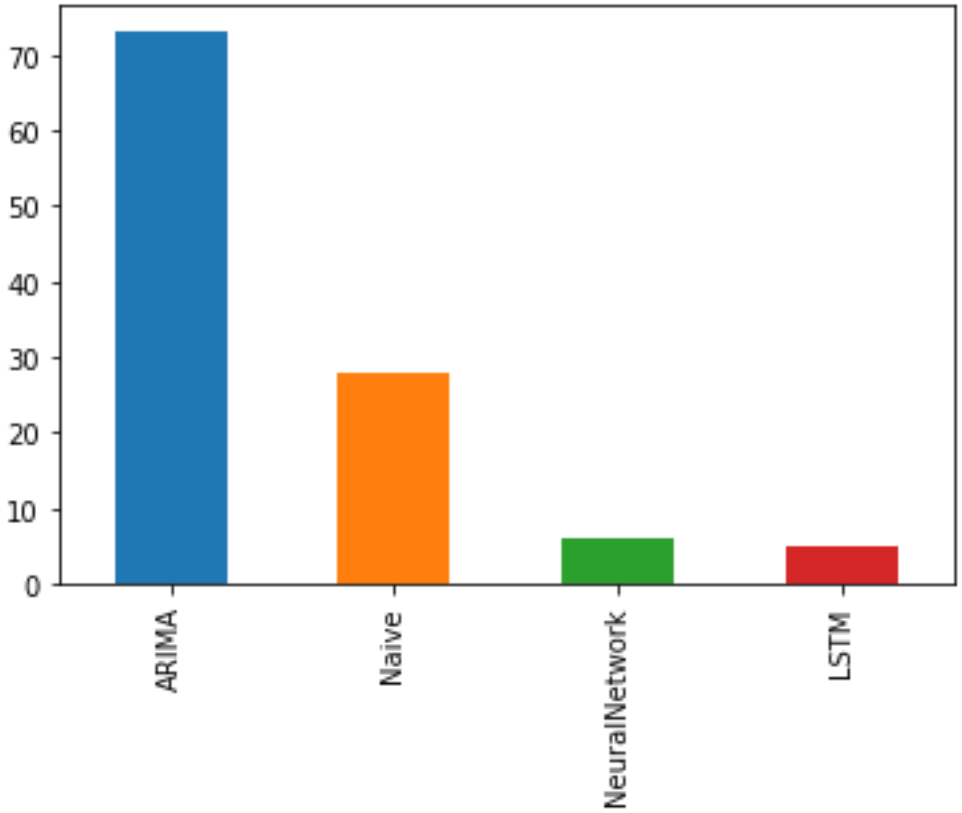

Figure 4. Count of best models by algorithm type based on test forecast RMSE

As can be seen from figure 4 the best performing model in most of the cases was ARIMA with different parameters which was chosen as the best model for 73 datasets.. For 28 of the datasets the naive model was deemed the best model. This suggests that for many datasets the models were not effective at forecasting future values. The neural network was chosen as the best for 6 datasets and LSTM for 5. If the metric for choosing the best model is changed from the test dataset RMSE to training test RMSE instead the models chosen models are different, as can be seen in figure 4 below.

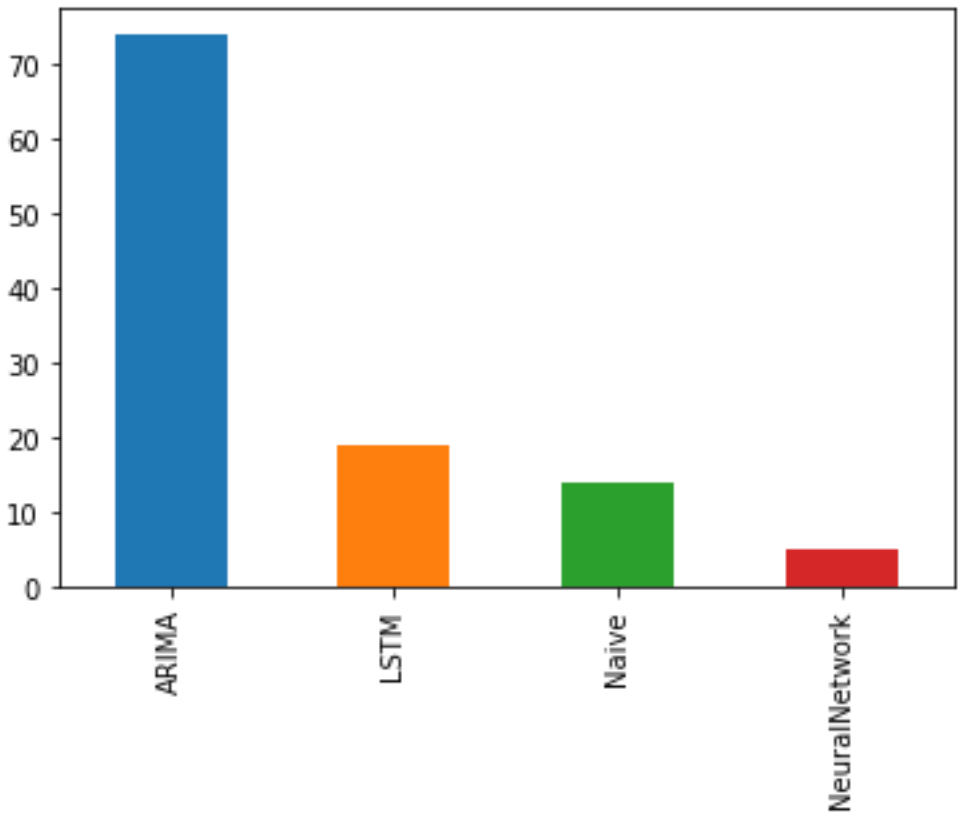

Figure 5. Count of best models by algorithm based on the training RMSE

As can be seen from figure 5 the number of naive models have decreased from 28 to 14 and have been surpassed as the best choice by the LSTM model which has increased from 5 to 19. The ARIMA models have increased by one and the neural network models stayed the same amount. This could suggest that the LSTM model is able to model the characteristics but can not reliably forecast samples out of the training data. This could be because of uncharacteristic data in the test set or a too small dataset to draw reliable conclusions from.

If the choice metric is changed to the sum of both the test dataset RMSE and training dataset RMSE the model choices are similar, as can be seen in figure 6 below.

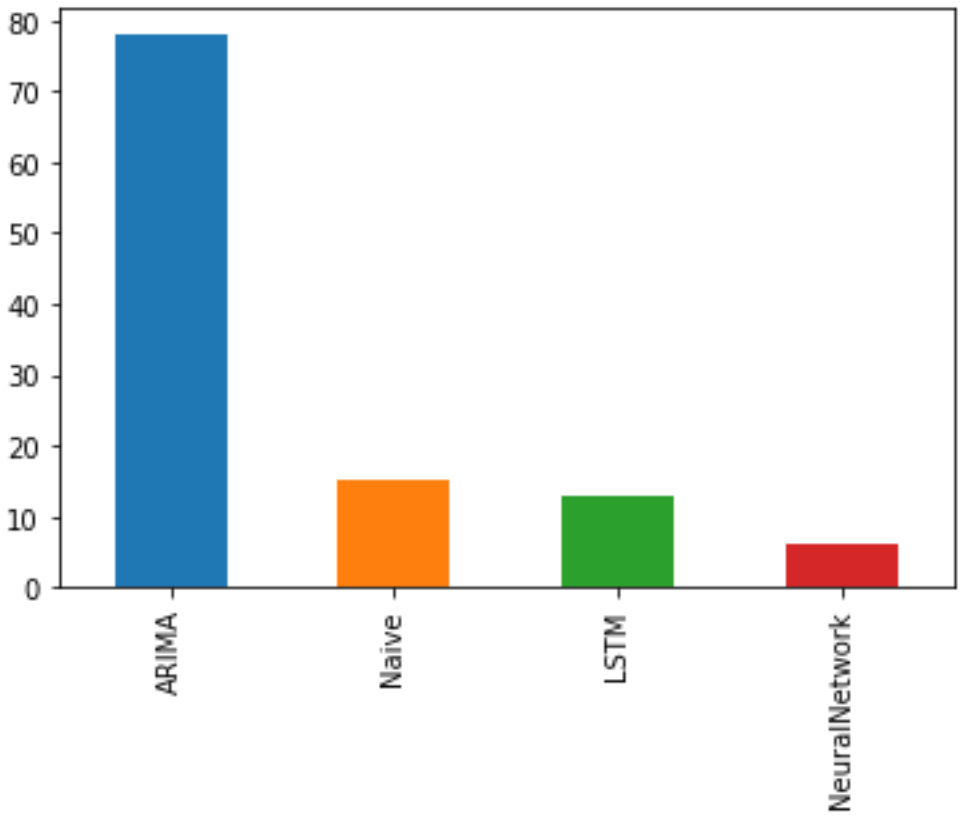

Figure 6. Count of the best models by algorithm based on the test and training sets summed RMSE

The number of naïve models have increased by one from 14 to 15 but the LSTM has decreased from the previous 19 to 13 and thus been surpassed by the naïve models. ARIMA has increased by 4 from 74 to 78 and the neural network models have increased by one from 5 to 6.

As can be seen from figures 4-6, the ARIMA model is the most popular choice regardless of the choice metric. The naive and LSTM models are the second best option, depending on the choice metric. And the neural network model seem to be the least popular model choice. Below in table 2 we can see the mean RMSE values for both the test and training values for each choice metric and model as well as the lowest RMSE value for each column.

Table 2. Mean test and training RMSE for each choice metric

Result models	Test-best test-RMSE mean	Test-best train-RMSE mean	Train-best test-RMSE mean	Train-best train-RMSE mean	Train&test-best test-RMSE mean	Train&test-best train-RMSE mean
ARIMA	0.106400 (1)	0.172604	0.140637 (1)	0.162044	0.117073	0.157877
Naive	0.141057	0.162180 (1)	0.203580	0.102414 (1)	0.178862	0.126405 (1)
LSTM	0.157142	0.186086	0.240079	0.127175	0.103569 (1)	0.145018
Neural network	0.123452	0.180064	0.154322	0.128927	0.124606	0.181119

As we can see from table 2 the ARIMA model had the lowest mean RMSE values in two cases, when the best model was chosen based on the test-set RMSE it had the lowest mean RMSE on the test set and when the model was chosen based on the lowest training-set RMSE it had the lowest mean test-set RMSE. The mean score for the ARIMA models on all the other cases are close to the lowest chosen values in all except for when the models were chosen based on the lowest training-set RMSE. The naïve model had the lowest mean RMSE values in three of the cases show in the table and LSTM in one.

In the following section we are going to display some handpicked product datasets and their associated models and explore the results.

5.1 Dataset 1

The first dataset, hereafter referred to as dataset 1, has 113 values which were split into a training set of 102 samples and a test set of 11 samples. Based on the previously ex-

plained choice of a best model, the best model choice for this dataset was ARIMA, LSTM and the neural network, depending on the choice metric. In figure 7 below can be seen the plotted graph for the dataset and the predictions for each model.

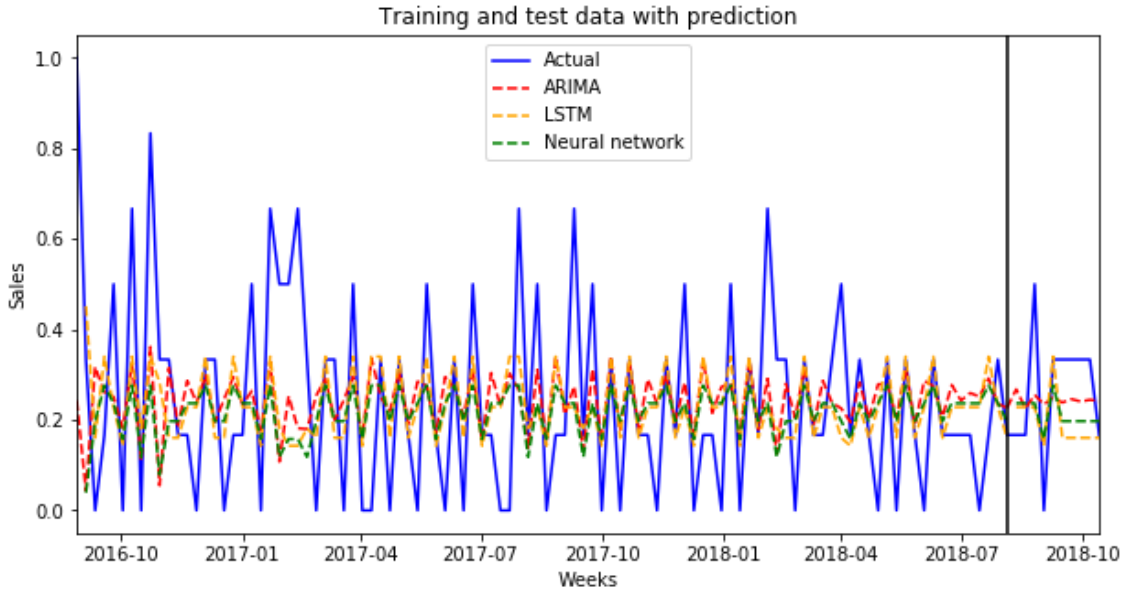

Figure 7. Chart for dataset 1 and its respective model forecasts

As we can see from figure 7 all of the models follow the trend of the dataset but not quite in the right magnitude. All of the different models predictions are very close to each other which might explain why the choice of the best model varies depending on the choice metric. The resulting RMSE values used for the choice can be seen below in table 3. The chart is only plotted for the best performing ARIMA model with pdq parameters: (1,0,1).

Table 3. RMSE values for dataset 1 model predictions

Model	RMSE test	RMSE train	RMSE train + test
ARIMA	0.12714002482188672 (1)	0.2093761101631603	0.336516134985047
LSTM	0.14885396408987933	0.18755067622934243 (1)	0.3364046403192218
Neural network	0.1350264979026197	0.199127640048553	0.33415413795117266 (1)

As seen in table 3 the RMSE values for different parts of the dataset do not have big variances, especially when in column 4 where the RMSE values for the test and training set are summed.

5.2 Dataset 2

In another products dataset, hereafter referred to as dataset 2, the best choice in all three choice metrics were different ARIMA models. Below in figure 8 we can see the actual dataset and forecast for the three different ARIMA models.

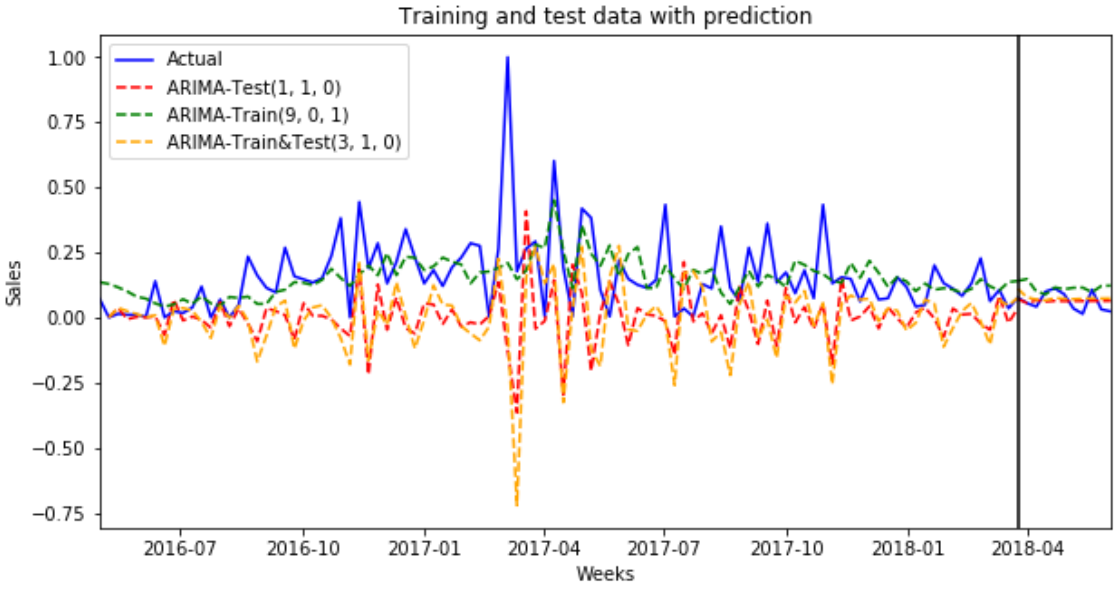

Figure 8. Chart for dataset 2 and its respective model forecasts

As we can see from figure 8 neither the models chosen by the test RMSE and the summed test and training RMSE seem to capture the characteristics of the dataset very well. However in the case of test-model, plotted as a red line in the graph, the test forecast is almost mean of the actual test set and thus the RMSE-value is the lowest of the three models. The train-model, plotted as a green line in the graph, seem to have captured the trend of the training set but performed poorly on the test set by overestimating the values. Below in table 4 we can see the RMSE values associated to the model forecasts.

Table 4. RMSE values for dataset 2 model predictions

Model	RMSE test	RMSE train	RMSE train + test
ARI-MA(1,1,0)	0.034364075288390326 (1)	0.17278687481249524	0.20715095010088558
ARI-MA(9,0,1)	0.06817034775259127	0.13093770549354014 (1)	0.19910805324613146
ARI-MA(3,1,0)	0.03730645993272622	0.14766763943454944	0.18497409936727566 (1)

As seen in table 4 the summed train and test RMSE values for all models are relatively close to each other, as are the RMSE values for the training set of each model. The biggest difference can be seen in the test RMSE values where the ARIMA model with pdq parameters (9,0,1), which was chosen as the best performing based on the training RMSE, is almost doubled compared to the other two models.

5.3 Dataset 3

In another example, referred to as dataset 3, the best model choice for the test metric was an ARIMA model but for the training and summed choice the best model was LSTM. Below in figure 9 we can see the plotted chart for the dataset and different model forecasts.

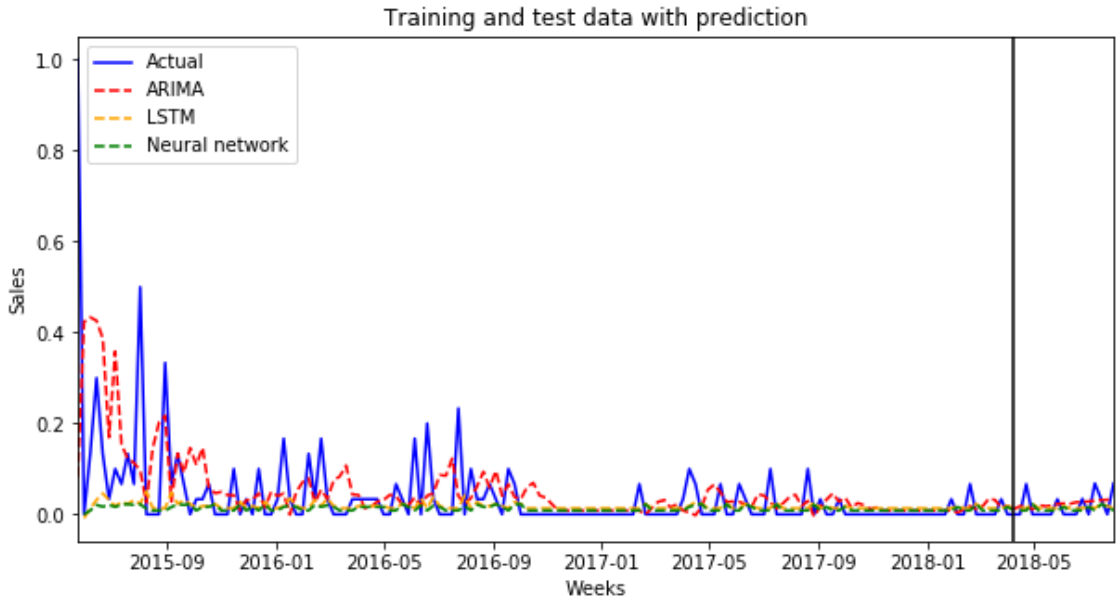

Figure 9. Chart for dataset 3 and its respective model forecasts

As we can see in figure 9 the dataset seems to have some trend where the products do not have sales rows in the start of the year but some sales during the summer and fall months. As well as some variance in the data where in the first year of the dataset the product has significant sales for almost all months but after that in much smaller quantities and different trend characteristic. None of the models seem to have captured the trend of the training data very well. The neural network and LSTM models seem to have modeled very little of the training data and the ARIMA model has more variance but seem to have poor results in doing so. In the test section none of the models seem to have predicted the spikes in sales but the ARIMA model has a slight increase in magnitude which explains the better validation results. Below in table 5 we can see the RMSE values for each model.

Table 5. RMSE values for dataset 3 model predictions

Model	RMSE test	RMSE train	RMSE train + test
ARIMA	0.02489656980084992 (1)	0.11088710600976243	0.13578367581061235
LSTM	0.027774874018382724	0.06981872072491342 (1)	0.09759359474329615 (1)
Neural network	0.02920031924118117	0.07119498639044111	0.10039530563162229

As we can see in table 5 the test RMSE values for each model have only small differences. The more significant changes are in the train and summed values. We can see that the LSTM and neural network models have scored much lower on the training set than the ARIMA model. The same difference can be seen in the summed RMSE values where the LSTM and neural network scored lower than the ARIMA model.

5.4 All models

5.4.1 Error metrics

In table 6 below we can see the mean RMSE values for each algorithm. For the ARIMA models the RMSE mean values are calculated for all of the different ARIMA models with different pdq-parameters as well as a mean for the best performing pdq-values within each product dataset.

Table 6. Combined RMSE values for all algorithms

Model	RMSE test	RMSE train	RMSE train + test
ARIMA (all)	0.172255	0.213035	0.385290
ARIMA (best models)	0.159872 (1)	0.204003	0.369214 (1)
LSTM	0.163662	0.224147	0.387809
Naïve	0.227374	0.197960 (1)	0.425334
Neural network	0.164653	0.222114	0.386767

As we can see from table 6 the ARIMA models where the pdq-parameters were optimized had both the lowest mean RMSE value on the test set as well as in the summed train and test set. The naïve model had the lowest mean RMSE value in the training set, but only marginally better than the optimized ARIMA models.

5.4.2 Training time

The total training and forecasting time for all of the previously presented models was three hours, 56 minutes and 47 seconds. The total time is for the whole process to finish both training and evaluating all of the models and writing them to memory and also includes time used for training models that failed and are not accounted for in table 7. In table 7 below we can see the training time per algorithm.

Table 7. Training time for all models

Model	Total training time (seconds)	Mean training time (seconds)
ARIMA	3.52	0.031
LSTM	7625.15	68.08
Neural network	1822.1	16.27

The training time for ARIMA models includes the training for each dataset with all of the different pdq-paramaters, that is 38 different model parameters and models. Exempt from the training time for the ARIMA models are models that did not converge or otherwise failed training. Naïve models are also exempt from the list since they were not actually trained and instead created by shifting the dataset so that at time t the forecasted value is t-1.

As we can see from table 7 the ARIMA models were the fastest to train by a large margin, even though for each ARIMA model the process actually trained in most cases up to 38 different models. We can also see that the slowest to train was the LSTM and second slowest was the neural network models.

6 CONCLUSION

The dataset used in this project shows how even for a relatively large product supplier company there might not be enough data to reliably forecast sales amounts for all of the products. Even for products with seemingly large amounts of sales rows for a long period of time, the data can have large gaps in between and the sales amounts vary in magnitude.

The project however shows that for products, that have enough sales data, different modeling algorithms and validation metrics show varying results. From the three tested models, excluding the naïve model that was used as a baseline, the ARIMA model seem to have the best results and was therefore chosen as the best suited model in many of the

product datasets. Even when comparing the mean RMSE values for all of the models, the ARIMA model had the lowest RMSE values for the test set as well as the summed values of the train set RMSE and test set RMSE.

From the three described choice metrics, using RMSE values for the test, training and summed test and training sets, the summed RMSE values of both test and train sets might give the best fitting model per dataset. As shown in the results the test dataset might not accurately represent the rest of the dataset due to the limited size of the whole dataset. Thus completely relying on the test sets RMSE might give a less than satisfactory forecasting model. The same principle affects using only the training sets RMSE as a choice metric where a seemingly good fit on the training data might be a result of the lack of enough training data. Thus using the summed values of both the test and training sets RMSE values might give a better representation of the actual characteristics of the dataset.

The other tested models, LSTM and neural networks, did not have seemingly better results in the few cases where they were chosen over the ARIMA models. This suggests that at least for this dataset the modeling could be restricted to only ARIMA models. However as the project is meant to be easily applicable to even other datasets, this might prove to be dependent on the dataset rather than the algorithms shortcomings. As such the modeling could be left as is and a best suited model for each dataset could be any of the three presented models.

6.1 Further research and recommendations

Further development and study could be done in optimizing the number and types of layers for the LSTM and neural network models. This could be done by adding more layers to both the LSTM and neural network models and experimenting with different optimizers. The layer optimization could also be improved by implementing a grid search function that explores the different configurations.

The training time for the LSTM and neural network models could be greatly improved by the usage of a GPU or a cluster of GPU enabled machines. This reduction in training time could be used for training more computationally demanding models as well as optimizing the configurations. The LSTM and neural network models could also be trained as multivariate models with the product identifier as a variable using the whole dataset. The predictions would then be given for a time t and a product identifier combination. This would give the models more training data and given the assumption that the different products might have some correlation in the time dimension the models might perform desirably.

The error metrics for choosing the best model could be extended to include different metrics such as the MAE. The best suited model could then be chosen based on a combination of the different error metrics.

Further study of the already established modeling could also be done on different datasets to see how well the presented architecture performs on never before seen data.

REFERENCES

Alon, I. & Qi, M. & Sadowski, R. 2001, Forecasting aggregate retail sales: a comparison of artificial neural networks and traditional methods. Journal of Retailing and Consumer Services, 8(3), pp.147-156.

Babu, C. & Reddy, B. 2014, A moving-average filter based hybrid ARIMA–ANN model for forecasting time series data. Applied Soft Computing, 23, pp.27-38.

Chai, T. & Draxler, R. 2014, Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. Geoscientific Model Development, 7(3), pp.1247-1250.

Doganis, P. & Alexandridis, A. & Patrinos, P. & Sarimveis, H. 2006, Time series sales forecasting for short shelf-life food products based on artificial neural networks and evolutionary computing. Journal of Food Engineering, 75(2), pp.196-204.

European Union. 2019. Available from https://europa.eu/youreurope/business/dealing-with-customers/data-protection/data-protection-gdpr/index_en.htm Accessed 19.5.2019

Lohmöller, J. 2013, Latent variable path modeling with partial least squares. Springer-Verlag Berlin An, 286 pages.

Makridakis, S. & Spiliotis, E. & Assimakopoulos, V. 2018, Statistical and Machine Learning forecasting methods: Concerns and ways forward. PLOS ONE, 13(3), p.e0194889.

Namin S. & Namin A. 2018, Forecasting economic and financial time series: ARIMA vs. LSTM.

Müller, A. & Guido, S. 2017, Introduction to machine learning with Python, First edition. Sebastopol (CA): O'Reilly, 376 pages.

Zhang, G. (2003). Time series forecasting using a hybrid ARIMA and neural network model. Neurocomputing, 50, pp.159-175.