Comparison of ARIMA, SVR and ANN models including exogenous variables for short-term freight demand forecasting at a large national food distributor

Juliana Schneider

March 8, 2019

1 Introduction

While technological progress in the past decades has led to a significant decrease in relative emissions per tonne-kilometre in truck haulage, this achievement is compensated by an increase in freight transportation (48). In fact, absolute emissions of carbon dioxide rose by 20 % between 1995 and 2017, while freight transportation in general increased by 64% between 1991 and 2016 (49)
Between 2010 and 2030, according to (11), transport volumes for road haulage (in millions of tons) will face an estimated increase of 17% while its transport performance (in billion kilometres) will increase by about 39%. Thus, about 80% and 74% of predicted growth until 2030 for transport volumes and performance respectively will be due to road haulage.

Traditionally, freight transportation demand models are concerned with either vehicle or commodity movements, focusing on origin-destination matrices and geographical scope (e.g. urban, regional, international (38). Accordingly, the question of how many passengers and how much of different kinds of commodities is transported by which type of vehicle on which route is a task of interest for political institutions such as the German Federal Ministry of Transport and Infrastructure (Bundesministerium für Verkehr und Infrastruktur, BMVI) as it is essential for the planning of infrastructure based on transportation demand.(11)
Furthermore, in order to comp with the enormous impact of road haulage on climate, it is a major task for the government to improve and adapt infrastructures as well as strategic planning to transportation demands (10).

However, considering the growing demand for freight transportation, not only politics but also companies share an interest in the modeling of freight transportation demand (47).

What's more, according to (19), freight forecasting can help improve profitability as empty or imbalanced trips are reduced. Furthermore, by planning efficiently, the drivers' quality of life and work-life-balance is enhanced and employee turnover is reduced. This, of course, again reduces a company's expenses, since a high turnover rate imposes costs by the need to constantly hire and train new employees.
Another advantage of sophisticated freight demand forecasting is the enhancement of customer satisfaction through shorter delivery times, higher reliability and adapted pricing.

Unfortunately, little advancement has been made in the forecasting of short-term freight demand for truckage companies so far. This thesis now aims at building a bridge towards the objective of providing a useful, and applicable model to users.
By comparing three different kinds of methods - a rather classic time series approach and two machine learning models - combined with the incorporation of exogenous variables provided by public sources, first advice for future forecasts for this business case shall be given as well as a framework for further research.

After a brief description of the use case at hand, an overview of existing approaches, the theoretical background for variable selection and statistical methods used in this thesis is provided, followed by an introduction to the business case, the truckload company providing the data and a description of these data. Afterwards, the procedure of variable selection, model establishment and model testing is decribed in detail. Finally, the results achieved for this business case are reported and discussed.

2 Use Case: Short-term forecasting of freight weight for road haulage - on the example of NAGEL group

2.1 NAGEL group

2.2 The data

2.3 External data

hier nur kurz beschreiben, dass es externe Daten geben wird.

3 State of th Art in the Modeling of Freight Transportation Demand

NOCH USE OF EXTERNAL DATA BESCHREIBEN

First of all, while browsing literature on the topic of freight transport demand, the sheer amount of different aspects to take into account and consequently the various different quantities under observation in different papers was striking. Common measures in freight demand modeling are: tonne-kilometres, mode choice, freight volume, origin-destination-matrices, and so on.
As the goal of this thesis is to establish a model to predict short-term freight demand for a single user, comparability and therefore scalability of the target variable is not necessary. Hence, the target variable "weight" included in the data provided by NAGEL is not altered, but of course, in further works it might be of use to modify the target variable, e.g. to value/weight-ratios.

As initially mentioned, freight transport modeling has so far been concerned with the so-called four-step transport models consisting of trip generation (number of trips from or to origin), trip distribution (destination of trips from origin), mode choice, and trip assignment (choice of path or route between origin and desitnation), adopted from traditional passenger transport modeling(14; 46).
However, as there are differences in freight demand and passenger demand modelling, adaptations have been proposed by (14). Among other things, the first step in the four-step model - trip generation - should not count the number of trips starting from a certain region of origin, but count the amount of freight (in tonnes) delivered from this origin.
In order to construct an entire freight demand model system, the separate measures of all four steps need to be connected, e.g. through value/weight ratios.
However, the objective of this thesis is not to establish an entire freight model system but it is rather concerned with questions that would belong to step one.
Despite the adaptations proposed by (14), not many other advances have been made. In fact, while passenger transport is a largely studied field due to government regulations, this does not hold true for freight demand (38).
(14) further states that within step one, models so far have only used aggregate data, where time series analysis models, with and without exogenous variables, have already been applied to short-term forecasting.
(38), besides distinguishing between aggregate and disaggregate models, classify into international, intercity and urban freight transportation. Here, at first glance, the category of an intercity model seems suitable, but in this category, freight transportation demand is either modeled as a utility max-

imisation task or an inventory based model for mode choice and production decision, which again is not the scope of this work.

Meanwhile, since, as mentioned above, most literature so far is concerned with aggregate data, within the use case at hand the focus is on the analysis of disaggregate data. Apart from this, further parts of the four step model are not of interest, i.e. neither other parts of step one (e.g. I/O-models) nor other steps, which is why the research on existing literature was expanded to include more specific studies.

(45) used artificial neural networks to predict the demand for weather-sensitive retail products.
(27) predicted container throughput using Dynamic Factor Anaysis and ARIMAX models.
(31) established hybrid ARIMA and ANN models based on DWT decomposition for general time series analysis. (34) used ARIMA to predict full truckload transportation prices.
(35) applied SARIMA models to forecast the demand in a beverage supply chain.
(20) combined grey models, ANN and Support Vector Machines to model full truckload volume. (3) use both univariate and multivariate ARIMA models, examining the impact of new links, new destinations and lower fares on air transport demand at the Port of Reggio Calabria. (29) build a hybrid of ARIMA and SVR for general long term time series prediction. (32) suggest Ensemble Neural Network models to yield more accurate results in time series forecasting.

As is evident, all of the papers mentioned above applied models to either measures similar to but not exactly weight or time series analysis in general. Several authors proposed hybrid models consisting of combined versions of the basic models to be compared further below.
Regardless of the lack of literature exactly matching the target of this thesis, the combined information gained from all of these papers provides reasonable background to be supportive of it. Furthermore, this scarcity in literature proves the necessity of a primary approach to adress the objective of the use case at hand.
Hence, it is important to first retrace the reasoning behind the choice of rather basic versions of AR(I)MAX, SVR and ANN models for the prediction of freight weight demand, which the following section is dedicated to.

4 Model Choice and Specification

There are several ways to model time series data, some of which have been well established for many decades already, while others have recently emerged

in the space of popular methods as well . ARIMA models are an example of the former kind of methods, while SVM and ANN are examples of the latter kind. (1)
(29) state ARIMA, SVM and ANN as the three major methods used in time series analysis.

ANNs are particularly useful as a data-driven method, if there are data but difficulties in describing the process behind them (52). In the case at hand, as there ist not much literature exactly fitting the purpose of my research question, ANNs might be a convenient approach to handle the data.

All three of the aforementioned methods may prove suitable to forecast freight weight, as they each have advantages (see table NUMMER). Furthermore, it might prove insightful to contrast a classical statistical, a machine learning and a deep learning method with each other.

4.1 Model specification

As we shall see, for model specification - i.e. the modelling of the parameters - *Information Criteria* are commonly used. For this reason, the most prevalent of them, the *Akaike Information Criterion* (2) and the *Bayesian Information Criterion* (or *Schwarz Information Criterion*, (44)) are shortly introduced in this section before moving on to descriptions of the specific types models.

$$AIC = 2k - 2ln(\hat{L}) \tag{1}$$

$$BIC = ln(n)k - 2ln(\hat{L}) \tag{2}$$

$\hat{L}$ denotes the Log-Likelihood of the model, k the number of parameters and n the sample size. The AIC's aim is to choose the model that approximates the true data generating process as closely as possible (51). Generally, the model yielding the smallest AIC or BIC respectively is chosen (22). Neither one of the two criteria is superior to the other; AIC tends to overfit the larger the sample size, whereas BIC tends to choose overly simple models in finite sample sizes (22).
Both of them have been used in the three models presented in this coursework. In the following sections, each model's theoretical framework and their associated specification techniques shall be illuminated in more detail. First, time series models are introduced as the constitutive baseline of time series modelling. Then, SVR and NNs are presented.

5 Time series

Forecasting of time series with (S)AR(I)MA models is a well-established concept that has been studied thoroughly for many decades and provides

good forecasting accuracy (5; 31).It has found application in many domains such as economy (MEHR QUELLEN),

The assumption ARIMA models are based on states that the values of a target variable are generated by a linear combination of past values of the same variable and white noise (31), thus making it a stochastic process, "i.e. an ordered sequence of random variables" (3), with data entries at equally distant intervals (25).

A mathematical assumption underlying times series processes is stationarity. (Weak) Stationarity is given when mean and covariance are independent oft time t, and the relationship between two values at time points t and $t + i$ is the same as the relationship between two values at time points s and $s + i$, i.e. independent of the exact position in the time series, but provided the distance between any two values is the same (50).

If $\mu(t) \neq \mu$, i.e. the mean is not independent of time, the assumption of stationarity is violated; there is a so-called trend. In the case of a non-stationary time series, stationarity can be obtained by differencing (3; 25) First-order differencing is denoted as:

$$BY_t = Y_t - Y_{t-1} \tag{3}$$

, where B is the backshift operator (25). If $\sigma^2(t) \neq \sigma$, i.e. the variance is not independent of time, the time series can be logarithmised to obtain stationarity (3). Tests for stationarity include (Augmented) Dickey-Fuller-Test (ADF test) (53) and a visual check of the time series plot.

ADF tests are part of the family of unit roots tests, i.e. the null hypothesis states that the process is non-stationary (53). See (50) for a detailed description of the test metric. (ODER DOCH AUSFÜHRLICHER?)

Another condition of time series processes is **causality**, i.e. future values only depend on current and past values and not on future values themselves (50).

If there are recurrent fluctuations that ocurr yearly or otherwise periodically, there is a seasonal or cyclical component, respectively (50) that has to be taken into account; the time series has to be "deseasonalized" accordingly (8).

The length of the period of a cyclical component may be determined through a fourier series (50). Tests for seasonal components include ??? and visual checks of the time series plots.

5.1 AR models

Autoregressive (AR) processes are processes where a value of a variable at t depends on weighted previous values of the variable itself plus a white noise term $e \sim WN(0, \sigma_\epsilon^2)$. An *AR(p)* process of order p has the form:

$$Y_t = C + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + e_t \tag{4}$$

where C is the intercept.
The order p of an AR process may be determined by a visual check of the plotted Partial Autocorrelation function (PACF). The PACF and the corresponding empirical PACF are denoted in the following form: Considering a time series as a regression where τ denotes the lag:

$$Y_{t+1} = \Phi_{\tau,1}Y_{t+\tau-1}+\Phi_{\tau,2}Y_{t+\tau-2}+...+\Phi_{\tau,\tau-1}Y_{t+1}+\Phi_{\tau,\tau}Y_t+\epsilon_{t+\tau}\text{for } \tau = 1,2,... \tag{5}$$

The PACF is now defined as the regression coefficient $\Phi_{\tau,\tau}$.

$$PACF : \pi(\tau) := \begin{cases} \Phi_{\tau,\tau} & \tau = 1,2,... \\ 1 & \tau = 0 \\ \pi(-\tau) & \tau = -1,-2,... \end{cases} \tag{6}$$

with $-1 < \pi(\tau) < 1$ (50). The empirical PACF can be determined by using Yule-Walker equations or the Durbin-Levinson algorithm (for a detailed description see (50)).

5.2 MA models

Moving Average (MA) processes of order q are denoted like this:

$$Y_t = \mu_{MA} + \epsilon_t - \phi_1\epsilon_{t-1} - \phi_2\epsilon_{t-2} - ... - +\phi_q\epsilon_{t-q} \tag{7}$$

This means the value of a target variable at t depends on a white noise process and previous white noise weighted by ϕ. ϵ in this context are called *innovations*. Furthermore, μ may be 0 and MA-processes are stationary and causal (50). The order q of an MA process may, equal to AR processes, be determined by a visual check of the Autocorrelation Function (ACF)'s plot (correlogram). The ACF and the empirical ACF are denoted as:

$$ACF : \rho(\tau) := \frac{\gamma(\tau)}{\gamma(0)} = \frac{Cov(Y_t, Y_{t+\tau})}{Var(Y_t)}, \tau \tag{8}$$

$$empirical\ ACF : \hat{\rho}_(\tau) := \frac{\hat{\gamma}(\tau)}{\hat{\gamma}(0)} = \frac{\sum_{t=1}^{n-\tau}(y_t - \bar{y})(y_{t+\tau} - \bar{y})}{\sum_{t=1}^{n}(y_t - \bar{y})^2} \tag{9}$$

A condition introduced by (9) is the **invertibility** of the MA process into an AR(∞) process, which ensures that only one parametrization of an MA process can be identified for any ACF.

However, the process underlying time series data may change over time - it is subject to uncertainty (1). A time series model may be biased or overfitted as well as its parameters misspecified.

5.3 AR(I)MA models

As the name suggests, AR(I)MA(p,d,q) - auto-regressive integrated moving average - models model time series data with an AR and an MA component, and trend in data through differencing (which is the "I-part").
The parameters p, d, and q respectively denote the order of the AR component, the degree of differencing and the MA component (53). If there is also a seasonal component - thus a SARIMA model is to be fitted - there are additional parameters P, D, Q referring to the seasonal orders (or degrees) of AR, differencing and MA.

$$Y_t = \mu_{ARMA} + \sum_{i=1}^{p} \Phi_i Y_{t-i} + e_t + \epsilon_t - \sum_{m=1}^{q} \Theta_m \epsilon_{t-m} \tag{10}$$

where μ_{ARMA} actually equals the intercept C of the aforementioned $AR(p)$ model.

5.4 The Box-Jenkins program

In order to iteratively model time series data, (9) proposed a method to identify suitable parameters - AR, MA and differencing - of an ARIMA model. It consists of the following four steps, as described in (17):

5.4.1 Order selection

The orders p and q of the AR and MA have to be determined. This can be done via a visual identification through ACF for MA- and PACF for AR-orders, as mentioned above.
(50) provides an overview for signs of AR, MA and ARMA processes in ACF and PACF plots:

AR(p) processes:
- ACF fades with increasing lag τ, possibly sinusoidally or alternatingly.
- PACF breaks off with lag $\tau > p$.

MA(q) processes:
- ACF breaks off with lag $\tau > q$.
- PACF fades with increasing lag τ, possibly sinusoidally or alternatingly.

ARMA(p,q) processes:
- Both ACF and PACF fade with increasing lag τ, possibly sinusoidally or alternatingly.

(54) recommend settling on an order of both p and q in a magnitude between 0 and 10, while d's most sensible magnitude is between 0 and 2.
For ARMA models, simple plot checks are insufficient and commonly, candidates for p and q are chosen by minimizing an Information Criterion such as the AIC or BIC (17):

$$AIC(p,q) := log(\hat{\sigma}^2_{p,q}) + 2\frac{p+q}{n} \tag{11}$$

$$BIC(p,q) := log(\hat{\sigma}^2_{p,q}) + \frac{(p+q)log(n)}{n} \tag{12}$$

$$\tag{13}$$

5.4.2 Estimation of parameters

Next, the parameters (or coefficients) of the AR and/or MA components of the model are estimated, e.g. by Ordinary Least Squares (OLS) or Maximum Likelihood Estimation (MLE) (3).
To estimate the parameters of an ARMA process, MLE is the most common method (50). However, it assumes a distribution for Y_t - usually a normal distribution.
(50) prefers to switch steps 1 and 2, reasoning that the estimation of the parameters depends largely on the order of the ARMA model. However, admittedly to estimate the parameters, the order of the process have to be known beforehand; although they can be re-adapted later on. For this reason, in this coursework, the procedure according to (9) will be followed.

5.4.3 Model diagnostics

These are mainly residual checks (5). After having fitted the ARIMA-moddels, it is necessary to test the residuals for autocorrelation and normal distribution. Using the Ljung-Box-Test (LB), it is possible to check whether the residuals are autocorrelated, and with the Jarque-Bera-Test (JB), one can check for deviation of the time series from a normal distribution (3). The null hypothesis in the LB-test is that the time series process Y_t consists of independent and identically distributed (*i.i.d.*) random variablest, whereas the null hypothesis in the JB-test states that the random variables Y_t are normally distributed. If the conditions do not hold, the model has to be re-specified as in step 1 (50).

$$LB = N * (N+2) * \sum_{\tau=1}^{m} \frac{(\hat{\rho}(\tau))^2}{N-\tau} \tag{14}$$

$$JB = \frac{N-n_p}{6} * (S^2 + \frac{(K-3)^2}{4}) \tag{15}$$

N is the sample size, τ the order of the lag, m the number of lags considered, $\rho(\tau)$ the autocorrelation function at lag τ, n_p the number of parameters, S the skewness and K the curtosis of the time series. If both tests are statistically significant, then the assumption holds that the error term is a white noise process $e \sim WN(0, \sigma_\epsilon^2)$ (27).

5.4.4 Forecasting

The objective in forecasting is to reduce the expected deviation of the estimated outcome $\hat{Y}_t$ from the actual outcome Y_t of a time series (17):

$$E(Y_t - \hat{Y}_t) \tag{16}$$

To forecast an *AR(p)* model, for $t = 1$, i.e. the first point in time ahead that we have no observed values for, one can simply replace the parameters in equation NUMBER by the parameters obtained in step 2 and insert past values for Y_{t-h}, h = 1, 2, ... For an $AR(2)$ process, the one step ahead forecast looks like this:

$$\hat{Y}_{t+1} = C + \Phi_1 Y_t + \Phi_2 Y_{t-1} \tag{17}$$

In *MA(q)* models, reconsider that the innovations are caused by white noise, and for a forecast of Y_{t+1}, ϵ_{t+1} is directly part of the equation whose expected value is 0. ϵ_{t-1}, so the one step ahead forecast for an $MA(2)$ process looks like this:

$$\hat{Y}_{t+1} = \mu_{MA} - \Theta_1 \epsilon_t - \Theta_2 \epsilon_{t-1} \tag{18}$$

But already with a two step ahead forecast, it is obvious that the $MA(2)$ process (as all $MA(q)$ processes) converges towards the mean μ the farther ahead the forecasting step:

$$\hat{Y}_{t+2} = \mu_{MA} - \Theta_1 \hat{\epsilon}_{t+1} - \Theta_2 \epsilon_t = \mu_{MA} - \Theta_1 * 0 - \Theta_2 \epsilon_t \tag{19}$$

because $E(\epsilon_{t+1}) = 0$. In $ARMA(p.q)$ models, the farther ahead the forecast, the more $Y_{t+\tau}$ converges towards the expected value of the time series $E(Y_{t+\tau}) = \mu_{ARMA}$ *for* $\tau = 0, 1, ...$

5.5 AR(I)MAX

So far, only univariate time series processes, where Y_t is solely predicted by its own past values, $Y_{t-\tau}, \tau =$ 1,2,..., have been considered. Naturally, the question arises whether it is possible to improve predictions based on further explanatory variables (denoted by X or X_t in the following).

Unfortunately, literature on AR(I)MAX models is scarce compared to literature on classic univariate time series analysis.
(27) use ARIMAX to forecast container throughput with additional information of macro-economic indicators at the Port of Koper, whereas (16) forecast macroeconomic time series themselves with both ARIMA and ARIMAX models.
(33) include Ramadan effect in their prediction of sales data.
(4) observed differences in the sales of muslim kids' clothing with Eid holidays every year and modeled this with an ARIMAX model.
(13) compare ARIMAX to SARIMAX modeling in daily traffic counts.

Common AR(I)MA models can be extended to so-called ARMAX models by adding lagged explanatory variables to the model (3):

$$Y_t = \sum_{i=1}^{p} \Phi_i Y_{t-i} + et + \mu_{ARMA} + \epsilon_t - \sum_{m=1}^{q} \Theta_m \epsilon_{t-m} + \gamma X_{t-1} \tag{20}$$

Equation NUMMER can be rewritten in terms of a transfer function model (16):

$$y_t = C + v(B)X_t + e_t \tag{21}$$

where $v(B)X_t$ is the transfer function with backshift operator B:

$$v(B)X_t = \sum_{j=0}^{\infty} v_j B^j X_t \tag{22}$$

AR(I)MAX models can be interpreted as regression problems, where one variable is explained by another so that it can be considered as a linear regression with autocorrelated error terms (13).

6 Artificial Neural Networks

Although seemingly new in the world of statistical tools, Artificial Neural Networks have been invented many decades ago and the enthusiasm for them HAS experienced several revivals throughout the years (20). The idea stems from human brains (52) and the way their neurons learn, store and process information together (6). Although this sounds like a very complex, ambitious venture, NNs are "simply a parameterized non-linear function that can be fitted to data for prediction purposes" (36). In principle, an n-dimensional input is mapped onto an m-dimensional output (39). The process, although very much data-driven (52) and sensitive to the training data (1), is not non-parametric (15).
Nowadays, with the advance of fast and high-performance personal computers, basically anyone can fit a neural network to their data (20).

A great strength of NNs is their ability to detect structures between variables that have not yet been discovered (52).
During these different stages of popularity, ANNs have been developed further. Refined architectures allow the researcher to adapt a neural network to their problem's structure, starting from rather simple one-layer feedforward algorithms to more complex long-short-term-memory multi-layer-perceptron NNs. (QUELLE)
Here, a short overview shall be given before discussing the choice of type of NN for this specific case.

6.1 The general structure of NNs

An NN typically consists of (at least) three layers: One input, one hidden and one output layer. ANNs learn by example (6; 20): Neurons in the input-layer receive the input from the data and pass them on to the hidden layer, where information is processed and then passed to the output layer, which usually consists of one single neuron and is equivalent to the target value. (QUELLE)

Layers - and neurons - are connected by activation, also called transfer functions, which multiply the output received by the preceding neuron by a weight and pass this new output on to the next layer and so on.
Lastly, a learning algorithm is applied to estimate the parameters in the network. When fed some training data, the network's output is compared to the actual output value of the target variable through a loss function. This error is then passed backwards through the layers to adjust the transfer functions' weights.

Multilayer perceptrons are the most popular type of (feedforward) NN (6; 8)

The most common learning algorithm is backpropagation as proposed by (40); others are e.g. Levenberg-Marquardt or

6.2 Network size and structure

A crucial step in modelling NNs is to define the network size and structure, i.e. define the number of layers and nodes (neurons), in order to accurately depict the complexity of the structure of the data and avoid overfitting or underfitting as well as keep computation times low (30).
There is no exact way to determine the number of nodes in each layer, although a lot of previous work has been dedicated to this problem (30). Of course, it depends on the problem at hand.
The number of input neurons may be equal to the number of input variables

in one's problem, but in a time series, the number of input neurons is not evident (52). However, obviously for each exogenous variable added there has to be (at least) one input node added to the NN.
It is already clear at this point that the number of output neurons is one, as we are interested in one continuous target variable and as is recommended by (30).

However, the number of hidden layers and the number of neurons within each hidden layer still has to be determined.
As the objective of this thesis is to present basic models to be contrasted with each other, one hidden layer should suffice. Now, the number of hidden neurons has to be established by trial and error - since there is no clear rule - as is done by most researchers (30). If, with an arbitrary small number of hidden neurons, the network should fail to converge even after multiple restarts, the number of hidden neurons should be increased successively.
To determine a suitable number of hidden nodes, (1) use the BIC:

$$BIC_{p,h} := h(p+2)+1 \tag{23}$$

Due to their ability to approximate any (linear or nonlinear) function underlying the data arbitrarily closely by a nonlinear function, MLPs and RBFNNs classify as universal function approximators (15). They can reduce to an AR[p]-model with an appropriate function that is estimated with a learning algorithm.
Considering the vast number of different types of NNs as well as their general flexibility, it is not a trivial task to choose a suitable model, specify it, and train it (32).

In time series analysis with NNs, Y_{t+1} is the output that is calculated by feeding the network lagged past obervations of $Y_t,\ tfor1,2,...T-1$ and possibly lagged values of exogenous variables (32). For this purpose, the time series is divided into equally sized chunks and fed to the network, where the the next value in the chunk is expected to be determined by the preceding elements of the same chunk (**?**).

(15) counts feedforward ANNs, Jordan ANNs, Elman ANNs (the latter two being types of RNNs), and Multicurrent Networks as most relevant for time series analysis. Feed-forward NNs an Recurrent NNs shall be described in more detailed in the following subsections.

6.3 Activation functions

Activation functions - also known as transfer functions - introduce nonlinearity by mapping the input node to the output node(52). They must be differentiable (39).
Generally, the activation a of a neuron n_i is given by

$$a_i = f(n_i) = f(\sum w_{ij} * x_j) \tag{24}$$

where $f(n_i)$ is the transfer function, w_{ij} is the connection weight between node j and node i, x_j the input signal from the node j, and a_i is the output of the neuron i (8). Among the most popular are sigmoid, hyperbolic tangent, sine or cosine, and linear functions.

$$f(x) = (1 + exp(-x)) - 1 \tag{25}$$

sigmoid or logistic. Sigmoid functions can take values strictly between 0 and 1 and their derivatives are always positive (39).

$$f(x) = (exp(x) - exp(-x))/(exp(x)) + exp(-x)) \tag{26}$$

Hyperbolic tangens (tanh) function can take values between -1 and 1.

$$f(x) = sin(x) \tag{27}$$

or

$$f(x) = cos(x) \tag{28}$$

sine / cosine can take values between -1 and 1.

$$f(x) = x \tag{29}$$

linear can take values between $-\inf$ and inf.

(52) point out that for target variables with continuous values, the use of a linear activation function in the output node is advisable. It must be noted that HIER SEKUNDÄRQUELLE COTTRELL ET AL IN ZHANG 1998 linear activation functions may not be applied on data with trends; hence differencing may be necessary before feeding the data to the network.

Since most papers reviewed for this thesis mention the use of sigmoid or logistic transfer functions in the hidden nodes and linear activation in the output nodes, it seems a reliable modus operandi.

6.4 Learning algorithms

NNs may be trained in various ways. Generally, learning algorithms - also called training algorithms - iteratively minimize an error function to adjust the weights on the nodes (36). Backpropagation as the most widely used algorithm is based on the gradient-descent method (39), which is why differentiability of the activation functions and therefore of the error functions as well is necessary. After a random initialization of the network, the initial weights are adjusted according to the gradient of the error function. The Sum of Squared Errors (SSE) might be considered (39):

$$E = \frac{1}{2} \sum_{i=1}^{p} ||\hat{y}_i - y_i||^2 \tag{30}$$

Another type of error function next to SSE may be Mean Squared Error (MSE) (52), which is defined by SSE/n. Because the training set size is commonly much larger than the test set's, the training MSE will usually underestimate the test MSE (28).
The derivatives of E,

$$\nabla E(w) = \frac{\partial E(w)}{\partial w_1} \frac{\partial E(w)}{\partial w_2}, ..., \frac{\partial E(w)}{\partial w_l} \tag{31}$$

are minimized and the weights iteratively updated:

$$w_i = -\gamma \frac{\partial E(w)}{\partial w_i} for i = 1, ..., l \tag{32}$$

, where γ is the step size, also called learning rate. It determines the extent to which the weights are changed in each iteration and must carefully be specified, since too steep a descent might not converge and a too shallow one might be too slow (52).
In the end, ∇E is expected to be zero. When given input data, the network is fed forward by calculating the input to each node,

$$x_j = \sum_i y_i * w_{ji} \tag{33}$$

, where j indexes output nodes, i indexes the nodes connected to node j, w_{ji} denotes the weight and y_i denotes the antecedent's node's output; outputs in turn are calculated by their activation functions. Furthermore, nodes can be given bias by adding an input with constant weight of 1(40).

Gradient-based learning algorithms can either be conducted in batches or on-line; in the former, all input-output patterns are processed parallely so that the weights are adjusted simultaneously; in the latter, the weights are updated sequentially (39).

The possible flaws of static learning rates have been tried to overcome by expanding the classic gradient descent method with a so-called momentum parameter that controls the direction of the change in weights to not diverge too far from the direction of the antecedent change in weight. (QUELLE)
Backpropagation uses the first derivative of the error function is minimized; other algorithms use the Hessian matrix and thus the second derivative (39):

$$h = -(\nabla^2 E(w))^{-1} \nabla E(w) \tag{34}$$

where w_i is then updated in the following way:

$$w_i = w_{i-1} + h \tag{35}$$

As it is an iterative algorithm and the Hessian matrix has to be computed in each iteration, Quasi-Newton methods make use of a simplification where only the diagonal elements of the Hessian matrix are used:

$$w_i = w_{i-1} + \frac{\nabla_i E(w)}{\partial^2 E(w)/\partial w_i^2} \tag{36}$$

This method is based on the assumption of a quadratic error term. (37) use the Levenberg-Marquardt algorithm, a quasi-Newtonian learning algorithm, to improve learning speed and because it works especially well when there are only few observations; so do (8), (32) and (54). (52) report "faster convergence, robustness and the ability to find good local minima" as key advantages of second-order methods.

As already mentioned, NNs are very sensitive to the data they are trained on; thus, overfitting might occur: the model fits the training data very accurately but performs poorly in predicting new (test) data (36). In order to avoid this issue, so-called early stopping of the training algorithm can be applied, i.e.training is stopped as soon as the training error term (e.g. MSE) starts increasing on a pre-selected validation data set. The flip side of the coin is the problem of NNs being stuck on local minima - independently of the choice of learning algorithm (52) - and therefore being biased; a method of circumvention is the running of multiple epochs, where all steps of training are passed with random initializations of parameters for each epoch. It is avisable to employ a *burn-in*

A third problem that may arise is the *Vanishing Gradient Problem* (43), which a network is more gravely affected by the deeper its architecture. With activation functions that can only take values between 0 and 1, and the cumulative backpropagated error terms that are computed by the chain rule, gradients become "vanishingly" small, as they are multiplied as many times as the network has layers, causing the weights to basically stop changing. If the activation function can take values larger than 1, the gradients may in contrast explode.
As a solution to *Vanishing Gradient Problem*, the Long Short-Term Memory proposed by (24), which will be introduced further below.

6.5 Network architecture

6.5.1 Feed-forward NNs

In feed-forward MLP NNs, an input is fed to the input layer and traversed through the hidden layers without cycles until it reaches the output layer (39). Usually, alls nodes of one layer are connected to all nodes of the next layer, as shown in Picture NUMMER (Rojas p. 134) Picture NUMMER (Benkachcha) shows an MLP feed-forward NN for time series forecasting

where Y_{t+1} is predicted by an input of its past values.

6.5.2 Recurrent NNs

According to (6), RNNs are better suited for time series data than FNNS; in fact, bad results in an RNN may hint that there is not time dependency in the data at all. A feedforward NN in its simplest form reduces to an $AR[p]$-model whereas an RNN may reduce to an $ARMA$-model, which is to be kept in mind when analyzing the data structure at hand.
In RNNs, additional "context units" are added to the model, next to the input layer. These store information from previous training loops. This way, they provide feedback about the past. Context units can either connect (1) the input layer with itself, (2) hidden layers with the input layer or (3)the output layer with the input layer. (2) are called *Elman networks*, whereas (3) are called *Jordan networks*. (6) provides two Pictures illustrating the schemes of these two *Simple Recurrent Networks*: BILDER AUS BALKIN NUMMER

Long Shert-Term Memories (LSTM, (24)) have evolved to become one of the most popular implementations of RNNs (21; Salehinejad et al.), as they can tackle the problem of *vanishing gradients*. In LSTM, hidden units are modified to form *memory cells* that have gates that control (1) input - how much new information is used - and (2) output - or (3) the extent to which information from previous outputs is stored or "forgot". The self-connected *Constant Error Carousels* within the memory cell block use the identity function as their activation function with weight 1; thus its derivative is always 1 so the gradients cannot vanish or explode (43). FIGURE shows an LSTM cell block.

$$g_t^i = \sigma(W_{Ig^i} x_t + W_{Hg^i} h_{t-1} + W_{g^c g^i} g_{t-1}^c + b_{g^i}) \tag{37}$$

$$g_t^f = \sigma(W_{Ig^f} x_t + W_{Hg^f} h_{t-1} + W_{g^c g^f} g_{t-1}^c + b_{g^f}) \tag{38}$$

$$g_t^o = \sigma(W_{Ig^o} x_t + W_{Hg^o} h_{t-1} + W_{g^c g^o} g_{t-1}^c + b_{g^o}) \tag{39}$$

where the matrices $W_{Ig^\bullet}$ is the weight matrix form the input layer, $W_{Hg^\bullet}$ is the weight matrix from the hidden layer and $+W_{g^c g^\bullet}$ is the weight matrix from the cell activation to the input, forget and output gate respectively, and $b_{g^\bullet}$ is the bias (Salehinejad et al.).

7 Variable selection

A well-defined model should include all relevant explanatory variables that contribute to the variance found in the target value and exclude irrelevant, noisy variables. This way, the data and the process constituting it can be

better understood, computational time and the so-called curse of dimensionality can be reduced while maintaining a high prediction accuracy (12). The curse of dimensionality describes the problem arising in a high dimensional (i.e. containing many variables) space, where one needs very large samples to accurately depict said space (QUELLE). Variable selection is furthermore important as irrelevant data may cause poor generalization when given new data (12; 7). Variable selection has become an important issue as research is increasingly focussing on high dimensional data, be it in the field of genetics or Data Mining (7). Fortunately, in this case $n > p$, which is why there is no need to give up on reasonable generalization ability, computational speed or model interpretability in favour of a heavily reduced variable subset.

As with many questions that arise in the process of analyzing data, the question of how to best select input variables in a time series context, and in this case, to predict freight weight, cannot be answered easily. Directly comparing all 2^p possible subsets of variables is computationally unfeasible, so it is most often infeasible to compare all possible variable subsets and choose the best one.
(51) argue that there is not variable that has zero contribution to the target, and therefore - philiosophically - model selection is a comparison between different sets of patterns (i.e. sets of variables) in order to choose which pattern to focus on, depending on the context in which a research question is proposed. Variable selection also takes great part in the variance-bias tradoff, or whether to seek a near-perfect but perhaps overly complex fit of past data or a sparse, economic but perhaps overly simple model. Furthermore, minimal computation time and and a reasonably good prediction accuracy are enhanced by thorough variable selection (41). Obviously, the aim should be to find a golden mean between sparsity and accuracy, too many and too few variables.

A first selection of variables should be given by experts, literature of the same field or common sense (23). This is an important step as is may enhance model performance and facilitate the choice of a suitable subset of variables (7) Next to the researchers' expertise, there exist several variable selection methods that quantify the explanatory value of each possible variable and include or exclude them in the final model by a prespecified selection criterion.

Albeit the growing number of available methods for variable selection, neither is there one gold standard that is generally useful, nor is there one method especially advisable for each kind of data (45)

7.1 Feature Selection Methods in Machine Learning

Feature selection methods, as variable selection methods are called in the context of Machine Learning, can be classified into *Filter*, *Wrapper* and *Em-*

bedded methods (12), .

7.1.1 Filter Methods

Filter Methods are mainly used in text classification and include standard statistical measures such as Euclidian distance, the Chi-Squared test, the t-Test, ANOVA, correlation coefficients, Markov Blankets and the Fisher Score. Adavantages of Filter Methods are their fast computation times, independency of the classification methods - they only make use of intrinsic properties of the variables - and that feature selection only has to be conducted once (41). But, as they test variables seperately "against" the target variable (e.g. for correlation) to rank them, there's a risk that redundant variables are taken into the model when chosen by a Filter Method (12). Consequently, it is possible that the model established by a Filter Method is not unique, as a different subsets of variables with similar properties w.r.t the Filter Method might have yielded an equal result.

7.1.2 Wrapper Methods

Wrapper Methods measure the predictors' performance through a search algorithm. These methods include *Forward* and *Backward Selection* as well as *Heuristic search algorithms* such as *Simulated Annealing* and *Genetic Algorithms*. Advantageously, *Wrapper Methods* take into account the interdependence of variables, but unfortunately require high computation times and are prone to overfitting (41).

7.1.3 Embedded Methods

Both *Wrapper* and *Embedded Methods* are closely intertwined with the classification algorithm (41) In *Embedded Methods*, variable subset selection is part of the model's training (12), and they combine the search for a variable subset and hypothesis testing (41). They are better able to reduce computation time than Wrapper Methods. An example for an Embedded Method are CARTs (Classification And Regression Trees).

7.2 Variable Selection in AR(I)MA, SVR and NNs

(12) mentions Support Vector Machines and Neural Networks as part of *Embedded Methods*. Resulting from this, for these two methods, variable selection (except for a literature- and expert-driven pre-selection) and model fit could be combined in one step. As for AR(I)MA, *Forward* or *Backward Selection* techniques using ICs such as AIC or BIC are favourable because ????.

Other approaches to variable selection in the context of freight demand and / or time series forecasting have been found: In their study on the influence of national, regional and local economic indices on freight volume for different segments of the truckload industry in the USA, (19) first examined correlation matrices before using stepwise multiple regression to draw a selection from the large set of initial variables collected. (45) suggest using ensemble methods such as Random Forest to take advantage of several different machine learning methods. (18) used Spearman rank correlation to select external variables to predict railway freight volume. Variable selection methods for time series data have also been explored, using NNs (QUELLE), SVMs (QUELLE) and other, Filter-like techniques (QUELLE).

Obviously, numerous approaches exist that use different methods and comprise of Filter, Wrapper or Embedded methods. To ensure comparability in this coursework, the results of the AR(I)MA and AR(I)MAX modelling process will be used as the baseline models for SVR and NN. This way, the performance of the AR(I)MA, SVR and NN can be directly compared without being confounded with the use of different subsets of variables for each method. Because the AR(I)MAX model can be specified as a linear regression with autocorrelated errors, it is possible to use a stepwise variable selection comparing AICs of the different specifications. Afterwards, the contribution of the selected variable to the forecast can be compared to the baseline AR(I)MA model using only the target variable.

7.3 Model Evaluation

Generally, data sets are partitioned into training, validation and test data sets in order to estimate parameters (with the training set), fine-tune them during training (on the validation set) and evaluate the model (on the test set). While a validation set is not needed for ARIMA models, it is necessary for NNs (54). Although usually the partitioning is random (in non-time-series data at least) to ensure a good variance of values in the partial data sets, the question arises whether a good (or bad) model performance on the test set is due to the model being well-specified or whether it is not - at least in part - due to the way the data was split. To help eliminate doubts on the accuracy of the model evaluation, *cross-validation* is used. QUELLE In *k-fold cross-validation*, the data set is split into k folds. The model is trained on $k-1$ folds and tested on the one retained fold, each fold in turn being the *"test fold"* once. In time series, it does not make sense to randomly split the data set into groups, since then the model would also be trained on future values attempting to predict past values. Using the idea of cross-validation sensibly, one could partition the data set in such a ways that in the training set, only values preciding the values in the test set are found. A possible setting may look like the one suggested by (26): The data is sorted by timestamps and then trained on the first p values. Then,

the model is tested on the single value $p+1$. Next, the model is trained on the first $p+1$ values and tested on the $p+2$'nd value and so on. This way, model evaluation makes use of all available observations and the results remain comparable since the test set size remains the same. Asymptotically, this technique yields the same results as minimization of the *AIC*.

7.4 Software implementation

Data preparation and analyses were conducted in `R` (QUELLE). The NN was constructed in Python using Keras with Tensorflow backend. A comprehensive list of the packages used in R and in Python can be found in Appendix (NUMMER).

7.4.1 AR(I)MA and AR(I)MAX

To specify and AR(I)MA model, the `arima()` function in `stats` was used. For the AR(I)MAX model, Rob Hyndman's `forecast` package was chosen. Its `Arima()` function fits a linear regression with AR(I)MA errors (QUELLE).

7.4.2 SVR

7.4.3 NN

7.4.4 Variable Selection

As the `Arima()` function in the `forecast` package in `R` uses a linear regression with AR(I)MA errors, variable selection for the exogenous variables in the time series model was conducted using the `step()` function comparing the AIC in the models.

7.4.5 Model evaluation

References

[1] Adhikari, R. (2015). A neural network based linear ensemble framework for time series forecasting. *Neurocomputing*, 157:231–242.

[2] Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723.

[3] Andreoni, A. and Postorino, M. N. (2006). Time meries models to forecast air transport demand: A study about a regional airport. *IFAC Proceedings Volumes*, 39(12):101–106.

[4] Anggraeni, W., Vinarti, R. A., and Kurniawati, Y. D. (2015). Performance comparisons between arima and arimax method in moslem kids

clothes demand forecasting: Case study. *Procedia Computer Science*, 72:630–637.

[5] Arlt, J., Trcka, P., and Arltová, M. (2017). The problem of the sarima model selection for the forecasting purpose. *Statistika*, 97:25–32.

[6] Balkin, S. (1997). *Using Recurrent Neural Networks for Time Series Forecasting*. International Symposium on Forecasting. Barbados.

[7] Ben Ishak, A. (2016). Variable selection using support vector regression and random forests: A comparative study. *Intelligent Data Analysis*, 20(1):83–104.

[8] Benkachcha, S., Benhra, J., and El Hassani, H. (2015). Seasonal time series forecasting models based on artificial neural network. *International Journal of Computer Applications*, 116(20):9–14.

[9] Box, G. E. P. and Jenkins, G. M. (1976). *Time series analysis: Forecasting and control*. Holden-Day series in time series analysis. Holden-Day, San Francisco, Calif., rev. ed. edition.

[10] Bundesministerium für Umwelt, Naturschutz und nukleare Sicherheit (2018). Klimaschutzbericht 2018: zum aktionsprogramm klimaschutz 2020 der bundesregierung: Referentenentwurf.

[11] Bundesministerium für Verkehr und Infrastruktur (2014). Verkehrsverflechtungsprognose 2030.

[12] Chandrashekar, G. and Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28.

[13] Cools, M., Moons, E., and Wets, G. (2009). Investigating the variability in daily traffic counts through use of arimax and sarimax models. *Transportation Research Record: Journal of the Transportation Research Board*, 2136(1):57–66.

[14] de Jong, G., Gunn, H., and Walker, W. (2004). National and international freight transport models: An overview and ideas for future development. *Transport Reviews*, 24(1):103–124.

[15] Dorffner, G. (1996). Neural networks for time series processing. *Neural Network World*, 6:447–468.

[16] Ďurka, P. and Pastoreková, S. (2012). Arima vs. arimax–which approach is better to analyze and forecast macroeconomic time series. *Proceedings of 30th International Conference Mathematical Methods in Economics*.

[17] Falk, M., Marohn, F., Michel, R., Hofmann, D., Macke, M., Sprachmann, C., and Englert, S. (2012). *A First Course on Time Series Analysis: Examples with SAS*. Epubli, Berlin.

[18] Feng, F., Li, W., and Jiang, Q. (2018). Railway freight volume forecast using an ensemble model with optimised deep belief network. *IET Intelligent Transport Systems*, 12(8):851–859.

[19] Fite, J. T., Don Taylor, G., Usher, J. S., English, J. R., and Roberts, J. N. (2002). Forecasting freight demand using economic indices. *International Journal of Physical Distribution & Logistics Management*, 32(4):299–308.

[20] Gao, S., Zhang, Z., and Cao, C. (2011). Road traffic freight volume forecast using support vector machine combining forecasting. *Journal of Software*, 6(9).

[21] George, K., Harish, M., Rao, S., and Murali, K. (2017). Comparison of neural-network learning algorithms for time-series prediction. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI): took place 13-16 September 2017 in Manipal, Mangalore, India, India*, pages 7–13, Piscataway, NJ. IEEE.

[22] Hastie, T., Tibshirani, R., and Friedman, J. H. (2017). *The elements of statistical learning: Data mining, inference, and prediction*. Springer series in statistics. Springer, New York, NY, second edition, corrected at 12th printing 2017 edition.

[23] Heinze, G., Wallisch, C., and Dunkler, D. (2018). Variable selection - a review and recommendations for the practicing statistician. *Biometrical journal. Biometrische Zeitschrift*, 60(3):431–449.

[24] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

[25] Hunt, Ü. (2003). Forecasting of railway freight volume: Approach of estonian railway to arise efficiency. *Transport*, 18.

[26] Hyndman, R. J. and Athanasopoulos, G. (2018). *Forecasting: Principles and practice*. 2. auflage edition.

[27] Intihar, M., Kramberger, T., and Dragan, D. (2017). Container throughput forecasting using dynamic factor analysis and arimax model. *PROMET - Traffic&Transportation*, 29(5):529–542.

[28] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: With Applications in R*, volume 103 of *Springer Texts in Statistics*. Springer, New York, NY.

[29] João F. L. Oliveira and Teresa Bernarda Ludermir (2014). Iterative arima-multiple support vector regression models for long term time series prediction. In *ESANN*.

[30] Karsoliya, S. (2012). Approximating number of hidden layer neurons in multiple hidden layer bpnn architecture. *International Journal of Engineering Trends and Technology*, 3:714–717.

[31] Khandelwal, I., Adhikari, R., and Verma, G. (2015). Time series forecasting using hybrid arima and ann models based on dwt decomposition. *Procedia Computer Science*, 48:173–179.

[32] Kourentzes, N., Barrow, D. K., and Crone, S. F. (2014). Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, 41(9):4235–4244.

[33] Lee, M. H. and Hamzah, N. (2010). Calendar variation model based on arimax for forecasting sales data with ramadhan effect. *Proceedings of the Regional Conference on Statistical Sciences*.

[34] Miller, J. (2018). Arima time series models for full truckload transportation prices. *Forecasting*, 1(1):121–134.

[35] Mircetic, D., Nikolicic, S., Maslaric, M., Ralevic, N., and Debelic, B. (2016). Development of s-arima model for forecasting demand in a beverage supply chain. *Open Engineering*, 6(1):14.

[36] Nakamura, E. (2005). Inflation forecasting using a neural network. *Economics Letters*, 86(3):373–378.

[37] Pinto, R. and Cavalieri, S. (2005). Seasonal time series prediction with artificial neural networks and local measures. *IFAC Proceedings Volumes*, 38(1):337–342.

[38] Regan, A. and Garrido, R. (2001). Modelling freight demand and shipper behaviour: state of the art, future directions. *Travel Behaviour Research*.

[39] Rojas, R. (1996). *Neural Networks: A Systematic Introduction*. Springer, Berlin and Heidelberg.

[40] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

[41] Saeys, Y., Inza, I., and Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics (Oxford, England)*, 23(19):2507–2517.

[Salehinejad et al.] Salehinejad, H., Sankar, S., Barfett, J., Colak, E., and Valaee, S. Recent advances in recurrent neural networks.

[43] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.

[44] Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464.

[45] Taghizadeh, E. (2017). *Utilizing Artificial Neural Networks to Predict Demand for Weather-Sensitive Products at Retail Stores*. Detroit, Michigan, USA.

[46] Transport and Infrastructure Council (2016). Australian transport assessment and planning guidelines: T1 travel demand modelling.

[47] Tsekeris, T. and Tsekeris, C. (2011). Demand forecasting in transport: Overview and modeling advances. *Economic Research-Ekonomska Istraživanja*, 24(1):82–94.

[48] Umweltbundesamt (2018a). Emissionen des verkehrs.

[49] Umweltbundesamt (2018b). Fahrleistungen, verkehrsaufwand und modal split.

[50] Vogel, J. (2015). *Prognose von Zeitreihen: Eine Einführung für Wirtschaftswissenschaftler*. Springer Gabler, Wiesbaden.

[51] Wit, E., van den Heuvel, E., and Romeijn, J.-W. (2012). 'all models are wrong...': an introduction to model uncertainty. *Statistica Neerlandica*, 66(3):217–236.

[52] Zhang, G., Patuwo, E. B., and Hu, M. Y. (1998). Forecasting with artificial neural networks. *International Journal of Forecasting*, 14(1):35–62.

[53] Zhao, J., Cai, J., and Zheng, W. (07022018). Research on railway freight volume prediction based on arima model. In Wang, X., Zhang, Y., Yang, D., and You, Z., editors, *CICTP 2018*, pages 428–437, Reston, VA. American Society of Civil Engineers.

[54] Zhou, L., Heimann, B., and Clausen, U. (2006). Short term demand forecasting for a typical logistics service provider.