Improving Call Center Load Prediction using News

A. C. Verheij

Erasmus School of Economics

Master Thesis Economics & Informatics

Computational Economics programme
Student id 308057
Supervisor Dr. T. P. Tervonen
Co-reader Dr. N. Baştürk

ERASMUS UNIVERSITEIT ROTTERDAM

July 2012

Improving Call Center Load Prediction using News

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTATIONAL ECONOMICS

by

Arnout Verheij

born in The Hague, the Netherlands

Erasmus University Rotterdam
Faculty Erasmus School of Economics
Rotterdam, the Netherlands
www.eur.nl

Syntona
Rouwkooplaan 5
Voorschoten, the Netherlands
www.syntona.nl

“ Predicting the future is hard, especially if it hasn’t happened yet ”

Yogi Berra, famous baseball player also known for his witticisms

 Cover picture: Antony Gormley’s *Quantum Cloud* sculpture in Greenwich, London, representing an outcome of the *random walk model*, which is the most basic version of the models used in this thesis.

Improving Call Center Load Prediction using News

Author: Arnout Verheij
Student id: 308057
Email: verheij.a@gmail.com

Abstract

This thesis evaluates the applicability of news text mining for enhancing daily call center load forecasting for a Dutch flight holiday company called Vliegtickets.nl. A seasonal autoregressive integrated moving average model (SARIMA) is enhanced using news events, that form the external regressors as category-counts before each predicted day based on the dictionary lists of the General Inquirer (GI). Ant colony optimization is used for feature selection. The used dataset contains the daily number of calls of 2009, 2010, and 2011. While news-based regressors do not consistently improve prediction accuracy in terms of RMSE or MAPE, they improve the RMSE slightly for 2010 that contains extreme news-related peaks.

Thesis Committee:

Supervisor: Dr. T. P. Tervonen
Co-reader: Dr. N. Baştürk

Preface

This paper is my master thesis submitted as conclusion of the Master programme Computational Economics following at the study Economics & Informatics at the Erasmus University Rotterdam. It reports on the work I have done during the internship I followed at Syntona, a business intelligence company in Voorschoten, The Netherlands. I am very grateful to all the people who assisted me during this project.

Firstly I would like to thank my two supervisors at Syntona, John Tak and Marnix Beljaars, for giving me the opportunity to do this project for Vliegtickets.nl. Without them I would not have been able to get access to the necessary data for the study. While we initially had some trouble finding a suitable case for this master thesis, they did not give up and finally provided me with this interesting and practically useful research possibility. The offered flexibility on the project is also greatly appreciated.

I also want to express my gratitude to my university supervisors Tommi Tervonen, Viorel Milea, and Nalan Bastürk, who all assisted me in performing my work and suggesting directions of the study. While they are often busy with many different tasks they always managed to find some time to answer my questions.

The assistance from other people is also greatly appreciated, including my friend, volleyball teammate, and colleague Michiel Oudshoorn who got me in touch with Syntona and helped me acquiring the thesis subject, and Gyan Panday from Vliegtickets.nl for providing me the necessary data. My gratitude also extends to my other colleagues Vaishali Urewar and Kirsten Beljaars for the sociability, the laughs we shared, and the many cups of coffee they brought me. Of course my thanks also go to the company Vliegtickets.nl and their directors who granted me access to the data of their call center.

Last, but definitely not least, I would like to warmly thank my parents who intensively supported me during my whole study, both financially and motivationally. It is safe to say that without them completing this study and obtaining my master degree would have been immensely much harder to accomplish.

Arnout Verheij
Voorschoten, the Netherlands
July 10, 2012

Contents

Chapter 1

Introduction

Many companies have call centers as information providing services. In this thesis a call center is defined as a service network in which human agents provide telephone-based services. A common problem with call centers is the forecasting of its load [2, 4, 7]. The size of the operator workforce can be based on the forecasted load. In case of call centers that prioritize sales, insufficient capacity may lead to opportunity costs due to possible customers abandoning the line before being served [44]. The load forecast should lead to a workforce strategy which is a tradeoff between quality (serve as many customers as possible) and efficiency (keep a workforce as small as possible) [41]. For profit-maximizing companies, this tradeoff is usually based on a calculation between the costs of abandonment and the costs of an extra operator.

Traditional decision making models for call centers make use of historical data, and base their decision on parameters like abandon rates, call arrival times, and average call duration [22]. However, these models purely focus on planning the number of operators needed and need a model which accurately predicts the offered traffic of the call center. Basic forecasting methods use day of the week and time of the year as input. However, as these methods only use historical data, they do not capture actual events which might have an influence on the call center load. Especially in certain company types, external events also have an impact [41].

Vliegtickets.nl is a Dutch company providing information about on aspects of flight holidays like flight tickets, hotels, car rental, parking, etcetera. Vliegtickets.nl has a website containing information related to this field, as well as a call center which can be called for any kind of questions or in order to receive a more personal advise. The call center has different lines. Customers entering the call center queue first go through a computer menu determining the purpose of their call. A call can be either a service related call or a sales related call, which are both treated as two different parts of the call center.

The load of the call center depends on the day of the week and the time of the year. The holiday season, for example, is likely to give a relatively high load to the services. Another parameter which is likely to influence the load of the services is the number of recent sales. A percentage of people who buy tickets at Vliegtickets.nl

might end up calling the service part of the call center with questions related to the bought tickets. This might indicate a lagged relationship between the load of the sales part of the call center and the service part. However, basic forecasting models in this application field do not capture external events happening in the world. The load of the service providing call center of a company like Vliegtickets.nl is also affected by such events. Events like the Icelandic volcano eruption causing trouble to all flights above Europe have an influence. In this study I model external events mined from news items taken from the Internet as external regressors in time series models.

The main research question of this thesis is:

- How could the load of call centers be forecasted using events mined from the news?

and sub-questions are:

- Which type or category of events affect the load of the call center of Vliegtickets.nl?

- How could the load of a call center be forecasted using actual events in combination with traditional forecasting methods?

- Does the use of actual events mined from the news improve traditional forecasting methods?

The remainder of this document is organized as follows. Methodologies for call center planning, forecasting, text analytics, and feature selection are discussed in Chapter 2. The study design is introduced in Chapter 3. The actual development of the models is described in Chapter 4, and the results of the study are given in Chapter 5. Finally, the conslusion along with possible directions for future work are presented in Chapter 6.

Chapter 2

Background

This chapter introduces the related work from the fields of call center decision making, load forecasting for call centers, natural language processing, forecasting techniques from other research fields using news events or other external variables, and feature selection algorithms.

2.1 Call Center Decision Making

The most commonly used method to determine the number of operators needed in a call center or a similar service-providing element is Erlang-C. An older method which is not used that often anymore is Erlang-B. Both methods are developed by A. K. Erlang [22] and use statistical variables like call arrival rate and average call-holding time. They are based on the values 'offered traffic' and 'carried traffic'. 'Offered traffic' indicates the total time of incoming calls, while 'carried traffic' indicates the time of answered calls. Both these values are measured in erlangs, dimensionless units representing continuous use of serving circuits. For call centers, one erlang represents 60 minutes of calling by a single operator. Other call center decision making methods are the Engset formula, Erlang-A, and the squareroot rule for safety staffing. The offered traffic E of a call center is defined by

$$E = \lambda * h \ , \tag{2.1}$$

where

$$\begin{aligned} \lambda &= \text{Call arrival rate in calls per minute} \\ h &= \text{Average call-holding time in minutes} \end{aligned}$$

2.1.1 Erlang-B

The Erlang-B method straightforwardly calculates the grade of service based on a given offered load and the number of operators. The grade of service of a call center

is defined as the probability that all operators will be busy when a call attempt is made. Erlang-B assumes an infinite population of sources and that callers who are blocked will not try to call again immediately. The formula of Erlang-B is:

$$P_b(E,m) = \frac{\frac{E^m}{m!}}{\sum_{i=0}^{m} \frac{E^i}{i!}} \quad , \tag{2.2}$$

where

$$\begin{aligned} P_b &= \text{Probability of blocking} \\ E &= \text{Offered traffic} \\ m &= \text{Number of servers} \end{aligned}$$

2.1.2 Extended Erlang-B

The problem with Erlang-B is that it assumes that callers who are blocked will not try to call back immediately. In real situations there is usually a percentage of blocked callers who will directly reattempt to call. This causes a higher load to the system than assumed in Erlang-B. The Extended Erlang-B (EEB) [36] method is an iterative process that repeatedly calculates the offered traffic to the system by using the basic Erlang-B formula and the blocking rate together with a recall factor. After some iterations the offered traffic will become stable, and the original Erlang-B can be used to calculate the grade of service with the given number of servers. A proof of the convergence of these iterations has been published in [77].

2.1.3 Erlang-C

Erlang-C calculates the probability the callers will have to wait before being connected to an operator. Like Erlang-B, it assumes an infinite population of sources. Erlang-C is based on a system with queues, opposed to Erlang-B that assumes callers to be blocked when there are no free operators available. Traditionally the model assumes Poisson arrivals and exponentially distributed service times. The Erlang-C formula is:

$$P_w(E,m) = \frac{\frac{E^m}{m!}\frac{m}{m-E}}{\sum_{i=0}^{m-1} \frac{E^i}{i!} + \frac{E^m}{m!}\frac{m}{m-E}} \quad . \tag{2.3}$$

where

$$P_w = \text{Probability that a customer has to wait}$$

2.1.4 Engset Formula

Erlang-B and Erlang-C both assume an infinite source of callers. In case there is a limited number of sources, the Engset formula [21] is more appropriate. Less than

200 callers is considered a limited number of sources. Just like the Erlang formulas, the Engset formula expresses offered traffic in erlangs. The Engset formula, like EEB, can be iteratively used by updating the offered traffic using the recall factor. Its formula is:

$$P_b(E, m, S) = \frac{E^m \binom{S}{m}}{\sum_{i=0}^{m} E^i \binom{S}{i}} \quad , \tag{2.4}$$

where

$$S \quad = \quad \text{Number of sources of traffic}$$

2.1.5 The Square Root Rule

The square root rule for safety staffing [70] is a simple formula derived from the classic Erlang-C model. It calculates the required number of operators and is given by the following formula:

$$m = E + \beta \sqrt{E} \quad , \tag{2.5}$$

where β is a positive constant that depends on the desired level of service. In practice, the value m should be rounded to an integer because it represents the amount of operators required to meet the desired service level. The second term on the right side of the formula can be described as the excess capacity needed in order to reach the given service level.

2.1.6 Erlang-A

All of the described models have as main shortcoming that they do not treat abandonment of customers who do not feel like waiting in the queue. The Erlang-A model [26] (A for Abandonment) is developed in order to cope with this problem. In this model each caller has an exponentially distributed random variable that corresponds to his patience. If the waiting time exceeds this value, the customer abandons the system. For tractability, the system assumes that callers who abandon do not retry. The square root rule for safety staffing (Equation 2.5) remains valid for this model. The only change to this rule is the formula to calculate the value of β, which now not only depends on the desired grade of service, but also on the abandon rate. In this setting, β might also take a negative value.

2.2 Call Center Load Forecasting

The discussed methods for call center staffing require an estimation of daily call center load. For this, a forecasting method can be used. Straightforward timeseries

models allow to forecast the aggregate daily load of the call center. Other methods try to predict the input variables for the methods described in the last section directly. These methods consist of three parts, predicting the call arrivals, the service times, and the patience of the customer. This subsection first introduces methods used to forecast the daily totals, after which the three parts of the more complicated methods are presented.

2.2.1 Aggregated Daily Load Forecasting

The aggregated daily load of call centers has been forecasted [4] using the autoregressive/integrated/moving average (ARIMA) models [11]. Autoregressive (AR) terms are lagged values of the dependent variable, and hence serve as independent values in the model. Moving average (MA) terms are lagged values of errors made in the past between actual values and their predicted values. These terms try to reduce these errors in future predictions. The term 'integrated' (I) refers to the practice of differencing, which transforms a time series by subtracting past values of itself. An exhaustive list of possible factors that are likely to have an influence on the call volumes is compiled, with for example holidays, catalogue mailings, and new items in catalogue. A cross-correlation method selects the independent variables that show a significant relationship to the daily call volumes. These are used and tested in the final models. An advantage of ARIMA modeling is that its extension SARIMA is able to capture seasonality in the dependent variable. In a similar study it is found that such a multivariate model outperforms univariate models [5]. ARIMA models are discussed in more detail in subsection 2.4.

A more recent study [61] applies multiple models, including seasonal ARMA models, double exponential smoothing methods for seasonality, and dynamic harmonic regression. The models were compared on two sources of data. The results indicate that for practical forecasting horizons, longer than 2 days, the most basic averaging model that simply averages historical variables outperforms all of the mentioned more complex alternatives.

2.2.2 Forecasting Call Arrival Times

Existing studies have specified the following four different properties about call arrivals in a call center:
Property 1: *The total daily number of calls has overdispersion relative to the Poisson distribution (the variance is greater than the mean) [18, 37].*
Property 2: *The arrival rate varies considerably with the time of the day [18, 60].*
Property 3: *There is a strong association between arrival counts within a certain time partition of a day [8].*
Property 4: *There is a significant dependency between arrival counts on successive days [12].*

A Poisson process with a deterministic arrival rate function (the standard nonhomogeneous Poisson process) is inconsistent with both Properties 1 and 3. In order to be consistent with Property 1, a doubly stochastic model was proposed under which arrivals follow a standard Poisson process with a random arrival rate [37]. This rate is modeled as a gamma random variable. Independent versions of the model are estimated for different time periods having *a priori* different arrival rates. Because the different time periods are randomized by independent gamma variables, the correlations between the call arrivals on different time periods of the day are zero.

In order to allow nonzero correlations with a time-varying arrival rate another study [71] proposed a doubly stochastic Poisson process model where the arrival rate function over a day is of the form $\Lambda(t) = Wf(t)$, where W is the only random quantity. It can be interpreted as the unpredictable busyness of a day, while $f(t)$ is simply modeling the normal time-varying arrival intensity.

These models all divide the day into equal time-periods, e.g. 15 or 30 minutes. The arrival rate over such a period is assumed to be a predefined constant for that period multiplied with the constant of the busyness of that day. The basic assumption of this kind of modeling is that the system is in a steady state with a constant arrival rate for certain periods of the day. To avoid this assumption the arrivals can be modeled as a time-inhomogeneous Poisson process, assuming that the arrival function can be well approximated as being piecewise constant. In [12] it is proven that the arrivals are indeed following an inhomogeneous Poisson process.

Another study [57] presents a model which yields not just the forecasted arrival counts, but also their distribution. This study analyses the effect of marketing strategies on call arrivals. The Bayesian analysis of this study is based on the Poisson distribution of arrivals over time periods measured in days with a cumulative rate function. Its conclusion is that the random effects model fits the observed load much better than the fixed effects model.

2.2.3 Forecasting Service Times

As already mentioned, most research is focused on exponentially distributed service times. This is because the different proposed Erlang models would be intractable. The methodology is referred to as the Quality- and Efficiency-Driven (QED) regime, because it has the ability of achieving both very high levels of server-efficiency and customer-service due to economics of scale. In the non-exponential case, the situation gets far more complex than in the exponential case. With exponential service times the state of servers can simply be summarized by stating the number of busy servers. In the non-exponential case, however, the exact state of each individual server must be maintained.

Other studies have indicated that service times are usually not distributed exponentially in practice. Both [12] and [25] find that service times show a remarkable fit to the log-normal distribution. As already described, such distributions are a lot less amenable to analysis than the exponential one.

Basic models all assume that the service center agents are homogeneous. However, in the real-world agents are clearly heterogeneous. In [25] an extensive analysis is presented with heterogeneous agents, modelled through variables as learning effects, agent-by-agent differences, shift fatigue, and system congestion. Simulations show that models not accounting for the agent heterogenity can lead to poor staffing and scheduling decisions.

2.2.4 Forecasting Waiting Times

One of the major problems with estimating the time a customer is willing to wait before reaching an operator is that it is hard to obtain reliable observations of it. The time a customer is willing to wait is referred to as 'customer patience'. The only exact values available of patience of a customer are from the customers who actually abandoned the system. The patience of customers who make it through the waiting line and get served by an operator can only be defined vaguely as 'more than the time needed to come through'. The time needed for a customer to reach an operator is referred to as the 'virtual waiting time' because it amounts to the time he would have to wait if his patience is larger than this time.

Research has shown that in heavily loaded systems where nobody abandons, the waiting time should be exponentially distributed [72]. This theoretical prediction also suits some systems which are not heavily loaded and where customers do abandon the line [12].

Both the virtual waiting time and the customer patience are censored data. This means that the observations of these data are only partially known, as described above. Because patience and virtual waiting times can be assumed to be independent given the covariates relevant to the individual customer, both distributions can be estimated and plotted as survival functions using the standard Kaplan-Meier product-limit estimator [12].

Another way of modeling impatience is using a hazard rate [12, 48]. The estimate of the hazard rate (H) for each interval of length δ is calculated as given in Equation 2.6. Plots of these data usually show relatively high hazard rates for a low amount and lower rates for a higher amount of time passed. This can be explained by the fact that callers who do not want to wait at all directly abandon when the first 'please wait' message is played for the first time. Another explaining effect might be that the longer someone is waiting, the lower the chance he would abandon due to the fact that he would have waited all that time for no result.

$$H = \frac{\text{number of events during}(t, t+\delta)}{(\text{number at risk at } t) * \delta} \quad . \tag{2.6}$$

2.3 Forecasting Models using External Variables

2.3.1 Models using Text Analytics

While there is no literature available on models that try to predict call center load using news, there exists literature in other fields that applies news or other items consisting of natural language for forecasting purposes. Some studies which develop prediction models based on natural language are discussed.

The authors of [17] try to predict stock value changes using sentiments from small talk on the Web. A sentiment is defined as the net positive and negative opinions expressed about a stock on its message board. Sentiment analysis from the small talk on a stock message board might be used to forecast stock values. Five different classifiers were used in order to determine whether a message is bullish (optimistic), bearish (pessimistic), or neutral (either spam or messages that are neither bullish nor bearish). The classifiers were mainly based on counting positive and negative words. The final classification uses a simple voting scheme that requires a majority over five algorithms. The net sentiment of all small talk about a stock is used to predict whether the value of the stock will increase, decrease or stay constant.

Another study [45] investigates whether the MSCI EURO index can be predicted based on the European Central Bank statements. This study defines a list of adjective categories with, for example, positive, negative, strong, and weak adjectives. All adjectives from the ECB statements are fed to the General Inquirer with this list, yielding a matrix of percentages that indicate for each document the percentage of words that fall under each category. Consequently, a Takagi-Sugeno fuzzy model [59, 76] is made based on these parameters. The system obtains an accuracy of 66% on average on the test set, which corresponds to a 16% increase over a random classifier.

In [49] the exchange rate is forecasted using news headlines. Where traditional exchange rate models are based on statistical analysis, this study uses the rich content of news headlines instead, giving not only better prediction possibilities, but also extra information about the reasons behind the prediction. The study develops a rule-based classifier using manually determined keywords from the headlines. Instead of single keywords, the study uses world tuples. The technique from [73] is used to determine the probabilistic decision rules based on these keywords. Simple examples of such rules from other studies are as follows:

$$competitive(x) \leftarrow int(x), pro(x) \tag{2.7}$$

$$competitive(x) \leftarrow int(x), rel(x) \tag{2.8}$$

This rule expresses that company x is competitive if it is intelligently managed (*int*) and professionally competent (*pro*) or if it is intelligently managed and reliable(*rel*). Different probabilistic rules are connected by ‘or’. The attributes of a rule can be weighted, meaning that for example the value of *int* does not have to be either true

(1) or false (0), but it can be anything in between. The value to be predicted (competitiveness of a company in this case) is calculated using the common probabilistic method (in case of the given example rules $int(x) * pro(x) + int(x) * rel(x) - int(x) * pro(x) * rel(x)$). The rules are generated based on a training set using a simple heuristic. All candidate rules containing only one attribute in their bodies are evaluated. The best performing one is selected for specialization, meaning another attribute to be added to this rule. This process is repeated until the performance of the rules does not increase anymore. The method saves the obtained rule and starts the process again using the remaining attributes. The process stops when the specified number of rules is generated. The discussed exchange rate prediction system yields better results than conventional numerical time series analysis and is significantly better than random guessing for the used data set.

In [62] a popular Wall Street Journal column is used to determine a 'pessimism media factor' in order to predict the stock market. Words from the General Inquirer (GI) categories are filtered from the column and a Principal Component Analysis determines which categories should be used in the factor. The study finds that high values of media pessimism indicate downward pressure on market prices, and unusually high or low values of pessimism indicate high market trading volume.

Another study [24] tries to align news articles to trends in stock markets using a clustering algorithm for news items based on their content. Different clusters are created for news items appearing during a rise or a drop in the trend. The trend is captured by regression lines established with hypothesis testing that decides whether the time series should be split. Another algorithm selects the terms from the documents that are likely to account for the trend. The incremental K-Means algorithm creates two clusters for both trends (Rise/Drop) based on the terms in the items, resulting in a total of 4 clusters. The cluster most similar to the clusters aligned to the other trend is discarded for both trends, resulting in one cluster of news items for each trend.

After this, the inter-cluster discrimination and the intra-cluster similarity coefficient are used in order to determine which terms from the clusters account for the trend. These coefficients are multiplied with each other to obtain a weight indicating the degree for which the feature appears in one of the clusters but not in the other, and hence which terms are most capable of differentiating between the trends. After this training procedure the final term classification is made using classification with Support Vector Machines [67] using the obtained weights.

Other studies try to quantify market behaviour using sentiment in news items [1, 27, 46]. The first study [1] shows that there is a certain agreement between periods of high-volatility in stock market indices and the volatility of 'bad' news during the year. It uses the two categories, Positive and Negative words from the Harvard Dictionary of Affect, to check the volatility of positive and negative sentiment in news items. The study uses the Irish Times archive filtered on three keywords, 'Ireland'/'Irish' and 'Economy', as news corpus. It finds some dependency between the obtained volatility and the time series of the stock market. However, it does not yet use these findings to predict or compute time series, risks, or other parameters.

The second mentioned study based on sentiment analysis [27] automatically la-

bels financial news items to a rising or a falling market index. It uses different kinds of features, including amongst others unigrams (nouns, verbs, adjectives and adverbs that appear at least three times in the corpus), stems (unigrams which have been stripped of their morphological variants), and pre-defined financial terms. The selection of specific features within the different categories is done with three different methods, Document Frequency calculating the number of documents in which the term occurs, Information Gain calculating the specificness of a term, and the χ^2 method measuring the lack of independence between terms. The methods all select the 100 most differentiating features from the corpus. News items are then automatically labelled as positive or negative based on the difference between the price of a stock at the closing of the market before the item was published and the price of the stock at the opening of the market after the item was published. All news items are linked to the specific companies they provide information about. Features are classified as negative or positive using Support Vector Machines based on the labelled news items they occur in. The study finds that the highest classification accuracy is obtained using unigrams as features and Information Gain as the feature selection method. A similar approach is used in NewsCATS [46], which categorizes the news corpus in 'Good News', 'Bad News', and 'No Movers'.

While all of the so far discussed models only predict a limited number of classes, like whether the trend will go up, down, or stay steady, [32] tries to predict the exact interest rate using both the trend and news information from the Internet. A cognitive map (CM) is made to represent expert knowledge. The CM consists of terms, decided by experts, that influence interest rates. The study develops a Knowledge-Based News Miner (KBNMiner), which searches and retrieves news information on the Internet according to this prior knowledge. This information, together with trend information like average exchange rates and corporate bond yields in the past days, is applied to a neural network [33], which then predicts the interest rate. The study shows that neural networks perform better on this prediction task than the random walk model, and that a neural network with event information performs better than a neural network without event information.

A recent trend in forecasting using text analytics is to build models based on the substantial amount of shared knowledge and information included in social media. An example of this is the use of *tweets* (periodic status updates on the social network site *Twitter.com* submitted by users) for predicting first week box-office revenues of movies [6]. This study constructs a linear regression model based on the rate of tweets per hour referring to a particular movie. The model yields better prediction results than older models using the Hollywood Stock Exchange. In addition, the study finds a strong predictive relationship between the ratio of positive and negative sentiments in tweets and box-office revenues after the movie release.

2.3.2 Other Models

Another interesting forecasting model, although not using text analytics, is developed in [65]. This study uses a combination of a neural network back propagation model (BP) and the SARIMA model. This SARIMABP model predicts the seasonal time series of production values. It first runs the basic SARIMA model, and uses these predictions and the residuals as inputs for the neural network. The study shows that the combination of these models perform better than either of them alone.

In [13] another forecasting model using external variables is used, namely the ARIMAX model. This study tries to predict the number of deceased people due to traffic accidents. The ARIMAX model combines the basic ARIMA model with external explanatory values. In the case of this study, examples of possible explanatory variables are total population, population of drivers, and the number of passenger transport. The next subsection describes ARIMAX models in more detail.

2.4 Timeseries Models

One of the most commonly used models for time series prediction is the Autoregressive Integrated Moving Average (ARIMA) model [11]. It is also called the Box-Jenkins model after the methodology designed by George Box and Gwilym Jenkins for estimating parameters of such models. Time series datasets are often autocorrelated, meaning that subsequent data points within the series are correlated, indicating presence of repeating patterns. The ARIMA model is a combination of the Autoregressive (AR) and Moving Average (MA) models. The term 'Integrated' stands for an order of differencing, which is needed for non-stationary time series (i.e. series with a trend).

2.4.1 ARIMA Model Definitions

The notation of an autoregressive model is AR(p), with p indicating the order of the model. It tries to predict a future value based on the previous outputs. Common values for p are 1 and 2. Essentially the model is a combination of the weighted past p values of X, where X is the variable to be predicted. The AR(1) model comes down to $X_t = X_{t-1} + \epsilon_t$, of which the random walk model is a limiting distribution. The random walk model is the mathematical formalisation of a path consisting of random steps. For higher order AR models certain stationary restrictions are needed in parameters ϕ_i. The AR model is given in the following equation:

$$X_t = c + \sum_{i=1}^{p} \phi_i X_{t-i} + \epsilon_t \ , \tag{2.9}$$

where

$$\begin{aligned} X_t &= \text{The predicted value} \\ c &= \text{A constant (often omitted)} \\ \phi_i &= \text{Parameters of the model} \\ \epsilon_t &= \text{White noise error terms} \end{aligned}$$

The notation of the moving average model of order q is MA(q). Commonly used values of q are 1 and 2. The essence of the MA model is to constantly adjust the model based on previous error terms. Depending on the nature of the data, either AR, MA, or the combination of both models provide the best forecasting results. Notice there are also stationary restrictions for the MA parameters. The following equation shows the MA model:

$$X_t = \mu + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} + \epsilon_t \ , \tag{2.10}$$

where

$$\begin{aligned} \mu &= \text{Expectation of } X_t \\ \theta_i &= \text{Parameters of the model} \end{aligned}$$

The combination of the AR and MA models leads to the ARMA(p, q) model, given in Equation 2.11. This model has p autoregressive terms and q moving average terms. In the ARMA model, μ, the MA model's expectation of X_t is replaced by the AR model. The error terms in the models are generally assumed to be independent and identically distributed random variables.

$$X_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \ . \tag{2.11}$$

A generalization of the ARMA model is the ARIMA model, which is notated as ARIMA(p, d, q). It includes p AR terms, q MA terms, and d specifies the order of differencing, which is only used for time series data which are not stationary. One order of differencing removes a linear trend, while $d = 2$ removes a quadratic trend.

$$\Delta^d X_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i \Delta^d X_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \ . \tag{2.12}$$

2.4.2 Seasonal ARIMA

Some time series display strong seasonal patterns. Call centers for example might yield less calls during weekends than weekdays. Depending on the call center, the load can be consistently higher or lower on certain days of the week, month, or year. Such time series require a seasonal ARIMA (SARIMA) model for sound prediction

performance. The notation of the SARIMA model is SARIMA$(p, d, q)(ps, ds, qs)$, with ps seasonal AR terms, qs seasonal MA terms, and an order of seasonal differencing ds.

Seasonal ARIMA models function similar to the non-seasonal ARIMA models, while utilizing a predefined seasonal lag. While normal differencing measures the difference between each successive value of X_t, seasonal differencing measures the changes between each successive season. For example, if the data is weekly, the seasonal difference of X at time t is measured as $X_t - X_{t-7}$. Likewise, seasonal AR and MA terms are treated equally to non-seasonal AR and MA terms including a lag equal to the period of the season.

2.4.3 ARMA with Exogenous Variables

It rarely happens that time series are perfectly modeled using only (seasonal) ARIMA models. Many time series are influenced by more factors than can be modeled based on only the past values. These external "shocks" can be incorporated in the model using exogenous input terms resulting in the ARMA model with exogenous or control variables, notated as ARMAX(p, q, b). It consists of p AR terms, q MA terms, and b exogenous input terms. The model is defined in Equation 2.13. It contains the already described AR and MA models, and includes a linear combination of the last b terms of the used external time series.

$$X_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} + \sum_{i=1}^{b} \eta_i d_{t-i} \ , \tag{2.13}$$

where

$$\begin{aligned} \eta_i &= \text{Parameters of exogenous input} \\ d &= \text{Exogenous input} \end{aligned}$$

2.4.4 Box-Jenkins Model Estimation

The Box-Jenkins methodology is often used in order to find the best fitting SARIMA model. This approach consists of three iterative phases: model identification, parameter estimation, and model validation.

Model Identification

The first step of SARIMA model identification involves determining the order of differencing to make sure the time series is stationary. A time series is not stationary if it contains a trend. Stationarity can be detected through autocorrelation function (ACF) [30] plots, that plot autocorrelations of subsequent values in the time series. Non-stationary time series generally show a very slow decay at the ACF plot. Another way of testing stationarity in time series is the Dickey-Fuller test [19]. If the

time series is non-stationary, the series is differenced once. If the time series is still not stationary after this, the data is differenced again. Generally models use a maximum order of differencing of 2.

If the data contains seasonality normal differencing does not return a stationary time series. Seasonal differencing can be applied in such cases. The data can be tested for seasonality using a seasonal subseries plot, that includes the values of the time series against the time ordered by season. The period of the possible seasonality must be known in advance for constructing the plot.

Once stationarity has been achieved in the time series, the next step is to determine the order of the autoregressive (p) and the order of the moving average (q) processes. These parameters are determined through visual interpretation of the ACF and the partial autocorrelation (PACF) [30] plots. Partial autocorrelation of a variable is the amount of correlation between the variable and a lag of itself that is not explained by correlations at all lower-order lags. Generally a sharp cutoff in the PACF plot combined with a slower decay in the ACF plot indicates an 'AR signature', meaning that adding an AR term will more easily explain the autocorrelations than adding an MA term would. If the ACF plot shows a sharp cutoff, the data contains an 'MA signature', which means it is better to add an MA term to the model.

The selection of seasonal AR and MA terms is similar to the non-seasonal methodology, except only the values of the PACF and ACF plots on the lags equal to the period of the seasonality are considered instead of all values. Iteratively executing the above steps results in the final model. After each step the ACF and PACF are plotted on the residuals of the updated model.

Model Estimation

Once the orders of the model are identified, the parameters should be estimated. The process of parameter estimation is complex and generally done with specific software. Often used methods are non-linear least squares and maximum likelihood estimation. Least squares estimation involves minimizing the squared errors of the model by setting the gradient to zero. Maximum likelihood estimation works with a likelihood function. The likelihood function returns the probability of values for the parameters of the model given the observed data. Likelihood is the data density given the parameter values. Maximum likelihood estimates provide the parameters for which the model is most likely (assuming that the underlying model is true). Maximizing the function results in a set of model parameters with the highest probability.

Model Validation

In the last step the model is evaluated. A well-designed SARIMA model removes all autocorrelations from the data. Hence the residuals of the model should be white noise or independent. If not, the model should be improved by going back to the first step. The residuals can be inspected either visual with the ACF and PACF plots, or

statistically with the Ljung-Box test [42] that assesses whether they are independently distributed.

2.5 Natural Language Processing

This section discusses the main tasks in Natural Language Processing (NLP) [14], and tools developed for their execution. NLP is a branch of artificial intelligence and referred to as the process of computers trying to extract meaningful information from natural language input. A common challenge in this process is due to words possessing different meanings depending on the context.

2.5.1 NLP Tasks

There are multiple elementary NLP tasks. A very basic NLP task is *tokenization* [69]. A tokenizer splits the text in tokens. Tokens are generally single words or numbers. Sentences are split based on word boundaries like spaces or commas. Tokenization varies by the type of language, as, for example, the Chinese language has no clear word boundaries. The obtained tokens can be used for *gazetteering* [23], in which predefined lists (gazetteers) of words are used to find domain-specific words in the text that are subsequently annotated.

A *sentence splitter* [64] groups tokens into sentences based on sentence boundaries, usually indicated by interpunction. Grouping tokens in sentences is needed in order to find grammatical structures that contain information on the meaning of its words. A *part-of-speech tagger* (PoS tagger) [55] determines the function of each word within a sentence. Many words can serve as different parts of speech, for example, the word 'head' can serve either as a noun or as a verb. Words can also have many different forms with similar meanings. A *morphological analyzer* [38] takes care of this issue by reducing all tokens to their lemma, i.e. their most basic form, for example by reducing 'opening' or 'opens' to 'open'.

Many words can have multiple meanings depending on the way they are used in the text. Given the tokens, PoS tags, and lemmas, a *word sense disambiguator* [75] determines the sense of a word group in order to extract its real meaning. Examples of ambiguous words are 'apple', which can mean either the fruit or the company, or 'cycle', which can mean an interval of reoccurring events or a bicycle. Predefined events can be extracted from the text using an *event phrase gazetteer* [31] that scans the text using a list of phrases or words that are likely to represent the description of this event.

Another mentionable NLP task involves *named entity recognition* [63], in which named entities, such as persons, companies, and locations are extracted from the text. Finally, *stemming* [43] is a simpler form of the morphological analysis, in which affixes are removed, reducing words to their stems, that are not necessarily the morphological roots. A summary of NLP tasks is given in Table 2.1.

Table 2.1: Description of different NLP tasks

Task	**Description**
Tokenization	The process of breaking up the text into single tokens (words, numbers, etcetera)
Sentence Splitting	The process of grouping tokens into sentences based on sentence boundaries
Gazetteering	Annotation of the text using predefined lists
Part-of-speech tagger	Determining the function of a word within the sentence using its definition and context
Stemming	Reduces words to their stem, base, or root form, usually by removing affixes
Morphological Analyzer	The identification of a words root form by reducing them to their lemma, going further than just the process of stemming
Word Sense Disambiguator	The process of identifying which sense or meaning of a word is used in a sentence
Event Phrase Gazetteer	Annotation of events in the text using predefined lists
Named Entity Recognition	Extraction of named entities from the text

2.5.2 NLP Tools

In order to automatically determine text contents, many NLP tools and frameworks have been developed. Most of these Information Extraction (IE) methods are based on the English language. One of the most commonly used IE frameworks is the freely available General Architecture for Text Engineering (GATE) [16]. It provides the possibility to construct processing pipelines from different components performing different tasks. Examples of these are linguistic, syntactic, and semantic analysis tasks. The A Nearly-New Information Extraction (ANNIE) system consists of some key components of a pipeline, i.e., the English Tokenizer, Sentence Splitter, Part-Of-Speech (POS) tagger, Gazetteer, Named Entity (NE) Transducer, and OrthoMatcher. By default, ANNIE is loaded by GATE. However, ANNIE lacks Word Sense Disambiguation (WSD) and the ability to look up concepts from a large ontology within a reasonable amount of time. However, because GATE is highly flexible and adapt-

able, ANNIE's components can be used together with external components which can mitigate these drawbacks.

The Conceptual Annotations for Facts, Events, Terms, Individual Entities, and RElations (CAFETIERE) [10] relation extraction pipeline is an example of such an adapted ANNIE system. It adds an ontology lookup process and a rule engine to the basic ANNIE system. CAFETIERE makes use of extraction rules defined at the lexico-semantic level. Because knowledge in CAFETIERE is not stored in Semantic Web ontologies, but by using Narrative Knowledge Representation Language (NKRL), there is no reasoning support. Other drawbacks of CAFETIERE are that gazetteering is a slow process when going through large ontologies, and that the pipeline also misses a WSD component.

The Java Annotation Patterns Engine (JAPE) [15] is a component of GATE. JAPE is a finite state transducer, which means it is able to translate the strings of the input into different strings as output. It operates over annotations based on regular expressions. JAPE grammar consists of a set of phases, each of which consists of a set of pattern/action rules. These phases are run sequentially. Rules have a left-hand-side (LHS) and a right-hand-side (RHS). The LHS contains an annotation pattern description, while the RHS contains annotation manipulation statements. Concretely, the LHS describes a pattern that should be matched in the text and the RHS describes what is to be done with the matched text. The rules in JAPE are similar to the rules in CAFETIERE. However, because the syntax in CAFETIERE is at a higher level, these rules are easier to use, but less flexible.

The Knowledge and Information Management (KIM) [50] platform is a general purpose framework that tries to provide an infrastructure for IE purposes by combining GATE components with semantic annotation techniques. In order to automatically annotate news articles, the system uses a pre-populated OWL upper ontology. The back-end consists of a semantically enabled GATE pipeline, including semantic gazetteers and pattern-matching grammars. The middle layer provides services like semantic repository navigation and indexing and retrieval, and the front-end consists of applications like the Annotation Server and the News Collector. The KIM platform focuses on semantic annotation. Like many other IE platforms, KIM lacks a Word Sense Disambiguator.

Another more extensive adaption of the basic GATE pipeline is the Semantic-Based Pipeline for Economic Event Detection (SPEED) [31]. Like KIM, but unlike many other IE platforms, this framework is semantics-driven and uses a domain-specific financial ontology developed to detect economic events in texts. The main differences with the basic GATE pipeline are that it uses an ontology for gazetteering instead of simple lists, it employs a Word Sense Disambiguator, and finally it uses an Event Phrase Gazetteer and an Event Pattern Recognition component to extract economic events from the text instead of merely performing annotation tasks. The implementation of the framework uses some of the main components of GATE like the English Tokenizer, the Sentence Splitter, and the Part-Of-Speech tagger. The functionality of some other GATE components is extended in this framework. Initial testing results have shown fast gazetteering and decent recall and precision rates for

concept and event identification [31].

2.6 Feature Selection

Selecting a set of well-performing exogenous variables for a forecasting model is a problem addressed in many studies [28, 39, 52, 74]. The manual selection of variables with explanatory possibilities is a relatively straight-forward and intuitive process. However, selecting a subset of variables with the highest forecasting power is not trivial. This procedure is called feature selection.

Evaluating all possible combinations of features is a very time-consuming process, especially for complex models with many features, because the amount of possible feature combinations increases exponentially with the number of features. Also, simply testing all single features separately in the model and selecting a certain amount of the best performing ones does not necessarily produce the best subset, because the performance of features that do not perform well alone can increase when used in combination with other features.

In order to obtain a well-performing feature subset a variety of feature selection algorithms has been proposed. Three of them will be described next, starting with two straight-forward greedy algorithms. The third discussed algorithm is the more sophisticated ant colony optimization feature selection algorithm. As feature selection algorithms work with local feature importance functions, first a description of such a function is given.

2.6.1 Local Importance

A local feature importance function tries to determine the predictive opportunity of adding a feature to the current subset. The local importance of f_i given current subset S_j is notated as $LI(f_i|S_j)$. A local importance function that is based on shared information between different features and between features and the forecasted variable is used in the Mutual Information Feature Selection (MIFS) [9]. The MIFS algorithm makes use of the mutual information (MI) principle in order to determine the probability that adding another feature will increase the forecasting performance of the model.

The mutual information of two variables X and Y, notated as $I(X, Y)$, is a value that indicates the degree these variables correspond to each other. It is also called the level of dependency between the variables. Completely independent variables (X, Y) have $I(X, Y) = 0$. A higher mutual information indicates more dependency between the variables. Note that the mutual information is a symmetric measure, i.e. $I(X, Y) = I(Y, X)$.

The mutual information is related to the measure of entropy. The entropy of a variable X, $H(X)$, measures the uncertainty of this variable. In this context generally the Shannon entropy [56] is used. The entropy is often measured in bits. The entropy

of a single toss of a fair coin is 1 bit, the entropy of tossing two coins is 2 bits, etc. If the outcome is not completely random, like in the example of a coin toss, the entropy is lower. For a random variable X with possible values $x_1, x_2, ..., x_n$, the entropy is based on the probability mass function $p(X)$, which gives the probabilities of X being equal to each separate value X_i. It is defined as follows:

$$H(X) = \sum_{i=1}^{n} p(x_i) log_2 \frac{1}{p(x_i)} \quad . \tag{2.14}$$

The mutual information in terms of entropy is defined in Equation 2.15. In this formula, $H(X|Y)$ represents the conditional entropy of variable X given variable Y. The conditional entropy of X given Y indicates the amount of randomness which remains in X when Y is known. It is defined in Equation 2.16, where $p(Y)$ is the probability mass function giving the probabilities of Y being equal to all of the possible values $y_1, y_2, ..., y_m$.

$$I(X, Y) = H(X) - H(X|Y) \quad . \tag{2.15}$$

$$H(X|Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) log_2 \frac{p(y_i)}{p(x_i, y_j)} \quad . \tag{2.16}$$

In the ideal situation feature selection based on mutual information uses the mutual information between the predicted variable C and each possible subset L. However, a huge computational load is involved with this approach, especially if the size of L increases. Therefore the MIFS algorithm uses an evaluation function which tries to determine the feature which is best to add given the current subset. This evaluation function takes into account the mutual information between different features, and therefore the possible redundancy. It is defined as follows:

$$g_{MIFS}(f_i) = I(C, f_i) - \frac{\beta}{|S|} \sum_{s \in S} I(f_i, s) \quad . \tag{2.17}$$

where

S = Current subset of selected features

β = Parameter regulating relative importance of $I(C, f)$ with respect to $I(f, s)$

2.6.2 Greedy Algorithms

A greedy algorithm is an iterative heuristic that tries to find the global optimum making the optimal local choice in each iteration. Greedy algorithms are not guaranteed to provide the global optimal solution. There are two greedy algorithms for feature selection, greedy forward selection and greedy backward elimination.

Greedy forward selection for feature set F containing all features $f_1, f_2, ..., f_n$ involves the repetitive selection of the most promising feature f_i and adding this to

Algorithm 2.1: Pseudo code for greedy forward selection algorithm

$S = \varnothing$

while !stop **do**

select f_i for which

$\max_{f_i \notin S}(LI(f_i|S))$

$S = S \cup f_i$

end while

Algorithm 2.2: Pseudo code for greedy backward elimination algorithm

$S = F$

while !stop **do**

select f_i for which

$\min_{f_i \in S}(LI(f_i|S \setminus f_i))$

$S = S \setminus f_i$

end while

the pool of used features. The algorithm starts with an empty subset. In each step, the feature with the maximum local importance, given the current subset S, $LI(f_i|S)$ is selected and added to the current subset of features. A specified stopping criterion determines when the algorithm terminates and the current subset is considered the optimal solution. A possible stopping criterion is a specified number of features, another possibility is to stop when the performance of the algorithm does not increase anymore when adding a new feature. The pseudo code of this procedure is given in Algorithm 2.1.

Greedy backward elimination, of which the pseudo code is given in Algorithm 2.2, functions similar to the greedy forward selection. It starts with the full set of features, and repetitively removes the least informative feature. In every iteration, the local importance of each feature in the subset is computed given the other features in the subset. The feature with the lowest local importance is dropped from the subset. Possible stopping criteria are reaching a certain subset size or obtaining a performance decrease when removing another feature.

2.6.3 Ant Colony Optimization

A more complicated feature selection algorithm is ant colony optimization. Ant colony optimization combines local importance functions, knowledge from past well-

performing features, and a random component. The advantage of adding a random component and using knowledge from the past is a decrease of chance to leave out well-performing features because they do not work well without the combination of others.

Ant colony optimization, initially proposed by Marco Dorigo [20], is based on the ant behavior seeking the shortest path between their colony and a food source. The original algorithm was aimed at finding an optimal path in a graph. An example of such a problem is the classical Traveling Salesman Problem (TSP), where a salesman has to visit a group of cities each exactly once and return to the original city. Since introduction of the original ant colony optimization algorithm, all kinds of extensions and generalizations have been developed to fit the model for other types of problems.

Background

In the natural world, ants initially wander around their colony randomly in search of food. While walking around, each ant lays down pheromone trails. If an ant finds a pheromone path laid down by another ant, it will likely stop walking around randomly, and follow this path instead. If the ant eventually finds food, it will return to its colony and reinforce the pheromone on the followed path. This way, paths leading to food will get higher amounts of pheromone, making them more attractive for other ants following these trails, each reinforcing the pheromone again. The more pheromone a path contains, the more likely it is an ant will follow this path.

Over time the pheromone on the paths evaporates. This process reduces the amount of pheromone on each path over time, making often traveled paths more attractive than less traveled paths. Long paths require more time to be traveled, resulting in more time for evaporation and less reinforcement. These mechanics make sure short paths contain higher amounts of pheromone than long paths. Eventually, situation converges to the 'optimal solution' where all ants travel the same shortest path because this contains significantly more pheromone than other paths.

Feature Selection Algorithm

The ant colony optimization algorithm for feature selection simulates the process followed by ants in the natural world. The algorithm runs in iterations, where each iteration a number of 'ants' develop a possible solution that is a subset containing a number of features. These subsets are evaluated, and the best subsets are used to update the pheromone intensity of their containing features. The pheromone trails are updated using the performance of the forecasting model with the features of the selected subset. Only the pheromone of the features contained in the best performing subsets is reinforced. A percentage of all pheromone values is subtracted after each iteration, imitating the evaporation.

While in the first iteration all subsets are selected randomly, the subsets in the following iterations are partly selected based on pheromone trails. Each subset in an

Table 2.2: Parameters and notation of ant colony optimization feature selection

Parameter	**Meaning**
na	Number of ants
F	Complete subset of features
n	Cardinal of F
f_i	Feature i
S_j	Subset of features for ant j
m	Cardinal of S
τ_i	Intensity of pheromone associated with f_i
k	Number of best subsets used to influence next iteration
p	Number of new features each iteration
$1-\rho$	Relative amount of pheromone evaporation
μ	Parameter to control relative effect of trail intensity
κ	Parameter to control relative effect of local feature importance
cc	Initial value of pheromone for each feature
β	Constant used in MIFS measure
$USM_i^{S_j}$	Updated Selection Measure of feature f_i given subset S_j
$LI_i^{S_j}$	Local Importance of feature f_i given subset S_j

iteration consists of a number of features selected randomly from the best performing subsets in the previous iteration, supplemented by the most promising features given the current subset and the pheromone trails. The most promising features are selected in a way similar to greedy forward selection, by using as selection function the Updated Selection Measure (USM):

$$USM_i^{S_j} = \begin{cases} \frac{(\tau_i)^\mu (LI_i^{S_j})^\kappa}{\sum_{g \notin S_j} (\tau_g)^\mu (LI_g^{S_j})^\kappa} & \text{if } i \notin S_j \\ 0 & \text{Otherwise} \end{cases} \tag{2.18}$$

This way, the features included in the best performing subsets have a higher chance to be included in the following iteration, while maintaining the possibility to explore the performance of other features. The pseudo code of the full procedure is given in Algorithm 2.3, while all input parameters and their notations are listed in Table 2.2.

Algorithm 2.3: Pseudo code of the ant colony optimization feature selection

$\tau_i = cc, \Delta\tau_i = 0$ for $i = 1, 2, ..., n$

define parameters $na, F, m, k, p, \rho, \mu, \kappa, \beta$

initialize na random subsets each containing m features

evaluate and sort these subsets based on performance

store best performing subset S_1 with performance indicator

update τ for $f_i \in S_{1,...,k}$

while !(maximum number of iterations reached) **do**

create na subsets with $m - p$ features where $f \in S_{1,...,k}$ from last iteration

complete subsets with greedy forward selection using as selection function the USM

replace duplicate subsets with random ones

evaluate and sort subsets based on performance

store best performing subset S_1 with performance indicator if new optimal solution is obtained

update τ for $f_i \in S_{1,...,k}$

end while

2.7 Evaluation Metrics

Time series forecasting models can be evaluated with two widely used performance metrics. These are the Root Mean Squared Error (RMSE) and the Mean Absolute Percentage Error (MAPE). In the case of call center load forecasting, RMSE indicates the absolute deviation of the forecast from the actual load in number of calls on average per day. In the same context, MAPE gives the relative deviation of the predicted load from the actual load as a percentage of the actual total calls per day. Both metrics are generally used for measuring the accuracy of the fitted time series model. The lower bound of both measures is zero, indicating a perfect fit. There is no upper bound on either metric. The definition of the RMSE is given in Equation 2.19 and of the MAPE in Equation 2.20.

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(F_t - A_t)^2} \quad . \tag{2.19}$$

$$MAPE = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| . \tag{2.20}$$

where

n = Number of predicted observations
F_t = Forecasted value for period t
A_t = Actual value for period t

Chapter 3

Study Design

This chapter discusses the study setup for enhancing the use of exogeneous variables in call center load foreacasting. Methodologies discussed in the previous chapter are evaluated with respect to the present study, and all the choices are explained.

3.1 Call Center Planning

As the call center of Vliegtickets.nl has a queue for callers when all operators are busy, both the basic Erlang-B and the extended Erlang-B methods might produce inaccurate results for not considering this. The Engset formula is not appropriate for this study because it is designed for service centers with a small number of sources. The number of sources of the call center of Vliegtickets.nl is clearly not limited to 200 callers.

The Erlang-C formula, in contrast to Erlang-B, is based on a system with queues. Therefore this formula is likely to give more accurate results. The square root rule can calculate the number of operators needed based on the offered load and an indicator for the desired service level, using the same mechanics of Erlang-C. As these formulas have more properties in common with the call center of Vliegtickets.nl, the square root rule will be used to determine the number of operators needed, and the Erlang-C to determine the expected probability of queuing based on this number of operators. To compute the offered load the daily totals were used. A percentage following from past data indicates the number of calls during the busy hour. Doing an in-depth analysis of call arrival times, call holding times, abandon rates, and their distributions is beyong the scope of this study. While Erlang-A fits the characteristics of the considered call center better because it accounts for abandonment, it is not used in this study for the same reason and because extensive data about abandon rates is not present.

3.2 Load Forecasting

The advantage of studies [17] and [45], that forecast stocks and the MSCI EURO index respectively, over ours is that the text items they use for the forecasting are clearly defined, due to input texts being specifically aimed at the forecasted variable. This is opposed to models predicting call center load, where it is usually not clear beforehand which news items will influence the load. However, to eliminate this difference, a filtering algorithm could be designed in order to obtain only the news items that do influence call center load. Using these methods to this filtered news might be applicable. However, the filtered news items do not directly describe the call center load, so it is not guaranteed to perform equally well in this study.

The goal of [49], where exchange rates are forecasted using news headlines, is comparable to ours because it uses news headlines as input which do not necessarily all have an impact on the predicted variable. However, the predicted variable is categorical and uses three mutually exclusive possible outcomes (the exchange rate will go up, remain steady, or go down), while the predicted variable in our study is continuous. Still, the presented techniques involving probabilistic decision rules might be usable in our study context for predicting whether there will be a peak in call center load in the next period. An advantage of using decision rules over many other methods is that they do not only present the outcome, but also indicate the causes in an easily understandable format.

Because the input source (the financial column 'Abreast of the Market') used in [62] directly describes the forecasted variable (stock market activity), the input variables, determined with the General Inquirer (GI), are supposed to be directly linked to this variable. Therefore a Principal Component Analysis could be used in order to determine the most influencing categories from the GI. PCA is a mathematical procedure to reduce the number of input variables whithout losing forecasting power. While our study is different, the use of the GI seems to be an interesting approach to determine the content of the used texts. Another difference between [62] and the current study is that the current one tries to accurately predict the exact load of the call center instead of merely researching the relationship between two variables.

For the analysis carried out in [24] using the clustering algorithm, a strong relationship between the news items and the trend is needed. In addition to this, many data instances backing up this relation are required for the clustering. Because call center load peaks are only occasionally caused by news events, there might not be enough training data available to use this approach in the current study. Also [24] only predicts whether the trend will be rising or dropping, in contrast to our study which tries to predict the exact load.

Sentiment analysis [1, 27, 46] is an interesting methodology for determining the terms from news items that have an influence on a certain linked time series. These studies have the advantage that there is a lot of data available for automatic labelling, which is not the case in our study. However, sentiment analysis is able to give potentially useful fingerprints of news items. As clearly not all news items influence the load of the call center in our study, a good filtering method for the news is needed

for the sentiment analysis to be applicable. The filtering algorithm should select the news items which have an influence on the load, after which the sentiment analysis automates the process by determining which terms are significant predictors of the observed changes.

The use of a prior knowledge base constructed by domain experts, as in [32], is a valid approach for call center load prediction, as there generally is not enough data available to automate this process. However, using solely neural networks based on event information and averages of rates in the past days may not capture extensive trend information, which can be seasonal or dependent on the day of the week. To capture this information another time series estimation model should be used in combination with an algorithm capturing the influence of the news events.

While the use of social networks for forecasting purposes [6] is surely interesting and has shown to provide good results, the utilized models in [6] are relatively straightforward. For the more complicated prediction of call center load using news, these models are not likely to produce acceptable results. Using the information from social networks in combination with more extensive models could possibly be useful for call center load prediction though, as people might post status updates about events which cause trouble with their flights. Basically studies using social networks measure the same information as studies using news items because both studies try to capture the same external events, only from different sources. However, the current study uses direct news text mining because this is easier from a data availability perspective.

While the model proposed in [65] does not use text analytics, it is still an interesting option for call center prediction as the used neural network could easily be extended with more input parameters depending on news events. This results in a model that combines basic time series analysis with inputs based on external events. One of the advantages is that neural networks and SARIMA models both are able to forecast continuous values, required for call center load prediction.

ARIMAX models, as in [13], are applicable to call center load prediction using news because the basic time series are then forecasted using the (possibly seasonal) ARIMA part of the model, while the news part is incorporated in the model as an explanatory variable. Advantages of this approach are that one model takes care of all variables, and that the model provides a prediction of a continuous variable. Considering these advantages and because it is a model designed for time series, the ARIMAX model is used in this study with parameters from the news as exogenous variables. This is also in line with the scope of this study, which concerns enhancing basic call center load forecasting models with exogeneous variables mined from the news.

3.3 Extended Models

The General Inquirer (GI) [58] is able to categorize texts based on large dictionary lists containing words along with tag categories they belong to. The tag categories

used by the GI are taken from four sources, the Harvard IV-4 dictionary, the Lasswell value dictionary, several recently constructed categories, and marker categories like numbers, prepositions, and pronouns. This results in a total of 182 categories that are not mutually exclusive. Examples of categories are 'positive', 'negative', 'hostile', 'strong', 'passive', etcetera.

As these categories basically give a 'fingerprint' of the news, it is promising to use category counts as features for the ARIMAX model. First a manual peak investigation is executed. In this process the news around peaks is manually searched for possible causes of the peak. Based on this analysis several categories from the GI are selected as features for the model. In order to automate the process and account for possible influences missed by the manual investigation, the feature selection algorithms are also used to determine some feature sets. All models are finally evaluated with the discussed evaluation metrics and compared to the basic time series model.

3.4 Tools Used

This study applies several tools for different tasks involving wrapping news items from the Internet, categorising, counting, and filtering news items, calculating mutual information and entropy, identifying peaks in the call center load, estimating and using SARIMAX models for prediction, evaluating SARIMAX models, and the greedy feature selection algorithms as well as the ant colony optimization feature selection. In this chapter I describe the tools used and developed for these tasks.

3.4.1 Processing News Items

For all news processing a News Processor is developed in Java. Many libraries have been made in Java which are able to process XML formats and get information online. In order to get the news items from both used websites the News Processor contains a newswrapper. The wrapper is able to download all news items of a specified date range from both used news sources. It distinguishes between title, date, time, abstract, body, and location, and is able to save it all in the XML format, in order to load it from the file again later. The News Processor is able to filter the news using a keyword-based search. The filter can handle both combined and separate 'and'- and 'or'-queries. It is also able to output the daily amount of filtered news items given the specified query.

The News Processor can categorise the news of a specified date range based on input CSV-files containing categories along with their containing words, like for example the General Inquirer dictionary master spreadsheet. It outputs a spreadsheet containing the counts of the categories for each day of the specified period. The categoriser uses a keyword-based approach. All parts of the News Processor can easily be used in a chain, allowing for categorising filtered news obtained from either the Internet or a local file.

For wrapping the news from the websites the Jaxen [35] library for Java is used in order to be able to get the needed text objects from the website using XPath [68] queries. Jaxen uses the JDOM package [51] for parsing the XML code. This package is also used to output the news to an XML-file.

3.4.2 Statistical Analysis

The tools for statistical reasoning are developed using R [53], that is a language for statistical analysis. It it also possible to use R in combination with Java using the rJava package [66] to call Java functions from R, or the other way around with JRI. A large variety of packages implementing all kinds of analyzing and forecasting models has been developed for R.

For calculating and saving the entropy and mutual information, as are used in the MIFS measure, a tool based on the 'entropy' package [29] is used. This tool outputs matrices containing the entropy of the features from the feature pool and the call center load, as well as their mutual information. The estimation of the SARIMAX models is done using the default 'stats' package, that also does the actual prediction. The functions used estimate the SARIMAX models based on the regression of the external inputs. Evaluation of the model on the training sets is done using the included accuracy report in the 'forecast' package [34], while a different tool was developed specifically for the purpose of evaluation of the models on the test sets. Both the greedy feature selection algorithms and the ant colony optimization feature selection are implemented in R using the 'forecast' package for local evaluation of all considered subsets. All packages used are taken from the Comprehensive R Archive Network (CRAN).

Chapter 4

Model Design

In this chapter I describe the procedures followed to develop the models used in this study. After briefly exploring the data, the optimal basic SARIMA model for the data is determined. Following up the peaks are manually investigated and finally the models using external variables are determined.

4.1 Data Exploration and Preparation

The dataset used in this study consists of the daily load of the call center of Vliegtickets.nl over the years 2009, 2010, and 2011. The average number of calls a day for the three years is 260. The average number of calls for the years 2009, 2010, and 2011 are 254, 250, and 302 respectively. No apparent trend can be deduced from these averages. The averages per day over the three years are given in Table 4.1.

Based on these averages we can assume there are less calls on weekend days

Table 4.1: Average number of calls per day.

Day	Average
Sunday	90
Monday	372
Tuesday	333
Wednesday	332
Thursday	296
Friday	286
Saturday	110

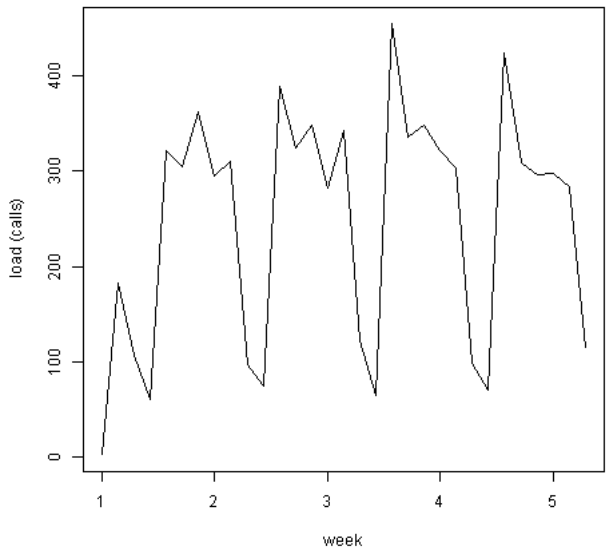

Figure 4.1: Vliegtickets.nl's call center load of January 2009

than on weekdays. On average most calls are on Mondays, and the mean amount decreases gradually over the rest of the week. To illustrate and backup this assumption the graph of the call center load of January 2009 is included in Figure 4.1. In this graph the negative peaks are on the weekends, usually followed by a high peak on Monday.

Call center load of 2010 is given in Figure 4.2. This graph shows that the number of calls during the year is relatively steady, except for some peaks. The peaks and their possible reasons will be discussed later.

The data contains six missing values. To ensure the correct working of all methods used in the study, the missing values are replaced with the average loads of the corresponding days during the same year. Because there are so few missing values, it is unlikely that these changes will influence the results significantly.

4.2 SARIMA Parameter Estimation

This section follows the model identification of the Box-Jenkins methodology in order to fit the best ARMA model or variant to the call center data of Vliegtickets.nl. The analysis is based on the daily call center load of the years 2009 and 2010.

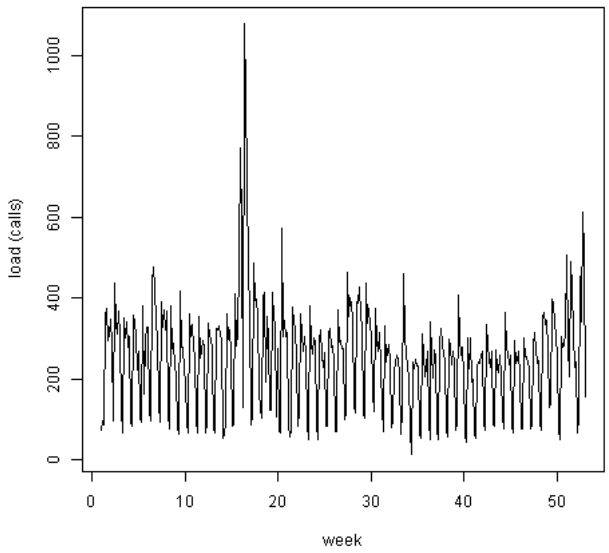

Figure 4.2: Vliegtickets.nl's call center load of 2010

4.2.1 Identification

The identification starts with demeaning the data by plotting the residuals of the simple linear model with constant. The Auto Correlation Function (ACF) and the Partial ACF (PACF) functions of these residuals with a maximum lag of 50 days are also plotted. The plots are given in Figure 4.3.

In these plots there is clearly a seasonal effect, because there is a pattern repeating each 7 days. The data exploration already suggested this. Because the length of the cycle is one week, the period of seasonality is 7. The seasonal subseries with a period of 7 as given in Figure 4.3 backs up these assumptions by showing a significantly higher load during weekdays than during the weekend.

The ACF and PACF plots from Figure 4.3 include large peaks, so the series seems to be non-stationary. At least one order of differencing is appropriate. Testing for stationarity is done by the Dickey-Fuller test, as described in subsection 2.4.4. This is testing if a unit root is present in an autoregressive model of the data, indicating non-stationarity. The outcome of this test is -19.2382, which is smaller than the critical value of 3.22 for $p = 0.01$. Therefore the null-hypothesis of stationarity is rejected, and the alternative hypothesis of non-stationarity is accepted. The Dickey-Fuller (DF) values for the load on different days of the week are given in Table A.1 in the appendix. As all the values are below the critical value, stationarity while taking seasonality into account is also rejected.

The Root Mean Squared Error (RMSE) is a good indicator for the variance of

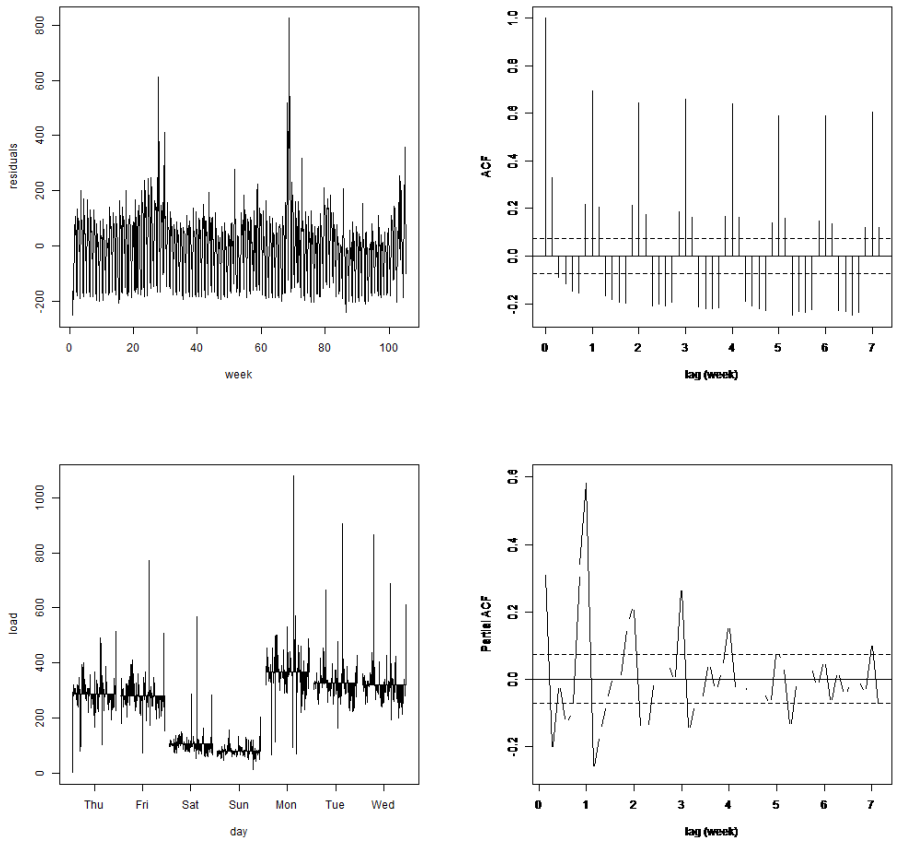

Figure 4.3: Clockwise, starting top right: Residuals, ACF, and PACF plots of de-meaned call center data, and seasonal subseries.

the residuals. The goal of differencing in ARIMA modeling is to achieve a stationary variance. Therefore the RMSE of the linear constant models with different orders of differencing is calculated. The results are given in Table 4.2. The lowest RMSE is produced by the model with one order of seasonal differencing and no non-seasonal differencing. Therefore these parameter values ($d = 0, ds = 1$) are used for the final model. The plot of the residuals, the ACF plot, and the PACF plot of the SARIMA(0,0,0) (0,1,0) model are shown in Figure 4.4.

The model is improved as the peaks in the ACF and PACF plots of Figure 4.4 are smaller than in the plots of the basic linear model (Figure 4.3). However the series are still mildly under-differenced, because there remain significant peaks in the ACF and PACF plots. In order to fix this a non-seasonal autoregressive term is included in the model, making it a SARIMA (1,0,0) (0,1,0) model. The RMSE of this model is decreased to 90 from 98 for the SARIMA (0,0,0) (0,1,0) model. The plot of

Table 4.2: Root Mean Squared Error values of linear models with varying orders of differencing

Model	**RMSE**
ARIMA(0,0,0)(0,0,0)	130
ARIMA(0,1,0)(0,0,0)	150
SARIMA(0,0,0)(0,1,0)	98
SARIMA(0,1,0)(0,1,0)	109
ARIMA(0,2,0)(0,0,0)	230

the residuals along with the ACF and PACF plots are shown in Figure 4.5 (left side). Both the ACF and the PACF plots still show a negative peak at lag 7 and lag 14 (week 1 and week 2), indicating the need for a seasonal moving average term. Including this term results in a SARIMA (1,0,0) (0,1,1) model. The RMSE of this model is 68, which is a significant improvement from the RMSE of 90 for the SARIMA (1,0,0) (0,1,0) from the last step. The usual plots are shown in Figure 4.5 (right side).

There are still small occasional peaks in both the ACF plot and the PACF plot of Figure 4.5 (right side). However, investigation shows that increasing any of the terms in the model does not remove these, nor does it yield a significant decrease of the RMSE. Therefore SARIMA (1,0,0) (0,1,1) model will be used for the basic call center load forecast.

4.2.2 Evaluation

The models are evaluated on a test set. Tables with RMSE and MAPE values are included in Appendix A. The load of each day of 2010 is predicted based on a model built on the 365 days before this day. Table A.2 shows the RMSE and the Mean Absolute Precision Error (MAPE) for different models. This evaluation shows that the SARIMA (1,0,0) (0,1,1) model is indeed one of the best performing models, with as good alternative the SARIMA (1,0,1) (0,1,1) model.

As possible improvement the AR term of the 4 best performing models with 1 AR term is increased by 1, making the AR term 2. The results of this test are included in Table A.3. Adding an AR term to the model results in a decrease of the MAPE and RMSE, resulting in a new best model being SARIMA (2,0,0) (0,1,1) with a MAPE of 20.9% and a RMSE of 75.3.

Trying to further enhance the model, more AR terms are added to the models. The evaluation results of these models are shown in Table A.4. As can be noted from these results, increasing the amount of AR terms does not or hardly improve the models, depending on evaluation measure. Therefore the SARIMA (2,0,0) (0,1,1)

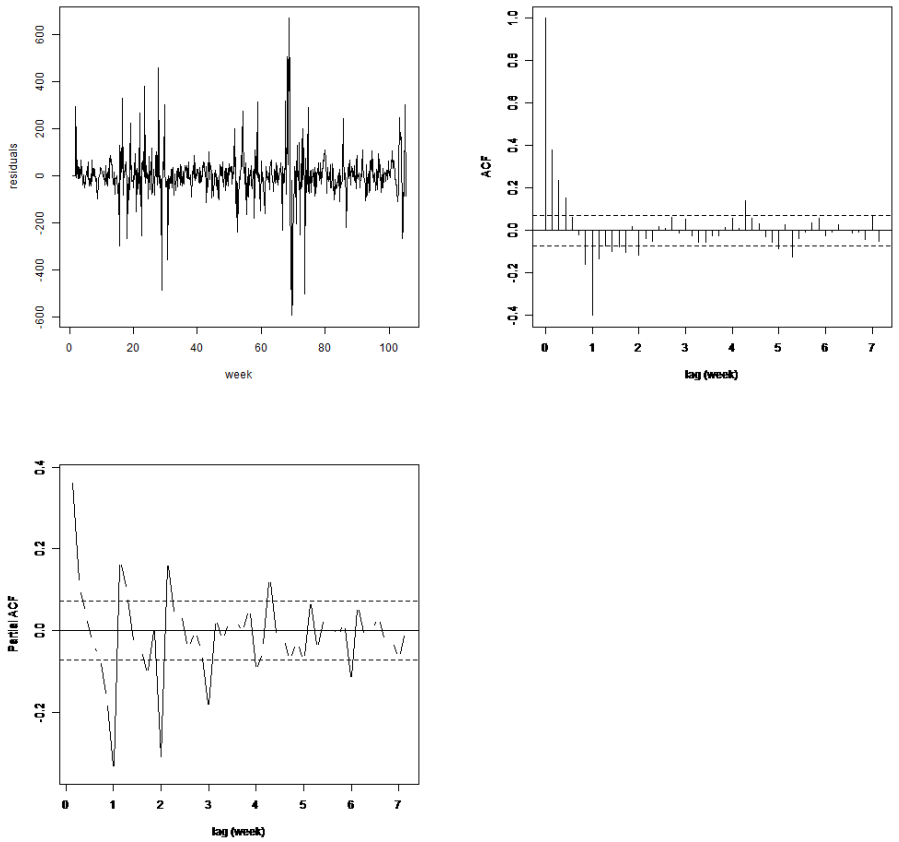

Figure 4.4: Residuals, ACF, and PACF plots of the seasonally differenced data.

model is used as final model, with a seasonal frequency of 7 days. As estimation method for the parameters the conditional sum of squares is used.

4.2.3 Robustness Test

From the original demeaned data the assumption arises that there are positive outliers in the data. The boxplot in Figure A.1 confirms this assumption. Outliers here are considered datapoints that lie more than 1.5 times the inner quartile range from the first and third quartiles. This is also in line with the nature of the data, because there will be extremely busy days due to external events which cause positive outliers. Negative outliers are less likely to occur because extremely quiet days are less likely and a negative load of the call center is not possible.

In order to check the robustness of the obtained model all outliers from the data are replaced with the averages for this day measured over the whole dataset and execute the same analysis in order to obtain the best model. The complete analysis

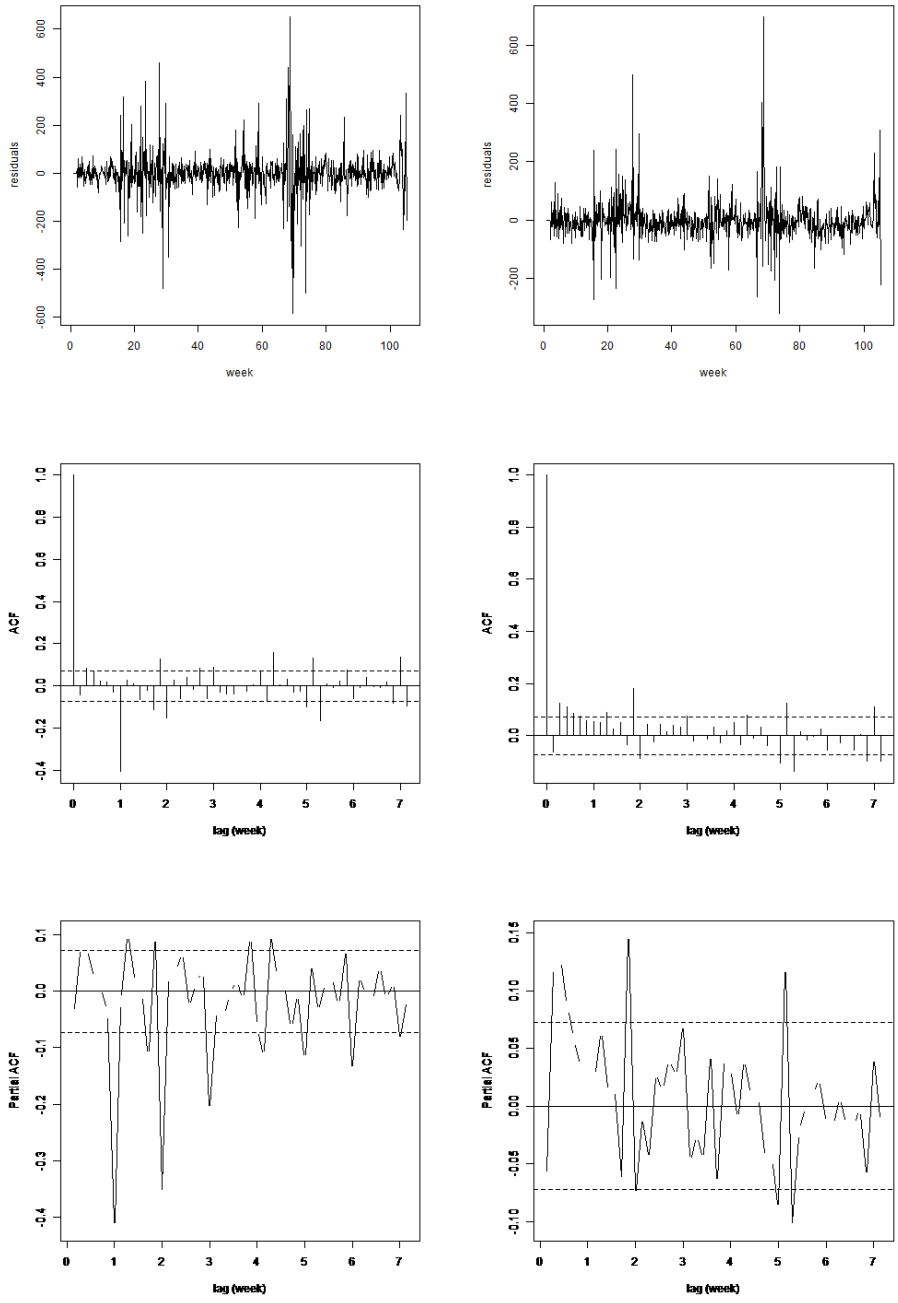

Figure 4.5: Residuals, ACF, and PACF plots of the SARIMA (1,0,0) (0,1,0) (left side) and the SARIMA (1,0,0) (0,1,1) (right side) models.

is carried out again, yielding the same results as the analysis on the dataset including outliers. The plots are omitted from this study because they are similar to the plots in the previous section.

Table A.5 shows the evaluation of the methods on the test set without outliers. This evaluation shows no significant change in the evaluation measures for the selected model, and the model is still one of the best performing ones. However, for data excluding outliers, increasing the number of AR terms will yield better results. This phenomenon is usually referred to as overfitting, resulting in models that are not robust to outliers. This is confirmed in the evaluation results of the models with multiple non-seasonal AR terms. As such models are less robust to outliers, the SARIMA (2,0,0) (0,1,1) model is used for basic call center load prediction in this study, as obtained in the previous section.

4.3 Peak Investigation

In this section peaks in the load of the call center are defined and located. In order to get an indication of the type of news events the news preceding peaks is manually searched for possible causes. The main results are reported here.

A possible way to locate peaks is by selecting all data points which have a large deviation from the average load. However, as the load significantly varies depending on the day of the week, weekdays will have a much higher chance of being a positive peak, while weekend days have a higher chance of being a negative peak. This can be solved by using the daily averages instead of the global average. However, this approach does not filter out peaks which are already accounted for in the SARIMA model, like naturally caused peaks for example by busy periods due to summer time. Therefore a peak is here defined as a data point with a positive or negative deviation from the predicted load by the obtained SARIMA(2,0,0)(0,1,1) model of more than a certain percentage, hence being unexplained variance in residuals. This definition makes sense also because the essence of this study is to be able to adjust and improve the prediction made by the basic model. Because the MAPE of the basic model is 20.9% for the years 2009 and 2010, the critical percentage for peaks is set a bit higher at 25%. This results in a reasonable number of positive and negative peaks. Both positive and negative peaks are considered, however finding many news events causing negative peaks is not likely. The reasoning behind this is that people probably will not stop calling Vliegtickets.nl due to some news event, but only call more often.

With this definition for peaks a total of 25 positive peaks and 36 negative peaks for the year 2010 is obtained, and 16 positive peaks and 30 negative peaks for 2011. With the used approach it is not possible to locate peaks in 2009, because this is the first available year in the dataset so previous years can not be used for the prediction. Therefore the lesser approach of using the deviation from the daily averages is used for this year. Because this approach is less accurate slightly larger critical percentage margin of 30% is used. This method resulted in 11 positive and 10 negative peaks.

Table 4.3: Possible peak-causing events

Event
Mexican H1N1 flu becoming pandemic
Heavy rain, storm in Europe
Heavy snowfall troubling flight travel
Dutch to use full-body scans for flights
Volcano eruption troubling flights
Airways company strikes
Smoke chokes Moscow, planes diverted
Major unrest in Arabic countries
Public sector strikes

Possible causes in the news for 8 positive peaks in 2009, 18 positive peaks in 2010, and 13 positive peaks in 2011 are found. Most possible explanations in the news occur one or two days before the peak. As expected no possible explanations in the news for negative peaks were found. However, further investigation showed that some of the negative peaks can be explained by Dutch national holidays like Easter, Pentecost, Christmas, and New Year's Eve. Almost all national free days in the three years yield negative peaks. A summary of the types of events which seem to have a high probability of causing a positive peak in call center load is given in Table 4.3. News sources used for this analysis are Reuters UK [54] and the Dutch NOS website [47]. Because there are no large differences between these two websites and because most existing language processing tools are English the English Reuters is used as the news source for forecasting in this study.

It is hard to use these specific events for prediction, because several of them are rather unique events which are not likely to happen very often. An example is the Icelandic volcano eruption in 2010 which grounded all flights in large parts of Europe and caused a large peak in call center load. It is not likely that a very similar event will happen often in the near future. An event with more promising forecasting possibilities is heavy weather causing airport closure in Europe, because this type of event is more likely to happen frequently.

Searching the obtained possible causes for similarities, most of these events are described by at least some news items which also mention the influence this event has on flight traffic. Operations on the news corpus like filtering or counting words using keywords concerning air traffic trouble might yield well-performing forecasting models.

Another problem with linking events to peaks arises specifically in the year 2011.

Table 4.4: Training sets

Training set	Years
1	2009
2	2009 & 2010

Overall, positive peaks in 2011 are more spread out and smaller than the peaks in the other two years. This makes it harder to link them to specific events, because several of the possible causes are on-going problems like, for example, the 'Arabic spring' revolutions in Tunisia, Libya, and Egypt. Because news sources frequently release news items about these subjects, it is hard to dedicate them to a specific peak. To conclude, manual peak investigation does not result in a clear methodology to use in order to improve the basic forecasting model using the news.

4.4 Extended Models

Because manual investigation of the peaks does not provide a clearly defined methodology to forecast peaks in call center load using news multiple models are designed, obtained either manually or by automating the process of feature selection. This section describes all the steps taken in order to develop a forecasting model that is able to cope with peaks caused by external events. All models are SARIMAX(2,0,0)(0,1,1) models using the regression of external inputs. The selection of exogenous variables or features is described in this section. Two partly overlapping training sets are used for model estimation: the first one contains the data from 2009, and the second one contains all data from 2009 and 2010. These training sets are designed because the obtained models should be tested on both 2010 and 2011, while a larger training set has a higher probability of producing a well-performing model. Training and testing on the same years can yield biased evaluation results [40].

4.4.1 Base Model

The first considered feature is an indicator whether the day is a Dutch national holiday. Because this feature is not news-related, it does not contribute to the main goal of this study. However, because the peak investigation suggested a strong relationship between national celebration days and negative peaks in call center load, I decided to use this feature in order to create the most optimal forecasting model. This feature is not lagged because national holidays are fixed days and their dates are known in advance each year. All features considered are positive numbers by nature.

As I am primarily investigating the impact of external events on the call center load I select the 'holidays' feature by default, thus incorporating it in the base model. A short evaluation predicting the years 2010 and 2011 using the previous

Table 4.5: Evaluation of 'holidays' feature

Predicted year	**RMSE without 'holidays'**	**RMSE with 'holidays'**
2010	77.02278	71.59584
2011	63.07692	59.46052

years shows that the SARIMAX(2,0,0)(0,1,1) model with the 'holidays' feature as the only exogenous variable performs better than the basic SARIMA(2,0,0)(0,1,1) model without external variables. The RMSE values of the models including and excluding 'holidays' are given in Table 4.5.

4.4.2 Obtaining Feature Pools

The manual peak investigation suggested a relationship between positive peaks and news items describing events that influence flight traffic. Because such news items usually mention the impact of the event on air travel, searching the news for flight related terms intuitively provides forecasting possibilities. The available feature pools are created manually.

The first pool of available features consists of the total number of words falling under each category, aggregated over all news items each day. Because today's news can not be used in order to forecast today's load, the category counts are lagged one day. This is also in line with the peak investigation which showed that positive peaks related to events usually occur one or two days after the first day of the event. It results in a model that forecasts call center load based on the news of one or more days before the forecasted day.

These measures might not contain all available information because they do not distinguish between one news item containing many flight related words and several news items each containing only one flight related word. Therefore another feature is included in the pool of available features containing the number of news items resulting after querying the news using keywords 'airport' and 'flight'. Because of the general nature of these two keywords, this approach is able to obtain the majority of news items that relate to air traffic. Like the categories, this feature is lagged one day. The feature is labeled 'filter_count'.

Added to this pool are two features representing the weather. The first of these is the average temperature per day, measured in Kelvin to avoid negative values, indicated by 'Temp'. The second indicator for the weather is the daily precipitation amount ('Precip'). Both metrics are not taken from the news but from the database of the Dutch weather institute Royal Netherlands Metereological Institute (KNMI). Both weather-related features are lagged one day.

The second pool of features developed for forecasting purposes contains the category counts from the same dictionaries, but instead of using the full news corpus, only the filtered news items by the keywords 'airport' and 'flight' are categorized.

Table 4.6: Explanation and examples of several GI categories

Category Name	Category Description	Examples
Hostile	Words indicating an attitude or concern with hostility or aggressiveness	Ambush, anger, anarchy, battle
Sky	Words for all aerial conditions, natural vapors, and objects in outer space	Air, sky, snow, weather, wind
Travel	Words for all physical movement and travel from one place to another in a horizontal plane	Arrival, depart, flight, journey
Rise	Words for upward movement	Ascent, elevate, fly, raise

This approach might make the suggested flight-related categories less useful, because they possibly become redundant. Other categories may be more interesting when this feature pool is used. This pool does not include the number of air traffic related news items or the weather-related features.

4.4.3 Feature Selection

As flight traffic related words are promising for forecasting purposes, the categories 'sky', 'travel', and 'rise' from the GI seem specifically interesting, especially because the category 'sky' also contains words related to the weather. The first models considered use the manually selected features 'sky', 'travel', 'rise', and 'filter_count' from feature set 1. Because some peaks seem to be more related to events involving revolutions in the Middle-East, the 'hostile' category feature is also used. To keep a reasonable amount of models just the five models using these single features, the ten models containing each possible combination of three out of the five mentioned features, and the model containing all of these features are considered, resulting in a total of 16 models. Next added is a model involving testing the impact of the weather on the call center load. The features in this model, beside the default 'holidays' feature, are 'Temp' and 'Precip'. The manually obtained 17 models are shown in Table 4.7 and labeled as model 1 to 17. A description of the used categories along with some examples of words contained in them is given in Table 4.6.

Because the now obtained models are solely based on presumptions about influencing events, the process of feature selection is automated using the feature selection algorithms described in Section 2.6. The greedy selection algorithm is used to de-

Table 4.7: Table of models

Model	**Features**
1	Hostile
2	Sky
3	Rise
4	Travel
5	Filter_count
6	Hostile, Sky, Rise
7	Hostile, Sky, Travel
8	Hostile, Sky, Filter_count
9	Hostile, Rise, Travel
10	Hostile, Rise, Filter_count
11	Hostile, Travel, Filter_count
12	Sky, Rise, Travel
13	Sky, Rise, Filter_count
14	Sky, Travel, Filter_count
15	Rise, Travel, Filter_count
16	Hostile, Sky, Rise, Travel, Filter_count
17	Temp, Precip
18	Doctrine, Economy, WealthOth, WealthTot, Precip
19	Our, WealthOth, WealthTot, WellBePhys, Precip
20	Affiliation, Undrstated, Race, Social, PowerEnds
21	Negative, Understated, Evaluate, TransactionGain, Temp
22	Understated, Economy, Complete, WealthOther, WealthTotal
23	Exchange, Sky, PowerDoctrine, SkillTotal, Nation
24	Academic, ComonObj, Time, AffectOther, Filter_count
25	SocialRelations, ComonObject, Travel, EnlightGain, Nation
26	Strong, Political, Space, Explanation, PowerTotal
27	Strong, Doctrine, Travel, Positive, Negate
28	Pain, Ordinal, AffectLoss, WealthTransaction, EnlightEnds
29	Race, Number, PowerLoss, RespectGain, AffectLoss
30	Sky, Vary, Decrease, WealthOther, TransactionGain
31	Sky, RectitudeLoss, RespectGain, EnlightLoss, Arena
32	Female, NaturalObject, Fall, PowerLoss, Anomie

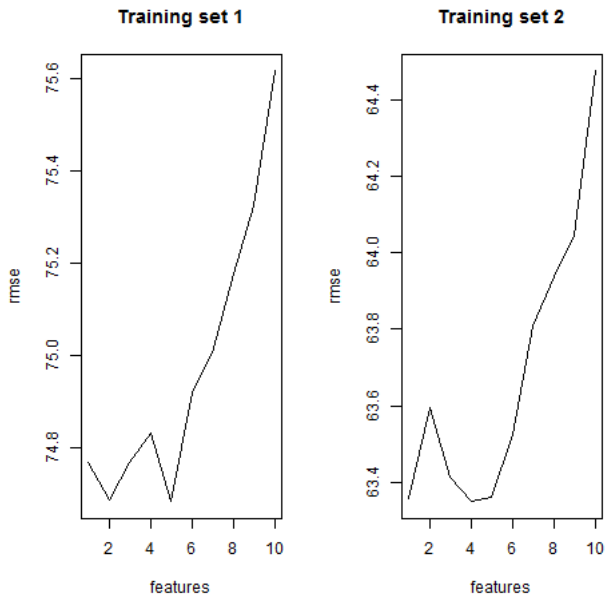

Figure 4.6: Determining the optimal feature subset size

termine the optimal number of features for the models. While this algorithm may not result in the optimal feature set, testing on multiple years might give a reliable indication of the number of features to be used. Two tests are designed, involving the use of training set 1 to predict the year 2010, and the use of training set 2 in order to predict the year 2011. The results are shown in the graph in Fig. 4.6. Both tests suggest that the optimal number of features to be used is approximately 5.

With the determined optimal number of 5 features more models are constructed using the three feature selection methods. Both greedy algorithms use a fixed approach, so each given number of features results in a feature set which is the same each run. Because the resulting sets of the greedy algorithms are almost equal for both training sets only the two obtained sets of these algorithms with training set 1 are used. For the MIFS measure the constant β is set at 0.6, as suggested in [9]. The results labeled as model 18 and 19 respectively are given in Table 4.7.

Because the ant colony optimization (ACO) incorporates a random component, and is sometimes depending on the randomly chosen features from the first iteration, it does not return the same feature set each run. Therefore the models that perform best on the training sets, after multiple runs of the algorithm, are selected. For the MIFS measure $\beta = 0.6$ again, and for the optimization algorithm $\mu = \kappa = 1$, trail evaporation $1 - \rho = 0.25$, and the initial value of pheromone for each feature $cc_i = 1$, as suggested in [3]. The number of ants or subsets *na* used is 30, the best k number of ants used to influence the next iteration is set at 10, and the number of new features for iteration p is 3. Finally, each run of the algorithm consists of 50 iterations. A total

Table 4.8: Methods used to obtain feature sets

Model	Method	Training set	Feature Set
1-17	Manual	-	1
18	Greedy Selection	1 & 2	1
19	Greedy Elimination	1 & 2	1
20-22	ACO	1	1
23-25	ACO	2	1
26	Greedy Algorithms	1 & 2	2
27-29	ACO	1	2
30-32	ACO	2	2

of six feature sets were selected using ACO for further evaluation, which are shown in Table 4.7, labeled as model 20 to 25. Three of these are trained on training set 1 and the other three are trained on training set 2.

The last models considered are automatically developed from feature set 2. The first of these models is constructed with the greedy algorithms on training set 1. Both greedy selection and greedy elimination resulted in the same model with this feature set, labeled model 26. The concluding six models, labeled model 27 to 32, are determined with the ant colony optimization, consisting of three models based on training set 1 and three models based on training set 2. A total of 32 models is obtained, all shown in Table 4.7. Table 4.8 shows the method, training set, and feature set used to obtain each model.

4.4.4 Planning

With a model for the daily call center load forecast, the model still needs to establish the number of operators needed to answer the calls and reach the desired service level. The square root rule, described in Chapter 2 and shown again in Equation 4.1, is able to calculate the number of operators needed based on the load of the call center and a constant as indicator for the desired service level.

$$m = E + \beta \sqrt{E} \ . \tag{4.1}$$

where

$$\begin{aligned} E &= \lambda * h \\ \lambda &= \text{Call arrival rate in minutes} \\ h &= \text{Average call-holding time in minutes} \\ \beta &= \text{Constant depending on desired service level} \end{aligned}$$

Because the daily load is not equally spread across the day, the call arrival rate can not be calculated using only the daily load and the time the call center is open. This would result in insufficient operators for the busy hours. Therefore the call arrival rate is determined based on the number of calls in the 'busy hour', that is the hour of the day with the most calls. Calculations on the years 2010 and 2011 show that the busy hour generally yields approximately 11 % of the daily calls. The average call-holding time over 2010 and 2011 was 5 minutes and 11 seconds. To account for a safety buffer and ring times, an average call-holding time h of 5.5 minutes is used.

Assume a daily number of calls of 400. The busy hour yields $400 * 0.11 = 44$ calls. This is a call arrival rate λ of $44/60 = 0.73$ calls per minute. The calculated values together result in a call center load E of $5.5 * 0.73 = 4.0$ Erlang. Applying the square root rule for safety staffing with a constant β of 0.9 shows that the desired number of operators is $4.0 + 0.11 * \sqrt{4.0} = 5.84$, rounded to 6 operators.

Using this number for the Erlang-C formula, as described in Chapter 2 and shown again in Equation 4.2, yields a probability that a customer has to wait of 11.8 % for 6 operators during the busy hour. Increasing the number of operators obviously decreases this probability. However, Erlang-C assumes that all blocked calls enter the queue and will stay there indefinitely. In reality, a part of these calls will abandon the system and either call back later or abandon the company. Because of this fact the results may not be entirely accurate. Another fact to keep in mind with these calculations is that they are calculating averages. Experiences with different days and times may vary from these averages.

$$P_w(E,m) = \frac{\frac{E^m}{m!}\frac{m}{m-E}}{\sum_{i=0}^{m-1}\frac{E^i}{i!} + \frac{E^m}{m!}\frac{m}{m-E}} \quad . \tag{4.2}$$

where

$$\begin{aligned} P_w &= \text{Probability that a customer has to wait} \\ E &= \text{Offered traffic} \\ m &= \text{Number of servers} \end{aligned}$$

Chapter 5

Results and Discussion

5.1 Forecasting

All developed models are evaluated using two tests, one based on the year 2010 and the other based on 2011. The tests predict each day of the year using the evaluated model built on the 365 days preceding this day. The evaluation is using one day ahead forecasting. All results are shown in Table 5.1, while the best performing models are also shown in the graph of Figure 5.1. The 'clean' model is the SARIMAX model without external parameters besides the default 'holidays' feature. Low RMSE values compared to the RMSE of the clean model are highlighted for separate years in the table.

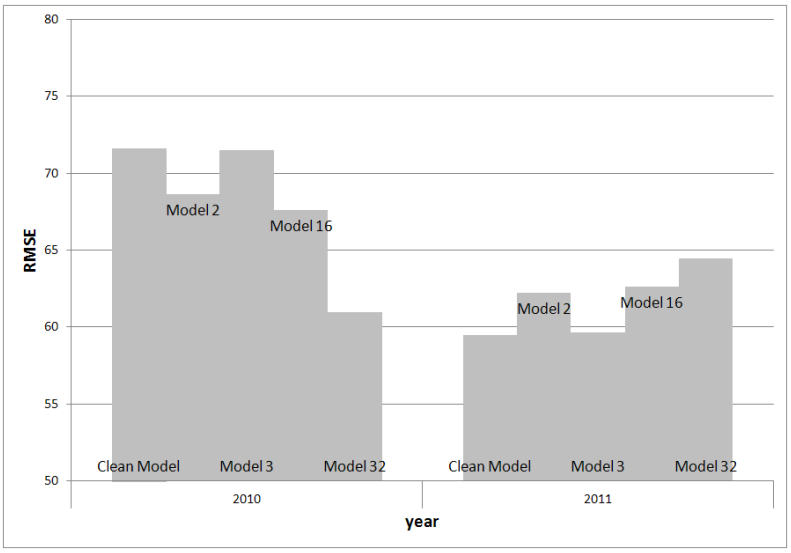

Figure 5.1: RMSE values for the best performing models for 2010 and 2011.

Table 5.1: Evaluation Results

	2010		**2011**	
Model	**RMSE**	**MAPE**	**RMSE**	**MAPE**
Clean	71.59584	0.1965481	59.46052	0.1620854
1	71.60340	0.1964228	**59.73793**	0.1629967
2	**68.63996**	0.2121450	62.20308	0.1712511
3	71.49465	0.1968789	**59.62552**	0.1626546
4	70.78567	0.2088980	60.76275	0.1655302
5	**69.46576**	0.2036352	61.58345	0.1685262
6	**68.99005**	0.2139527	62.34568	0.1704671
7	**68.43640**	0.2169298	62.46222	0.1715471
8	**67.56614**	0.2102386	62.03777	0.1684082
9	70.72288	0.2125537	61.18166	0.1668626
10	**69.53713**	0.2036616	62.01281	0.1684027
11	**69.32407**	0.2092855	62.00606	0.1689659
12	**68.65833**	0.2156885	62.64251	0.1710041
13	**67.50674**	0.2097757	62.68043	0.1705077
14	**67.65980**	0.2124005	62.26051	0.1697821
15	**69.34965**	0.2082018	62.43100	0.1694163
16	**67.57991**	0.2141129	62.63181	0.1691009
17	72.12051	0.1990305	**59.66569**	0.1625560
18	71.75493	0.2054651	**59.95725**	0.1661760
19	71.48609	0.2009431	60.18074	0.1648616
20	72.1323	0.2002387	60.07343	0.1639492
21	72.03634	0.1986477	**59.78743**	0.1646538
22	71.40773	0.2006877	60.03562	0.164716
23	**67.97988**	0.2122895	62.33069	0.1720297
24	**69.10595**	0.2072089	62.55832	0.1700503
25	**70.51194**	0.2195307	61.88039	0.1684595
26	**71.38840**	0.2109367	62.11184	0.1693930
27	72.68947	0.2143197	62.15707	0.1686879
28	73.27861	0.2110278	60.7696	0.1665167
29	73.98268	0.2075295	61.02424	0.1669809
30	**67.32328**	0.2165034	61.94159	0.1646685
31	**67.26591**	0.2192662	61.56716	0.1638218
32	**60.95447**	0.2008072	64.437140	0.1719295

The first thing noted from these results is that none of the models performs better on 2011 than the clean model. Both performance indicators show less forecasting accuracy of the extended models on this year, in varying degree. Several models perform better in forecasting 2010 than the clean model on the RMSE metric. According to this metric some of the manually designed models, along with the models 23 to 25 slightly outperform the clean model, while the models 30 to 32 outperform the clean model by a more substantial amount. However, evaluating models 23 to 25 and 30 to 32 on 2010 is biased because the features of these models are partly determined based on the same year using the ant colony optimization. Omitting these results yields some of the manually obtained models as best performing ones on 2010, especially the models containing the features 'Sky' and 'feature_count'. The model containing both weather-related features does not perform better than the clean model on both years.

The reasons behind the poor performance of the models constructed by the different feature selection algorithms are related to the specific dataset. As already stated in the peak investigation of Chapter 4, the call center load in 2011 contains less clearly delineated peaks which can be directly linked to external events than 2010. Because of this fact, the clean model already performs significantly better on 2011 than on 2010. Therefore models with well-performing features taken from 2010 are not likely to increase the forecasting accuracy of the clean model on 2011.

The Pearson correlations between the load and the manually selected features, shown in Table 5.2, confirm these observations. While both weather-related parameters have low correlations with the load across all three years, most category-features and the 'filter_count' have substantially higher correlations in 2010 than in the other two years. This is in line with the performance of the different models as shown in the results table.

For illustrative purposes an example of the actual load along with the clean prediction and the prediction of the best performing model with manually determined exogenous variables around one of the major peaks is included in Figure 5.2. The shown peak was caused by the outbreak of the Icelandic volcano in April 2010. The time of this graph is measured in days. What can be noted from this graph is that while there is no difference between both predictions of the first major peak where both models perform equally poor, the second major peak is slightly better predicted using the model with the exogenous variables. However, as the shown peak is one of the most extreme ones available in the dataset, it is not strange that the model is not trained well enough to forecast it more precisely.

The performance of categories from the second feature set, where the categorisation is done on filtered news items, suffers from the same problem as the models using feature set 1. This can be noted by comparing the RMSE of models 27 to 29 to the RMSE of models 30 to 32. Models 27 to 29 perform less than the clean model because their feature sets are deduced from the year 2009, while the models are tested on 2010. However, doing the categorisation based on the filtered news items certainly is promising because models 30 to 32 perform significantly better on 2010 than models 23 to 25, which are also created partly based on the year 2010.

Table 5.2: Pearson correlations between call center load and manually selected features

	Pearson Correlation		
Feature	**2009**	**2010**	**2011**
Hostile	0.18	0.24	0.16
Sky	0.03	0.46	0.12
Rise	0.22	0.18	0.22
Travel	0.14	0.31	0.14
Filter_count	0.07	0.40	0.10
Temp	0.10	-0.11	-0.03
Precip	0.09	-0.04	0.06
Holidays	-0.21	-0.19	-0.21

While both tests are biased because they are trained and evaluated on the same years, the higher performance of the filtered categorisation might indicate stronger forecasting possibilities on datasets containing more peaks across all years. Another thing that can be noted is that the improvement of these models on 2010 is larger than the declined performance on 2011.

The results table shows a lack of general agreement between the RMSE and the MAPE metric. Sometimes the MAPE goes up where the RMSE goes down, or vice versa. I focused the evaluation of the models on the RMSE measure because it is more important to keep the accuracy high in terms of absolute calls than in terms of a percentage, considering the fact that each missed call might cause a lost customer. An increase of the MAPE together with a decrease of the RMSE indicates a better forecast on busy days (generally weekdays in the used dataset) and positive peaks and a lower performance on days with a lower call center load. The trend of lowering the RMSE regardless of the MAPE can be explained by the fact that the feature selection algorithms are focused on minimizing the RMSE. The RMSE metric is also considered more important in this study because this study is aimed at predicting positive peaks using news.

While these results do not directly give models that outperform the basic trend models, it does give some pointers for future improvements of call center prediction. Possibilities of this are similar models for different call centers which are more dependent on external events than the studied call center. This follows from the slightly improved accuracy on the year 2010 which has higher correlations to several categories taken from the news. For the call center of Vliegtickets.nl better models might be developed when more data is available. Another option to improve the model is

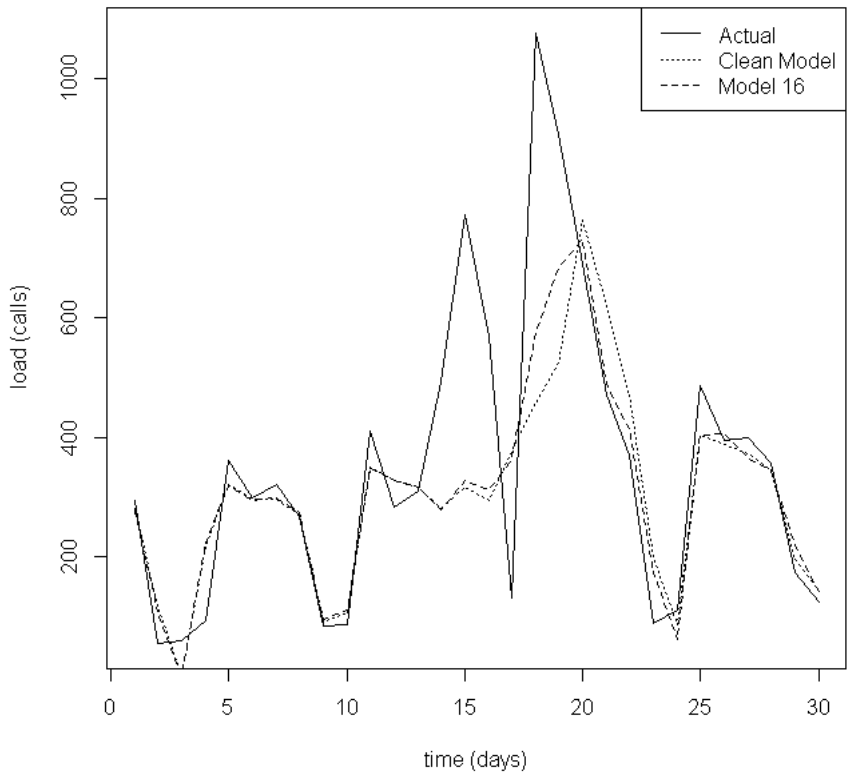

Figure 5.2: Comparison of two predictions along with the actual load for April 2010.

adding a feature indicating closure of airports or delay of flights, or a feature indicating heavy weather.

5.2 Planning

The influence of different forecasting models on the actual operator planning is calculated to check their practical relevance. In the high peak example of April 2010, one day the call center load amounts 1078 calls. The square root rule for safety staffing similar to subsection 4.4.4 yields a necessary number of operators of 14. The clean model forecasts a load of 458 calls, resulting in 7 needed operators. Model 16 forecasts 573 calls, suggesting 8 needed operators. Clearly both forecasting methods result in heavy understaffing, hence further research would be interesting from a practical point of view. However, this example uses the most extreme peak of the used dataset. Other parts of the dataset yield much smaller differences.

Chapter 6

Conclusion

This thesis explored a field that has not been served before in the literature, namely the prediction of daily inbound load of call centers using events text-mined from the news. While basic load prediction can be done using time series models based on historic data, these models do not capture the influence of external events. Many call centers experience a higher number of calls on days following a far-reaching event related to their business field. The call center considered in this study is from Vliegtickets.nl, a Dutch company providing information concerning all kinds of aspects related to flight holidays. The number of incoming calls of their call center is likely to be affected by actual events related to flight traffic. An accurate prediction of the call center load could save money because the company will not hire too many operators being idle, nor will it hire too few operators, resulting in missed sales calls leading to increasing opportunity costs.

The first part of this study consists of an extensive analysis of existing methodologies in literature. Both basic call center planning and more extended forecasting models were discussed. It also includes other forecasting fields which already make use of external parameters such as news. Concluding the related work a discussion of available text analytics techniques and feature selection algorithms was given.

The study continued with a description of the study design where different methods were evaluated with respect to the current case. Following up the actual model design is described. The most optimal seasonal autoregressive integrated moving-average (SARIMA) model was determined using the Box-Jenkins approach. According to this analysis carried out on the years 2009 and 2010 the best performing SARIMA model is the SARIMA (2,0,0) (0,1,1) model with a seasonal frequency of 7 days.

Peaks in call center load were located using the trend model, with as indicator the deviation between the forecasted load and the actual load. The news around large peaks is analyzed, and possible causes are listed. While this manual analysis does not clearly point to certain types of events influencing the call center load, there is evidence that some peaks, especially in 2010, are caused by events related to flight traffic. Another suggestion that rises from the peak investigation is that Dutch public

celebration days generally yield less calls than normal days. Because the study is aimed towards predicting the call center load using external events taken from the news, an external parameter indicating national free days is included by default in the base model. The results in a model that gives an RMSE of 71.6 for the prediction of 2010 and 59.5 for 2011, and a MAPE of 19.6 % for 2010 and 16.2 % for 2011.

Several models were created using features based on the categorisation of the news with the dictionaries of the General Inquirer. The feature pool was completed by a variable indicating the number of news items about flight traffic, and two variables indicating the weather. Both manually selected features like 'sky', 'travel', 'rise', and 'hostile', and automatically selected features by the feature selection algorithms were evaluated. While some of the manually selected models perform better than the clean SARIMA model on 2010, none of the models outperforms the clean model in predicting the call center load of the year 2011. A second feature set containing the categorisation of the filtered news items about flight traffic did not show a different pattern. Features from feature set 1 performing well on predicting the load of 2010 are the categories 'sky', 'travel', 'rise', and the feature containing the counts of daily filtered news items. The two weather-related features performed poorly. The best-performing model on 2010, according to the RMSE, which was not determined based on 2010 is the model including categories 'hostile', 'sky', 'rise', 'travel', 'filter_count', and 'holidays'. This model had an RMSE of 67.6 on 2010, while the clean model yielded an RMSE of 71.6. However, the performance of this model on 2011 is decreased from an RMSE of 59.5 to an RMSE of 62.6.

The performance of models based on both feature sets is probably suffering from the fact that the year 2011 contains less clearly defined peaks related to actual events than 2010, and hence has smaller correlations with most categories from the news. This problem might be solved for the call center of Vliegtickets.nl by a larger dataset containing more peaks. However, the evaluation and improved performance on the year 2010 suggests that the followed approach is valid for call centers with are more influenced by certain news events than this call center.

Possible directions for future work concerning flight traffic-related call centers include adding and evaluating other features such as ones related to the weather. While this study simply considered the daily average temperature and the total daily precipitation, other features indicating heavy weather hindering traffic might perform better. However, designing such features is not trivial and is hard to automate because many aspects, such as snow, temperature, wind, and precipitation, should be considered. Also it is not certain that such features would outperform weather-related categories based on the news, such as 'sky', that are already evaluated in this study, although it is likely that they would remove a lot of unrelated noise contained in the news.

Another possibility to improve load forecasts would be to add a variable that accounts for airport closure or flight delays. Generally events causing large flight delays result in positive call center peaks. However, designing such a variable is again not trivial and would not certainly improve the flight-related categories taken from the news used in this study.

Finally, using a semantic approach instead of the keyword-based approach of this

study might improve the model. While the performance of this study is possibly not optimal due to word ambiguity, a semantic approach would remove this flaw from the models, resulting in using only news items that are truly related to flight traffic.

Bibliography

[1] Khurshid A. The 'Return' and 'Volatility' of Sentiments: An Attempt to Quantify the Behaviour of the Markets? In *Affective Computing and Sentiment Analysis: Metaphor, Ontology, Affect and Terminology*, pages 99–110. Springer London, Limited, 2008.

[2] Z. Aksin, M. Armony, and V. Mehrotra. The Modern Call Center: A Multi-Disciplinary Perspective on Operations Management Research. *Production and Operations Management*, 16(6):665–688, 2007.

[3] A. Al-Ani. Ant Colony Optimization for Feature Subset Selection. In *WEC (2)*, pages 35–38. Enformatika, Çanakkale, Turkey, 2005.

[4] B. Andrews and S. Cunningham. L. L. Bean Improves Call-Center Forecasting. *Interfaces*, 25(6):1–13, 1995.

[5] A. Antipov and N. Meade. Forecasting Call Frequency at a Financial Services Call Centre. *The Journal of the Operational Research Society*, 53(9):953–960, 2002.

[6] S. Asur and B. A. Huberman. Predicting the Future with Social Media. In *Web Intelligence*, pages 492–499. IEEE, 2010.

[7] J. Atlason, M. A. Epelman, and S. G. Henderson. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research*, 127:333–358, 2004.

[8] A. Avramidis, A. Deslauriers, and P. L'Ecuyer. Modeling Daily Arrivals to a Telephone Call Center. *Management Science*, 50(7):896–908, 2004.

[9] R. Battiti. Using Mutual Information for Selecting Features in Supervised Neural Net Learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, 1994.

[10] W. J. Black, J. McNaught, A. Vasilakopoulus, K. Zervanou, B. Theodoulidis, and F. Rinaldi. CAFETIERE: Conceptual Annotations for Facts, Events,

Terms, Individual Entities, and RElations. Technical Report TR-U4.3.1, Department of Computation, UMIST, Manchester, 2005.

[11] G. E. P. Box and G. M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1976.

[12] L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, and L. Zhao. Statistical Analysis of a Telephone Call Center: A Queueing-Science Perspective. *Journal of the American Statistical Association*, 100(469), 2005.

[13] C. L. J. Chen. Traffic Accident Macro Forecast Based on ARIMAX Model. *Measuring Technology and Mechatronics Automation, 2009. ICMTMA '09. International Conference on*, 3:633–636, 2009.

[14] G. Chowdhury. Natural Language Processing. *Annual Review of Information Science and Technology*, 37(1):51–89, 2003.

[15] H. Cunningham. JAPE: a Java Annotation Patterns Engine. Research Memorandum CS-99-06, Department of Computer Science, University of Sheffield, 1999.

[16] H. Cunningham. GATE, a General Architecture for Text Engineering. *Journal Computers and the Humanities*, 36(2):223–254, 2002.

[17] S. R. Das and M. Y. Chen. Yahoo! for Amazon: Sentiment Extraction from Small Talk on the Web. *Management Science*, 53(9):1375–1388, 2007.

[18] A. Deslauriers. Modélisation et Simulation d'un Centre d'Appels Téléphoniques dans un Environnement Mixte. In *Master's Thesis, Department of Computer Science and Operations Research, University of Montreal, Montreal, Quebec, Canada*, 2003.

[19] D. Dickey and W. Fuller. Distribution of the Estimators for Autoregressive Time Series With a Unit Root. *Journal of the American Association*, 74(366):427–431, 1979.

[20] M. Dorigo, V. Maniezzo, and A. Colorni. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, 26(1):29–41, 1996.

[21] T. Engset. Die Wahrscheinlichkeitsrechnung zur Bestimmung der Wahleranzahl in Automatischen Fernsprechamtern. *Elektrotechnische Zeitschrift*, 39(31):304–306, 1918.

[22] A. K. Erlang. On the Rational Determination of the Number of Circuits. *The Life and Works of A. K. Erlang*, 1948.

[23] S. Flank. A Layered Approach to NLP-Based Information Retrieval. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 397–403. Association for Computational Linguistics, 1998.

[24] G. P. C. Fung, J. X. Yu, and W. Lam. News Sensitive Stock Trend Prediction. In *PAKDD*, volume 2336 of *Lecture Notes in Computer Science*, pages 481–493. Springer, 2002.

[25] N Gans, N. Liu, A. Mandelbaum, H. Shen, and H. Ye. Service Times in Call Centers: Agent Heterogeneity and Learning with some Operational Consequences. In *Borrowing Strength: Theory Powering Applications, A Festschrift for Lawrence D. Brown*, pages 99–123, 2010.

[26] O. Garnett, A. Mandelbaum, and M. Reiman. Designing a Call Center with Impatient Customers. *Manufacturing & Service Operations Management*, 4(3), 2002.

[27] M. Genereux, T. Poibeau, and M. Koppel. Sentiment Analysis using Automatically Labeled Financial News. *Health San Francisco*, 2008.

[28] I. Guyon and A. Elisseeff. An Introduction to Variable and Feature Selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[29] J. Hausser and K. Strimmer. *entropy: Entropy and Mutual Information Estimation*, 2012. R package version 1.1.7.

[30] C. Heij, P. De Boer, P.H. Franses, T. Kloek, and H.K. Van Dijk. *Econometric Methods with Applications in Business and Economics*. Oxford Univ. Press, 2004.

[31] F. Hogenboom, A. Hogenboom, F. Frasincar, U. Kaymak, O. van der Meer, K. Schouten, and D. Vandic. SPEED: A Semantics-Based Pipeline for Economic Event Detection. In *ER*, volume 6412 of *Lecture Notes in Computer Science*, pages 452–457. Springer, 2010.

[32] T. Hong and I. Han. Knowledge-Based Data Mining of News Information on the internet using Cognitive Maps and Neural Networks. *Expert Systems with Applications*, 23(1):1–8, 2002.

[33] J. J. Hopfield. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8):2554–2558, 1982.

[34] R. J. Hyndman, S. Razbash, and D. Schmidt. *forecast: Forecasting Functions for Time Series and Linear Models*, 2012. R package version 3.19.

[35] Jaxen Team. Jaxen. from `http://jaxen.codehaus.org/`, 2010.

[36] J. Jewett, J. Shrago, and B. Yomtov. Designing Optimal Voice Networks for Businesses, Government, and Telephone Companies. *Telephony Publishing Corp.*, 1980.

[37] G. Jongbloed and G. Koole. Managing Uncertainty in Call Centres using Poisson Mixtures. *Applied Stochastic Models in Business and Industry*, 17(4):307–318, 2001.

[38] D. Karp, Y. Schabes, M. Zaidel, and D. Egedi. A Freely Available Wide Coverage Morphological Analyzer for English. In *Proceedings of the 14th conference on Computational linguistics - Volume 3*, COLING '92, pages 950–955. Association for Computational Linguistics, 1992.

[39] K. Kira and L. A. Rendell. The Feature Selection Problem: Traditional Methods and a New Algorithm. In *AAAI*, pages 129–134. AAAI Press / The MIT Press, 1992.

[40] Ron Kohavi. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, IJCAI'95, pages 1137–1143. Morgan Kaufmann Publishers Inc., 1995.

[41] G. Koole and A. Mandelbaum. Queueing models of call centers: An introduction. *Annals of Operations Research*, 113:41–59, 2002.

[42] G. M. Ljung and G. E. P. Box. On a Measure of Lack of Fit in Time Series Models. *Biometrika*, 65(2):297–303, 1978.

[43] J. Lovins. Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics*, 11(1), 1986.

[44] V. Mehrotra, O. Ozlük, and R. Saltzman. Intelligent Procedures for Intra-Day Updating of Call Center Agent Schedules. *Production and Operations Management*, 19(3):353–367, 2010.

[45] V. Milea, R. J. Almeida, U. Kaymak, and F. Frasincar. A Fuzzy Model of the MSCI EURO Index Based on Content Analysis of European Central Bank Statements. In *FUZZ-IEEE*, pages 1–7. IEEE, 2010.

[46] M. Mittermayer. Forecasting Intraday Stock Price Trends with Text Mining Techniques. In *HICSS '04: Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 3*, page 30064.2. IEEE Computer Society, 2004.

[47] NOS. NOS Nieuws. `http://nos.nl/nieuws/archief/`, 2012.

[48] C. Palm. Methods of Judging the Annoyance Caused by Congestion. *Tele*, 4:189–208, 1953.

[49] D. Peramunetilleke and R. K. Wong. Currency Exchange Rate Forecasting From News Headlines. In *Thirteenth Australasian Database Conference (ADC2002)*, volume 5. ACS, 2002.

[50] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, and D. Ognyanoff. KIM: A Semantic Platform for Information Extraction and Retrieval. *Journal of Natural Language Processing*, 10(3-4):375–392, 2004.

[51] The JDOM Project. Jdom. from `http://www.jdom.org/`, 2000.

[52] P. Pudil, Novovicová, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15:1119–1125, 1993.

[53] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[54] Reuters. Reuters UK. `http://uk.reuters.com/news/archive`, 2012.

[55] H. Schmid. Probabilistic Part-of-Speech Tagging using Decision Trees. In *Proceedings of International Conference on New Methods in Language Processing*, volume 12, pages 44–49, 1994.

[56] C. E. Shannon. Prediction and Entropy of Printed English. *Bell Systems Technical Journal*, 30:50–64, 1951.

[57] R. Soyer and M. Tarimcilar. Modeling and Analysis of Call Center Arrival Data: A Bayesian Approach. *Management Science*, 54(2):266–278, 2008.

[58] P. J. Stone, D. C. Dunphy, M. S. Smith, and D. M. Ogilvie. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, 1966.

[59] T. Takagi and M. Sugeno. Fuzzy Identification of Systems and its Applications to Modeling and Control. *IEEE Transactions on Systems, Man and Cybernetics*, 15:116–132, 1985.

[60] O. Tanir and R. Booth. Call Center Simulation in Bell Canada. In *Winter Simulation Conference*, pages 1640–1647, 1999.

[61] J. Taylor. A Comparison of Univariate Time Series Methods for Forecasting Intraday Arrivals at a Call Center. *Management Science*, 54:253–265, 2008.

[62] P. Tetlock. Giving Content to Investor Sentiment: The Role of Media in the Stock Market. *The Journal of Finance*, 57(3):1139–1168, 2007.

[63] E. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147. Association for Computational Linguistics, 2003.

[64] K. Tomanek, J. Wermter, and U. Hahn. Sentence and Token Splitting Based on Conditional Random Fields. In *PACLING 2007 - Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 49–57, 2007.

[65] F. Tsenga, H. Yub, and G. Tzengc. Combining Neural Network Model with Seasonal Time Series ARIMA Model. *Technological Forecasting & Social Change*, 69:71–87, 2002.

[66] S. Urbanek. *rJava: Low-level R to Java interface*, 2011. R package version 0.9-3.

[67] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[68] W3C. XML Path Language (XPath). from `http://www.w3.org/TR/xpath/`, 1999.

[69] J. Webster and C. Kit. Tokenization as the Initial Phase in NLP. In *Proceedings of the 14th conference on Computational linguistics - Volume 4*, COLING '92, pages 1106–1110. Association for Computational Linguistics, 1992.

[70] W. Whitt. Understanding the Efficiency of Multi-Server Service Systems. *Management Science*, 38(5):708–723, 1992.

[71] W. Whitt. Dynamic Staffing in a Telephone Call Center Aiming to Immediately Answer All Calls. *Operations Research Letters*, 24(5):205–212, 1999.

[72] W. Whitt. A Diffusion Approximation for the G/GI/n/m Queue. *Operations Research*, 52(6):922–941, 2004.

[73] B. Wüthrich. Discovering Probabilistic Decision Rules. *Int. Syst. in Accounting, Finance and Management*, 6(4):269–277, 1997.

[74] Yiming Yang and Jan O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420. Morgan Kaufmann Publishers Inc., 1997.

[75] D. Yarowsky. Word-sense Disambiguation using Statistical Models of Roget's Categories Trained on Large Corpora. In *Proceedings of the 14th conference on Computational linguistics - Volume 2*, COLING '92, pages 454–460. Association for Computational Linguistics, 1992.

[76] L. A. Zadeh. Fuzzy Sets. *Information and Control*, 8:338–353, 1965.

[77] G. Zeng. On the Convergence of the Extended Erlang-B Model. *Mathematical and Computer Modelling*, 52(1-2):128–133, 2010.

Appendix A

SARIMA Parameter Estimation

Table A.1: Dickey-Fuller statistics for different days of the week

Day	**DF Statistic**
Sunday	-9.1052
Monday	-10.0671
Tuesday	-9.0103
Wednesday	-8.7969
Thursday	-11.1112
Friday	-9.3286
Saturday	-12.1662

Table A.2: MAPE and RMSE of a selection of (S)ARIMA models

Model	**MAPE**	**RMSE**
ARIMA(1,0,0)	0.6893205	125.4069
ARIMA(0,0,1)	0.6626713	122.6503
ARIMA(1,0,1)	0.6716172	124.4566
ARIMA(1,1,0)	0.5960528	147.0444
ARIMA(0,1,1)	0.7903393	134.3845
ARIMA(1,1,1)	0.6848284	128.1025
SARIMA(0,0,0)(1,0,0)	0.40553	107.3186
SARIMA(0,0,0)(0,0,1)	0.6091221	117.2814
SARIMA(0,0,0)(1,0,1)	0.3881951	226.7999
SARIMA(0,0,0)(1,1,0)	0.283699	107.2019
SARIMA(0,0,0)(0,1,1)	0.2611378	92.69493
SARIMA(0,0,0)(1,1,1)	0.2539103	93.21731
SARIMA(1,0,0)(1,0,0)	0.3462678	96.2442
SARIMA(1,0,0)(0,0,1)	0.5215151	107.4959
SARIMA(1,0,0)(1,0,1)	**0.2498855**	**89.52293**
SARIMA(1,0,0)(1,1,0)	0.2610529	95.32604
SARIMA(1,0,0)(0,1,1)	**0.2116746**	**75.95171**
SARIMA(1,0,0)(1,1,1)	**0.2110769**	**78.09946**
SARIMA(0,0,1)(1,0,0)	0.3634658	98.47098
SARIMA(0,0,1)(0,0,1)	0.5169031	106.849
SARIMA(0,0,1)(1,0,1)	0.290225	117.9785
SARIMA(0,0,1)(1,1,0)	0.2670421	99.1672
SARIMA(0,0,1)(0,1,1)	**0.2309432**	**81.70059**
SARIMA(0,0,1)(1,1,1)	**0.2305064**	**83.3006**
SARIMA(1,0,1)(1,0,0)	0.3460593	97.665
SARIMA(1,0,1)(0,0,1)	0.5143663	106.9366
SARIMA(1,0,1)(1,0,1)	0.3101831	109.224
SARIMA(1,0,1)(1,1,0)	0.2641394	94.3284
SARIMA(1,0,1)(0,1,1)	**0.2113326**	**75.66693**
SARIMA(1,0,1)(1,1,1)	**0.2125085**	**77.44316**
SARIMA(1,1,1)(1,0,0)	0.3397523	100.1561
SARIMA(1,1,1)(0,0,1)	0.5150296	108.9306
SARIMA(1,1,1)(1,0,1)	**0.24538**	**82.33439**
SARIMA(1,1,1)(1,1,0)	0.2807013	100.1358
SARIMA(1,1,1)(0,1,1)	**0.229874**	**80.04783**
SARIMA(1,1,1)(1,1,1)	**0.2279868**	**80.77901**

Table A.3: MAPE and RMSE of SARIMA models including 2 non-seasonal autoregressive terms

Model	**MAPE**	**RMSE**
SARIMA(2,0,0)(0,1,1)	0.2090753	75.29462
SARIMA(2,0,0)(1,1,1)	0.2097894	77.33686
SARIMA(2,0,1)(0,1,1)	0.2132632	76.41703
SARIMA(2,0,1)(1,1,1)	0.2127058	78.66309

Table A.4: MAPE and RMSE of SARIMA models including multiple non-seasonal autoregressive terms

Model	**MAPE**	**RMSE**
SARIMA(3,0,0)(0,1,1)	0.2133153	76.69787
SARIMA(3,0,0)(1,1,1)	0.2087394	77.55484
SARIMA(3,0,1)(0,1,1)	0.2131227	76.42688
SARIMA(3,0,1)(1,1,1)	0.2079392	76.25195
SARIMA(4,0,0)(0,1,1)	0.2136701	77.08966
SARIMA(4,0,0)(1,1,1)	0.2088471	77.17783
SARIMA(4,0,1)(0,1,1)	0.2171183	77.91481
SARIMA(4,0,1)(1,1,1)	0.2067771	77.86687

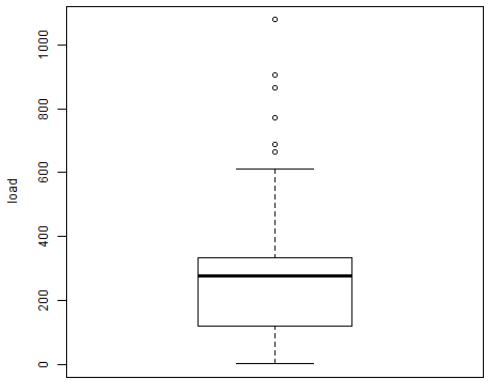

Figure A.1: Boxplot of call center load.

Table A.5: Evaluation results of SARIMA models used on data without outliers

Model	**MAPE**	**RMSE**
ARIMA(1,0,0)	0.7008772	113.4496
ARIMA(0,0,1)	0.6629248	111.2968
ARIMA(1,0,1)	0.6888684	114.7808
ARIMA(1,1,0)	0.5941554	137.7383
ARIMA(0,1,1)	0.7792771	119.7029
ARIMA(1,1,1)	0.694887	116.6927
SARIMA(0,0,0)(1,0,0)	0.3462252	83.78561
SARIMA(0,0,0)(0,0,1)	0.5862809	99.60654
SARIMA(0,0,0)(1,0,1)	0.2835776	75.72082
SARIMA(0,0,0)(1,1,0)	0.2628542	77.81278
SARIMA(0,0,0)(0,1,1)	0.2478369	70.06906
SARIMA(0,0,0)(1,1,1)	0.2445867	70.23764
SARIMA(1,0,0)(1,0,0)	0.3314573	81.36109
SARIMA(1,0,0)(0,0,1)	0.5417161	96.53065
SARIMA(1,0,0)(1,0,1)	**0.2480042**	**70.34605**
SARIMA(1,0,0)(1,1,0)	0.2509501	74.86837
SARIMA(1,0,0)(0,1,1)	**0.2255927**	**64.94344**
SARIMA(1,0,0)(1,1,1)	**0.2231654**	**65.45358**
SARIMA(0,0,1)(1,0,0)	0.3342124	81.57396
SARIMA(0,0,1)(0,0,1)	0.5261629	95.79017
SARIMA(0,0,1)(1,0,1)	0.258876	71.42611
SARIMA(0,0,1)(1,1,0)	0.2535868	75.61946
SARIMA(0,0,1)(0,1,1)	**0.2309623**	**66.19729**
SARIMA(0,0,1)(1,1,1)	**0.2309882**	**66.89356**
SARIMA(1,0,1)(1,0,0)	0.3314468	81.57476
SARIMA(1,0,1)(0,0,1)	0.534437	96.24246
SARIMA(1,0,1)(1,0,1)	0.238362	67.93216
SARIMA(1,0,1)(1,1,0)	0.2509468	74.34459
SARIMA(1,0,1)(0,1,1)	**0.2153629**	**63.58114**
SARIMA(1,0,1)(1,1,1)	**0.2150861**	**64.13096**
SARIMA(1,1,1)(1,0,0)	0.3205444	81.93425
SARIMA(1,1,1)(0,0,1)	0.5366594	98.73243
SARIMA(1,1,1)(1,0,1)	**0.234173**	**67.15837**
SARIMA(1,1,1)(1,1,0)	0.2580783	76.33686
SARIMA(1,1,1)(0,1,1)	**0.223086**	**65.01443**
SARIMA(1,1,1)(1,1,1)	**0.2160622**	**64.94598**
SARIMA(2,0,0)(0,1,1)	**0.2142393**	**63.44967**
SARIMA(2,0,0)(1,1,1)	**0.2125659**	**63.76054**
SARIMA(2,0,1)(0,1,1)	**0.2171228**	**63.97799**
SARIMA(2,0,1)(1,1,1)	**0.2121876**	**63.96319**
SARIMA(3,0,0)(0,1,1)	**0.2183875**	**64.03588**
SARIMA(3,0,0)(1,1,1)	**0.2118142**	**63.47703**
SARIMA(3,0,1)(0,1,1)	**0.216793**	**63.93531**
SARIMA(3,0,1)(1,1,1)	**0.2079606**	**62.4146**
SARIMA(4,0,0)(0,1,1)	**0.218922**	**64.31564**
SARIMA(4,0,0)(1,1,1)	**0.2112321**	**63.23167**
SARIMA(4,0,1)(0,1,1)	**0.2199235**	**64.6454**
SARIMA(4,0,1)(1,1,1)	**0.2048905**	**61.93575**