Universidad Politécnica de Madrid

Escuela Técnica Superior de Digital Innovation

Máster Universitario en Digital Innovation

Master Thesis

Hotel Room Demand Forecasting

Time series forecasting using SARIMA, LSTM and Prophet

Author: **Thomas van Dommelen**

Supervisor: **Ernestina Menasalvas**

Madrid, July 2021

Abstract

Problem statement: Hotel industries have always struggled with predicting the right price for the right room at the right time. With Covid19 hitting the hospitality sector, it is even more difficult to make forecasts. Hotels in Stockholm have this problem as well. Therefore this research focuses on finding: which algorithm between SARIMA, LSTM and Facebook Prophet performs best in forecasting hotel room demand in stockholm. **Academic/Practical Relevance:** This research helps in creating a better understanding of the algorithms and which ones are better to apply in the hotel industry. It shows the differences between the algorithms and makes decision making easier for data scientists in the hotel industry. **Methodology:** The root mean squared error is used as performance measurement to compare three different algorithms; SARIMA, LSTM and Prophet. Lastly predictions graphs are given to see the comparison between actual and predicted values. **Results:** By comparing all the calculated mean squared errors, the best performed algorithm is SARIMA. But taking into account other factors a different approach might be preferred. Although the root mean squared error of LSTM is lower, the algorithm can handle more complex data and is able to consider more variables, which makes the algorithm more reliable. The Prophet model performed worse then both the SARIMA and the LSTM, but is much easier to implement and more userfriendly for inexperienced data scientists. **Managerial implications:** Hotel revenue managers in Stockholm will be able to choose the right algorithm for their data which they can use to implemented and adjust prices for hotel rooms accordingly. This ultimately results in more revenue and profits.

Resumen

ResumenEnunciado del problema: Las industrias hoteleras siempre han tenido problemas para predecir el precio correcto de la habitación adecuada en el momento adecuado. Con la llegada del Covid19 el sector hoteleroes aún más difícil realizar predicciones. Los hoteles de Estocolmo también tienen este problema. Por lo tanto, esta investigación se centra en encontrar: qué algoritmo entre SARIMA, LSTM y Facebook Prophet funciona mejor en la previsión de la demanda de habitaciones de hotel en Estocolmo. **Relevancia académica/práctica:** Esta investigación ayuda a crear una mejor comprensión de los algoritmos y cuáles son mejores para aplicar en la industria hotelera. Además, es capaz the mostrar las diferencias entre los algoritmos y facilitar la toma de decisiones a los analistas de datos del sector hotelero. **Metodología:** El error cuadrático medio se utiliza como medida de rendimiento para comparar tres algoritmos diferentes: SARIMA, LSTM y Prophet. Por último, se presentan gráficos de predicciones para ver la comparación entre los valores reales y los predecidos. **Resultados:** Al comparar todos los errores medios cuadrados calculados, el algoritmo con mejores resultados es SARIMA. Pero después de tener en cuenta otros factores, se podría preferir un enfoque diferente. Aunque el error cuadrático medio de LSTM es menor, el algoritmo puede manejar datos más complejos y es capaz de considerar más variables, lo que hace que el algoritmo sea más fiable. El modelo Prophet obtuvo peores resultados que el SARIMA y el LSTM, pero es mucho más fácil de implementar y más fácil de usar para los analistas de datos con poca experiencia en el sectoy. **Implicaciones para la gestión:** los gestores financieros de los hoteles de Estocolmo podrán elegir el algoritmo adecuado para sus datos que podrán utilizar para implementar y ajustar los precios de las habitaciones del hotel en consecuencia. Lo que se traduce en más ingresos y beneficios.

Acknowledgement

This thesis has been written to fulfill the requirements of the Master Digital Innovation at Universidad Politécna de Madrid. The thesis consists of an internship which was conducted from February until July 2021 at Taktikon. During this internship the goal was to combine research with practical problems.
I would like to express my gratitude to my thesis supervisor, Ernestina Menasalvas for helping and guiding me during this period. I would like to thank her for her feedback and inputs on this thesis. Furthermore I would like to thank Mats Grähs for being my supervisor at the company. Next I would like to thank Annemarie Gubanski as head of the company for allowing me to work on my thesis there.
Lastly I would like to thank UPM, UNS and the EIT Digital Masters for giving me the opportunity of studying and improving my knowledge for the last two years which helped me greatly in attaining enough knowledge and expertise to complete this thesis.

Contents

List of Tables

List of Figures

List of Equations

1 Introduction

More then a year after the first case of Covid-19 appeared in Europe, everyone around the world is still struggling with the virus and its impact on society as we know it. One of the hardest hit sectors around the world is the hospitality sector. Tourism has been extremely low for the past year due to travel restrictions, lockdowns and fear for contracting the virus. In Sweden, the restrictions have been less strict in comparison to other European countries, there the effects of Covid-19 are clearly visible. Especially in the hospitality (and in particular the hotel) market. In the hotel sector one of the most important factors of sustaining profitability is hotel revenue management. Hotel revenue management is all about optimizing the revenues of your hotel. Academic journals have been written to stress the importance of revenue management in hotel industries: *Journal of Revenue and Pricing Management* (2002) [1] and *International Journal of Revenue Management* (2007) [2]. Interesting to see is that there are multiple deciding factors in optimizing the hotel revenue management, but one of them stands out in particular. This is the optimizing and forecasting of hotel room demand. Forecasting hotel room demand means predicting how many rooms are in demand at any given day in the year. If hotels or hotel revenue managers knew the exact demand they would be able to optimise their revenue by adjusting their prices accordingly. Hotel room demand is different everywhere in the world and influenced by numerous factors. This is why there is no 'best' way of optimizing hotel room demand forecasting. Factors like location, culture, weather and more play too large a roll in determining optimal hotel revenue management. This is why in this thesis the focus is on optimizing the hotel room demand in Sweden and in particular in Stockholm. The supply and demand for hotel rooms in Sweden and in particular in Stockholm has been increasing until Covid19 set in. The increased supply of hotel rooms has made it difficult for the industry to handle the low amount of demand of the past year. Before that it was difficult to maintain the opposite, the high demand of hotel rooms. The change of the past year should return to normal when Covid-19 is no longer an issue around the world. But according an article by Mc Kinsey[3], the recovery of the hotel industry to pre-Covid-19 levels will take until 2023 –or later. Their research is however about the US hotel industry. Although Sweden's hotel industry has been taking a smaller hit, it is still reason for concern. Until this happens, however, hotels will have difficulty estimating how many rooms they are going to rent out, when they're going to rent them out and at what price. To forecast the estimation of demand of rooms, some hotel revenue management companies are using different algorithms to make predictions on time series data. This thesis will present a number of algorithms to make time series forecasts and compare them to see which one works best on the given data. This will help hotel revenue management companies to choose a preferred algorithm, knowing the pros and cons of such algorithm better. The objective is not necessarily to optimize hotel revenue management across the world, but especially see if it is possible to find the best algorithm on data concerning hotels in Stockholm. The thesis is in assignment of the company Taktikon. This is a company which dedicates itself to hotel revenue management in Stockholm and this will help them in their decision making. Over the years a lot of algorithms have proved useful in forecasting time series data in the hotel revenue management industry. A hotel revenue management report about Norwegian hotels concluded: *"The empirical work with the Norwegian hotel guest series reveals that different model specifications are useful*

for different purposes." [9]. This suggests more research and models need to be done and implemented. Another article [4] mentions: *"A lot of the work done on the hotel revenue management systems deal with the optimization problem [5][6][7]. There has, however, been little or no published work on the forecasting aspect."*. This also indicates more research can be done on this subject. Of course this was a while ago and the last few years there have been more reports on the forecasting of hotel room demand [8]. Because there have been many different models presented over the last few years it is important to compare some of them to find out which ones work best on particular hotel markets. That is why this thesis will focus on three different algorithms to find which one works best and should be used by hotel revenue managers in Stockholm and/or Sweden to optimize their predictions of demand. The algorithms which will be implemented and compared to eachother are: SARIMA, LSTM and Prophet. This thesis will focus on solving the following research question: **Which algorithm between SARIMA, LSTM and Prophet performs best in forecasting hotel room demand in Stockholm.**

2 Theoretical framework

In this section everything about hotel revenue management will be explained for creating and understanding this thesis. Furthermore a general explanation about time series will be given to present the state of art in this field.

2.1 Hotel revenue management

In the hotel industry the agreed definition of hotel revenue management is: *"Selling the right room, to the right client, at the right moment, for the right price, through the right distribution channel, with the best cost efficiency".* One of the most important aspects of hotel revenue management is estimating the demand. The demand means the amount of rooms which are needed to be available in the future, based on how many customers a hotel is going to receive. The demand on the hotel markets vary a lot dependent on season, week days, events and other factors. Knowing what the demand for hotel rooms in a certain market will be in advance helps the hotel operator with for instance staffing and setting the correct price of the rooms. This saves costs and increases income.

2.2 Hotel industry in Stockholm

The research in this thesis will regard data from the hotel industry in Stockholm. It is therefore important to first explain more about its market. In March 2020 Invest Stockholm released a report [10] which gives a summary about the hotel market and forecasts for the coming few years. The report was created before the problematic decline in demand due to Covid-19 and might not be as accurate in some aspects as the writer had hoped. It is still an important report which helped in analysing and predicting the demand for rooms in the hotel market. An interesting thing mentioned in the report about the demand is: *"In addition, the county´s business structure is largely made up of sectors that normally have a high demand for hotel rooms, such as companies in Information and Communication, Business services, Banking and Insurance. Many large international companies also have their headquarters in Stockholm."* This is interesting, because this means the hotel market is not only important during weekends (in most smaller touristic cities), but also important during weekdays. This is could be one of the reasons the hospitality sector in Stockholm has taken less of a hit compared to other countries. It could also imply that the market might stabilise faster and start growing again soon.

2.3 Taktikon

Taktikon is the company who provided the assignment and the data necessary for conducting this thesis. Taktikon was founded in 2010 by Annemarie Gubanski. Her vision was to enable companies to grow their business and to help people to grow their knowledge about Revenue Management & Distribution. Today the company is working on many things, one of them being the outsourcing of consultancy on hotel revenue management. They have a platform which gives an overview of hotels per market niche. On this platform a hotel within a specific market can view and analyze their direct competitors and see how they performed against each other. This provides a lot of data which in turn can be used to predict the demand for hotel rooms, hence this thesis. The company currently uses a

random forest model to predict the demand for their markets, but they would like to see how other models perform. The goal of Taktikon is improving, optimizing or implementing a better model. The accuracy of their current model is 98% which makes the fitting of the model questionable. Furthermore, the model has proven not to be completely accurate in practical cases thus far.

2.4 Time series forecasting

In section 5 it is clearly visible that the data concerns a time series data set. Therefore, this section is to find and explore options to analyse and forecast time series. In "The Visual Display of Quantitative Information." E. Tufte said: *"The time-series plot is the most frequently used form of graphic design. With one dimension marching along to the regular rhythm of seconds, minutes, hours, days, weeks, months, years, or millennia, the natural ordering of the time scale gives this design a strength and efficiency of interpretation found in no other graphic arrangement."* [11] Although the book is quite dated by now (originally published in 1983) the time-series plot is still relevant. To this day this type of data can be found anywhere. In for example; financial data, weather data, social data, hospitality data and more. The book also shows graphic images of time series plots dating back to the 19th century. With the first found time-series plot dating back to possibly eleven hundred years ago. This shows the relevance of time-series up to now. Because of the long time use of time series there are a number of ways and models in existence to analyse and predict time series data. One of the most important and widely used time series models is the autoregressive integrated moving average (ARIMA) model [14]. This model was first introduced by Box and Jenkins in 1976 [13]. Some variations of this model are the SARIMA and SARIMAX model, which both include seasonal components found in time series data and the latter containing external factors which could influence the data. Because of the widely known effective usage of this model it will be the first model done in this thesis.
Zhang mentiones another approach which is suitable for time series forecasting, which is artificial neural networks (ANN). According to Sagheer and Kotb [16] the best ANN approach for time series forecasting would be a recurrent neural network approach (RNN). But it also mentions there is one weakness in RNN during the requirement of learning long-range time dependencies.[17] In 1997 however Hochreiter and Schmidhuber [18] came up with long short-term memory to make up for this. The second model we will be exploring for our data set will be a LSTM model.
Another interesting approach to forecasting time series is the relatively new Facebook Prophet model. This model became available to the public in 2017. The article 'Forecasting at scale' [20], which came out with Prophet to clarify the use and possibilities of the model mentions two components. The second one being the most important: "The second component is a system for measuring and tracking forecast accuracy, and flagging forecasts that should be checked manually to help analysts make incremental improvements.". This is a vital component that helps analysts to determine when the model needs to be adjusted or when a different model is required. In 2020 both the models of LSTM and prophet were used and compared to predict air temperature [39], the conclusion was that both models performed well on different aspects. Therefore, it is good to implement both models.

3 Materials and methods

This section contains information on all the materials and methods used in this thesis. First an overview of all the data sets is given. This includes a short explanation on how the data sets were cleaned and which information was used and which information was omitted. Next, is a short overview of the programs used. Lastly, is an overview of all the models that were used including a description of how each of these models work in theory.

3.1 Data sets

The first data set is a combined data set of daily reports in a hotel market niche from April 2020 to May 2021. This particular niche regards a market from different hotels in Stockholm. The data from March 2021 until May 2021 is about rooms that have already been booked (even though the day has not arrived yet). This makes the data from March 2021 until May 2021 unreliable and therefore it was decided to not take this data into account for now. The data also includes several columns which are not important. All the columns are listed below in table 1, with a short description and whether or not they will be used in the rest of the analysis. Another important note to mention is that rows which contain a price value which is impossible (for example a standard room which costs more than 10.000 euro) will be deleted.

Column	Description	Kept or omitted
Id	Non-unique Id 0 for all rows	Omitted
Pax	Non-unique Id referring to market nr. 2	Omitted
UniqueID	Unique Id different for all rows	Kept
City	City of hotel, Stockholm for all rows	Omitted
CheckInDate	Check in date	Kept
Date_Retrieved	Date, when the data has been collected	Kept
Price	Price of the room in SEK	Kept
RoomType	Type of room	Kept
RoomName	Name of room (named by hotels themselves)	Omitted
Cancellation	True or False whether room has been cancelled	Kept
OTA	Booking platform	Omitted
Description	Blank for all rows	Omitted
ImportDate	Same as Date_Retrieved	Omitted

Table 1: Columns of first data set including description and whether or not to keep them

The second data set used is a collected bench marking report. In this data is the comparison of hotel demand and supply (the supply is mostly just the maximum number of hotel rooms, thus the capacity of the hotel) between two years. The data dates back

to 2017 (and with that its comparison with the year before 2016), until March 2021. This data set does not only contain the comparison between years, but also between competitors. The data set regards one hotel and compares it with its direct competitors in the same market. It also gives a ranking each day on how well it performed compared to its competitors. Furthermore, it shows the difference in revenue and demand. The most important data for now however is the daily income and daily demand of the hotel. Below is a short table on the data set and its columns. The demand of the hotel rooms is called PropDemTY and PropDemLY. This stands for property demand this year and property demand last year. These and especially the one from this year are the most important columns, because these are the columns which need to be predicted. When a hotel knows its future forecast for the amount of hotels will be rented, it can anticipate and change prices accordingly which results in higher revenues and profits. The second most import columns are the date columns, these are are important to make sure the forecast is made accurately.

Column	Description
DateTY	The date this year
DateLY	Same date the year before
PropDemTY	Demand of rooms this year
PropRevTY	Revenue this year
PropDemLY	Demand of rooms last year
PropRevLY	Revenue last year
CompSupTY	Supply of rooms by competitors this year
CompDemTY	Demand of rooms by competitors this year
CompRevTY	Revenue by competitors this year
CompSupLY	Supply of rooms by competitors last year
CompDemLY	Demand of rooms by competitors last year
CompRevLY	Revenue by competitors last year

Table 2: Important columns of second data set

There were a number of columns which contained identical numbers for each row in the column and they were thus ommited. These columns were: HotelID, PropID, Currency, PropSupTY, PropSupLY (The supply is consistently 144 for each day, the number of available rooms), TotalTY, TotalLY, ComplianceTY, ComplianceLY and ComparisonGroup (only one and the same group will be analysed).

3.2 Programs

All of the exploring and analysis is done in python. The data sets are both uploaded and formatted in python. All of the following machine learning methods etcetera are also done

in Python. The program used by Taktikon is Visual Studion with C#. The company can implement any algorithms (if needed) into Visual Studion.

3.3 Models

In this section is a further explanation of each of the models used in this thesis. The models which are explained are SARIMA, LSTM and Prophet respectively.

3.3.1 SARIMA

The first used model is a SARIMA (Seasonal Auto Regressive Integrated Moving Average) model. In section .. it was mentioned that the SARIMA model was derived from the ARMA model first introduced by Box and Jenkins [13]. The ARMA model is short for auto regressive moving average model; ARMA(p,q)(AR of order p and MA of order q). A stationary time series $\{X_t\}$ follows this model if the following difference equation is statisfied [22].

$$X_t - \alpha_1 X_{t-1} - ... - \alpha_p X_{p-t} = \varepsilon_t + \beta_1 \varepsilon_{t-1} + ... + \beta_p \varepsilon_{p-t} \tag{1}$$

Model (1) can be written as:

$$A(L)X_t = B(L)\varepsilon_t \tag{2}$$

Where A(L) indicates the AR operator and B(L) indicates the MA operator. Box and Jenkins [13] also said that in the case of non-stationary time series (which is in most cases) differencing can be used to make the series stationary. If we take d as the minimum degree of differencing and following differenced model $\{\nabla^d X_t\}$ satisfies the first model (1), then the model follows an ARIMA(p,d,q) model (autoregressive integrated moving average with respectively the orders p, d and q). Furthermore, Box and Jenkins also mentioned that if the original series $\{X_t\}$ also includes a seasonal component s, the new model is:

$$A(L)\Phi(L^s)\nabla^d\nabla^D X_t = B(L)\Theta(L^s)\varepsilon_t \tag{3}$$

$\Phi(L)$ is the seasonal AR operator and $\Theta(L)$ is the seasonal MA operator. If these are polynomials of order P, Q nad L respectively and model (3) is invertible and stationary, the series $\{X_t\}$ follows a SARIMA(p,d,q)(P,D,Q,s) model (seasonal autoregressive integrated moving average with respectively the orders p, d, q, P, D, Q and s). If the integrated part is not necessary (i.e. the series is stationary), then SARIMA can still be used with parameter I = 0. This is done to still be able to use the seasonal components of the model. In conclusion the following model is used:

$$SARIMA(p,d,q)(P,D,Q,s) \tag{4}$$

With:

p: The order of the non-seasonal autoregressive model.
d: The number of non-seasonal differences.
q: The order of non-seasonal moving average model.
P: The order of seasonal autoregressive model.
D: The number of seasonal differences.
Q: The order of seasonal moving average model.
s: The periodic term.

SARIMA is an effective linear model to effectively capture the trend of seasonal the seasonal series [23]. But it is limited in that it can't effectively predict with more variables and if there are few linear dependencies in the data. It is also most effective on stationary data with few missing data points. In the case the data is non-stationary and has a lot of missing data points or a lot of outliers the model performs poorly. Another shortcoming of the SARIMA model might be that it is difficult to include conditional seasonality. Meaning seasonality which depends on certain conditions. That is why it is important to look for more then just the SARIMA model in predicting time series.

3.3.2 LSTM

The next model is LSTM (long short term memory). This model is a type of RNN (recurrent neural network), which is capable remembering previous steps in a series for future use and predictions. An artificial neural network (or neural nets in short) is a computing system, which mimics the way the brain works. The neural part stands for the neurons in the brain and the network for the structure. These neurons communicate to each other through this structure. Because a NN closely follows the way the brain is modeled a great deal of vocabulary comes from neurology. Dongare [24] gave a good overview of some of the vocabulary used as seen in table 3.

Biological Vocabulary	**ANN Vocabulary**
Neuron	Node/Unit/Cell/Neurode
Synapse	Connection/Edge/Link
Synaptic Efficiancy	Connection Strength/Weight
Firing Frequency	Node Output

Table 3: Vocabulary of neuron

In short it could be said table 3 has all the components of a neural network, except for the hidden layers. The hidden layer represent "hidden" neurons which is located between the input and output layers, where the function applies the given weights to the output. In essence, the hidden layers apply nonlinear operations on the network's inputs. Graph 2 shows an image of how a simple network works with only one hidden layer.

The arrows before the first nodes represent the inputs, the arrows between the nodes represent the weights and the arrows after the last nodes represent the outputs. The first nodes are the input layers. The second set of nodes the hidden layers which perform the nonlinear transformation and the last nodes the output layer which transforms into outputs.

A RNN is slightly different from an ANN. In RNN the connections form a directed graph along a temporal sequence. The hidden layer receives its output back for the preceding outputs. The output of a neuron at time t is used as input at time $t + 1$. A long short-term memory (LSTM) is a type of RNN which is very powerful at making predictions based on time series data [26]. In this model, the nodes in a hidden layer can be either a normal node or a LSTM cell. A normal node only passes its input to its output using the corresponding weight. On the other hand LSTM cells are able to remember and/or forget specific type of patterns that have occurred over time due to the extra components in the node. LSTMs can learn at a much faster rate than normal neural networks. A problem with using recurrent neural nets on time series data is called the vanishing gradient problem. RNNs basically have to remember which data has to be kept and used to create an output. But this means for long data sets the RNN has to remember a lot of previous data and long sequences. In a long sequence of hidden layers inputs and outputs of each layer are influenced by inputs and outputs which are closeby, thus given a very long sequence the last outputs are hardly (almost not all) influenced by the first inputs. For instance, with "backpropagation through time" [27] or "Real-Time Recurrent Learning"

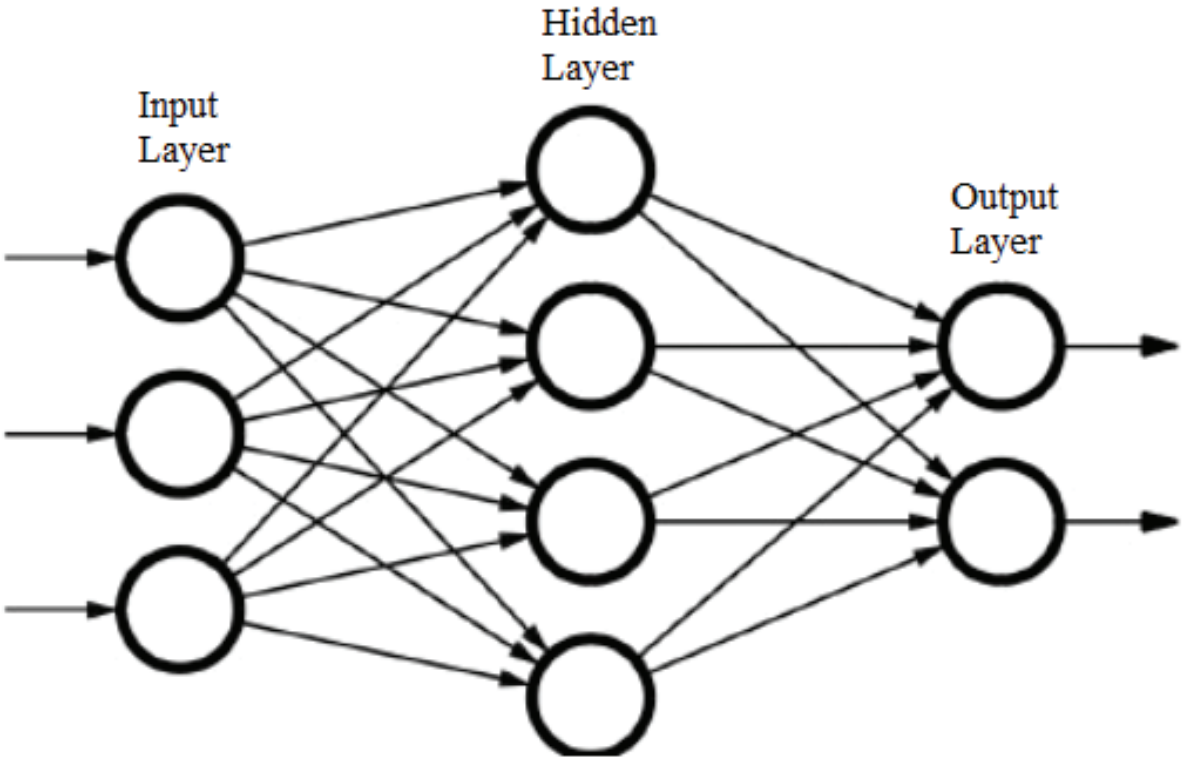


Figure 2: Simple neural network with one hidden layer

[36], error signals flowing backwards in time tend to vanish. Long-term dependencies are hard to learn because of insufficient weight changes [25]. To make up for the vanishing gradient problem LSTM models make use of (next to an input and output gate) a forget gate. These gates form a cell and the three gates control the flow of information into and out of the cell, the cell remembers values across arbitrary time intervals.

Below are a number of steps to simplify the workings of a neural network (in this case the LSTM):

Forward propagation: The input is first fed through the LSTM cell. Next, the input is passed through the hidden states and finally an output is received. This output is given at the end or at each time step. This depends on which type of problem it is about (i.e. regression, classification, forecasting, etc.).

Error calculation: The error between the predicted value and the actual value is calculated after receiving the output. This is done using a loss function. The most commonly used loss functions for LSTM in the case of regression are the mean squared error (MSE) and the mean absolute error (MAE). The MSE is calculated as follows 5:

$$MSE = \sum_{i=1}^{N} \frac{(Y_i - \hat{Y}_i)^2}{N} \tag{5}$$

where N is the number of non-missing data points, x_i are actual observations time series

and $\hat{Y}_i$ the predicted values.

The MAE is calculated using the following equation 6:

$$MAE = \sum_{i=1}^{N} \frac{|(Y_i - \hat{Y}_i)^2|}{N} \tag{6}$$

where N is the number of non-missing data points, x_i are actual observations time series and $\hat{Y}_i$ the predicted values.

Backpropagate trough time: After the error is calculated using the loss function the network backpropagates through time (BPPT). When using RNN neural networks, the weights of the matrices and the cell states of the LSTM cells are updated. These bacpropagation algorithms which are used are called optimization algorithms. The standard algorithm for this is the gradient descent.

Optimizer: The objective function of a neural network is minimized using an optimizer. As mentioned earlier the standard is gradient descent, but this has many short comings and thus other optimization algorithms have been introduced. Reimers and Gurevych wrote an article [28] on the performance of the following optimizations algorithms: Adagrad [29], Adadelta [30], RMSProp [31], Adam [32] and Nadam [33]. From these algorithms they concluded three of them worked relatively better with less variance, thus those will be tried on the LSTM model.

One of the limitations of LSTM is when the predictions in the model are mostly based on recent past observations [34]. In LSTM there is also a disadvantage in the the inability to ascribe contributions of input features to model outputs, common to all artificial neural networks [35]. Another disadvantage of LSTM might be the long training time. LSTM is a form of deep learning and will therfore have longer training time then models which are not. This can become a problem if the prediction and the model has to be run often.

3.3.3 Prophet

The last model which is used to fit the data is Facebook's Prophet model. This model is a simple modular regression model. The underlying equations will be explained here, but can also be found in the article [20]. The reason this model is good to use is, because it gives the user easy access to a number of implementations:

Capacities: Data scientist can have more knowledge or data from other sources which can be relevant for determining the market size. With the ability to specify capacities in Prophet, they can directly apply that knowledge and those sources.

Changepoints: Prophet automatically selects changepoints which are points in the data that indicate growth changes. Data scientists can directly adjust and specify those points as well.

Holidays and seasonality: Data scientists have external knowledge on holidays and events which impact growth in specific regions. Prophet allows them to change and specify this in the model.

Smoothing parameters: The model can be changed from more global to more locally smooth models. A data scientist is able to do this by τ. Furthermore, a data scientist can use the seasonality and the holiday smoothing parameters (α, ν) to specify the model in how much seasonal variation is needed in the future. The equation of the model is as follows 7:

$$y(t) = g(t) + h(t) + s(t) + et \tag{7}$$

With:

y(t): Additive Regressive model
g(t): Trend factor
h(t): Holiday component
s(t): Seasonal component
et: Error term

The model parameters g(t), s(t), h(t), et are piecewise linear curves for modeling non-periodic changes in time series, periodic changes, the effects of holidays with irregular schedules, and error term accounts for any unusual changes not accommodated by the model respectively [21]. The first parameter in the model is the trend factor. The trend factor in Prophet can be one of two models. Either a logistic growth model or a piecewise linear model. The logistic growth model is used in case the data is nonlinear. The most basic form of the logistic growth model is:

$$g(t) = \frac{C}{1 + e^{(-(k)(t-m))}} \tag{8}$$

In this model C is the carrying capacity (maximum sustainable range). The k parameter is the growth rate and m is an offset parameter. However k must be a changing variable. The

model needs to take into account the constant changing of growth to better fit historical data. Moreover, the capacity is not constant in most cases and in Prophet is therefore written as a changing time dependant variable C(t). Thus, to take into account changes in the growth model, Prophet configures changepoints. These are the points where k is allowed to change. If at times s_j there are S changepoints ($J = 1, ..., S.$). Then a vector of rate adjustments can be defined as $\delta \in \mathbb{R}^S$, at time s_j, where δ is the rate change. This can be structured as a vector $\mathbf{a}(t) \in \{0, 1\}^S$ where:

$$a_j(t) = \begin{cases} 1, & \text{if } t \geq s_j \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

Where $k + \mathbf{a}(t)^\top \boldsymbol{\delta}$ is defined as the rate at time t. In the case k is changed, m also needs to be corrected. This correction at changepoint j is defined as:

$$\gamma_j = \left(s_j - m - \sum_{l<j} \gamma_l \right) \left(1 - \frac{k + \sum_{l<j} \delta_l}{k + \sum_{l \leq j} \delta_l} \right) \tag{10}$$

Model (8) can then be written as:

$$g(t) = \frac{C(t)}{1 + \exp\left(-\left(k + \mathbf{a}(t)^\top \delta\right)\left(t - \left(m + \mathbf{a}(t)^\top \gamma\right)\right)\right)} \tag{11}$$

If the forecasting problem does not have signs of saturating growth a piece-wise constant rate of growth gives a parsimonious model. In that case the trend is modeled as follows (12):

$$g(t) = \left(k + \mathbf{a}(t)^\top \delta\right) t + \left(m + \mathbf{a}(t)^\top \gamma\right) \tag{12}$$

Prophet also has a number of limitations. When the model has large errors which are related to the baselines the model can be misspecified. Furthermore the model is relatively high responsive to outliers. This makes it difficult to work with a lot of data which has many outliers like corrupted Covid19 data. Lastly in the case of changepoint detection Prophet detects a lot of false positives which results in wrong predictions.

4 Framework design and development

4.1 Data Analyses & Exploration

This section will start with a further look into the data, analyses and exploration of the data sets. Next, the different models and implementation are given. Each model has different results, but to compare them all the root mean squared error (RMSE) of each of these models will be presented.

4.2 Data set 1

The most important data of the first data set regards the dates, the price and the type of room. To do some more analyses it would be better to divide the data set into the different types of rooms. This resulted in data sets for: budget, single, standard, premium and apartment. The first type to be explored in more detail is budget. The two most interesting insights we can get from this data are the prices of the room and the amount of rooms booked (and not canceled). When the average price per day is analysed a time series plot is expected following some form of seasonality. However, as seen in figure 3 this is not the case. There is a sign of a time series plot, but there is no clear form of seasonality. Thus, further analyses is needed to determine whether or not there is seasonality.

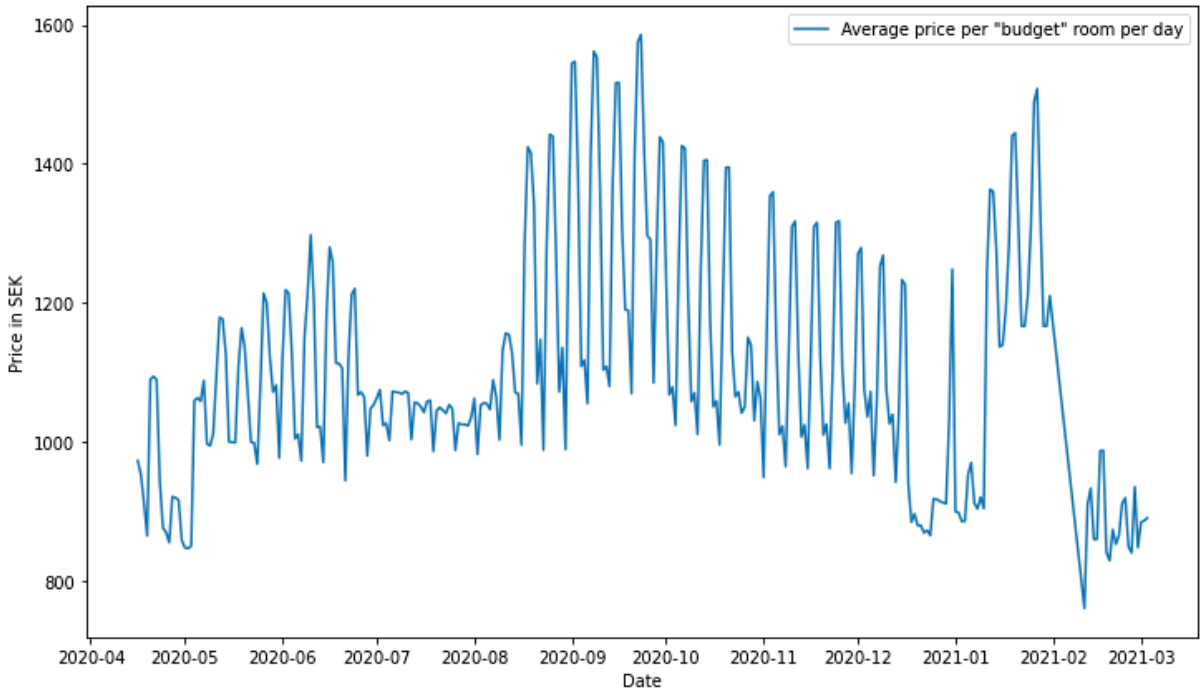


Figure 3: Average price per ”budget” room per day. Over the last year between April 2020 and March 2021

The time series is in this case the result of the prices fluctuating per weekday. This is to be expected, for example: rooms might have higher prices on fridays and saturdays

as this is a popular time for tourists and locals to book a room in comparison to other weekdays. There is also see a drop in prices in June/July of 2020. This can be explained due to Covid-19 and the restrictions in Sweden at the time. This is also the same reason there is a spike in prices afterwards. In all of Europe the restrictions in August/September were also less strict and allowed for more travel then other periods in 2020. The decline in prices in February 2021 is due to the second wave of Covid-19 hitting Sweden and the restrictions that yet again came with.

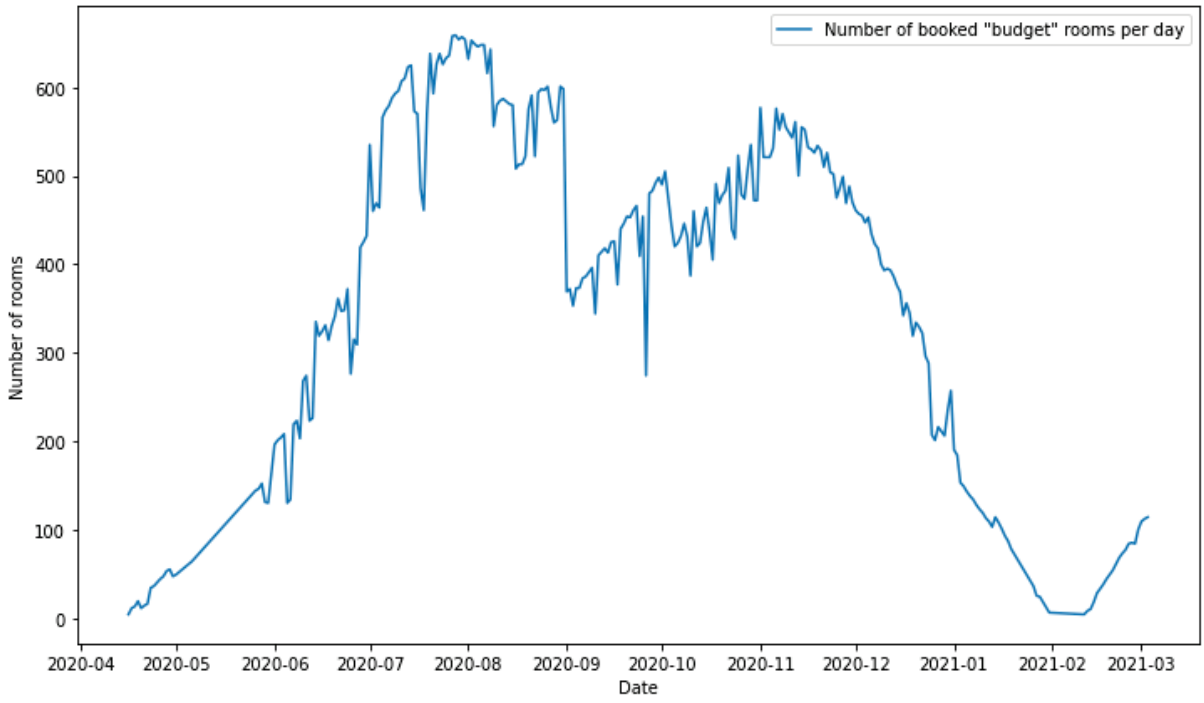


Figure 4: Total number of booked ”budget” rooms per day. Over the last year between April 2020 and March 2021

In figure 4 the number of booked ”budget” type rooms per day can be seen. The plot looks like it shows only a small correlation with figure 3. To test if there is a small correlation between the average price per day and the number of rooms booked per day and whether or not this result is significant the the following hypothesis is formulated:

Hypothesis H_0: There is no correlation between the average price and the number of rooms booked per day.

Hypothesis H_1: There is correlation between the average price and the number of rooms booked per day.

The average price of rooms per day is continuous and the number of rooms per day is discrete. The data is also parametric and therefore it is best to use the Spearman’s correlation test to test the above stated hypothesis. Spearman’s test gives a correlation coefficient of 0.2180 with a p-value of 0.0001. This is in the 95% confidence interval and therefore H_0 is rejected and the assumption is that there is a small to medium positive correlation between the average price and the number of rooms booked per day. This can be used

later in predicting pricing models for the data, or in predicting the future demand of hotel rooms. It is, however, interesting to note that despite the small positive correlation there is no clear sign of figure 4 following a time series trend like figure 3. This might suggest that the fluctuation in prices of hotel rooms over the week are not as important as first expected.

Next, was looking into the average price for all the different types of rooms, to see if they follow the same pattern and trend as the ”budget” type room. As seen in figure 5 all of the rooms follow the same trend as the ”budget” type rooms. The only anomaly being the ”apartment” type rooms, but there is very limited data regarding this type, because there are not a lot of these types of rooms available and in demand.

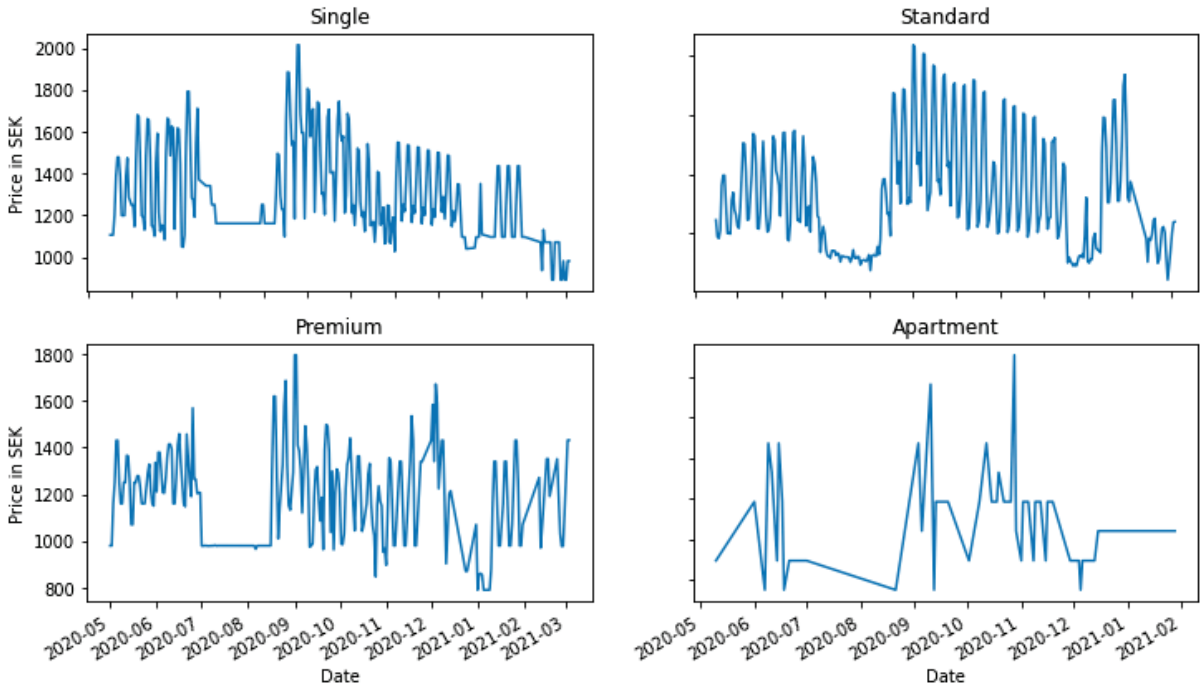


Figure 5: Average price per room type per day. For types: single, standard, premium, apartment. Over the last year between April 2020 and March 2021

4.3 Data set 2: Benchmarking data set

The benchmarking data set contains the demand and revenue of one hotel (and its direct competitors) from 2017 to 2021. There is no distinction between the type of rooms. To better understand the data we are working with, we plot both the demand of rooms and the revenue of the hotel and its competitors over the past few years and see if they all follow the same trend. Below in figure 6 we can see the demand of this year and the demand of competitors this year. For convenience the data is only shown until 01-01-2020, because the data after that point gets messy and unreliable due to Covid19.

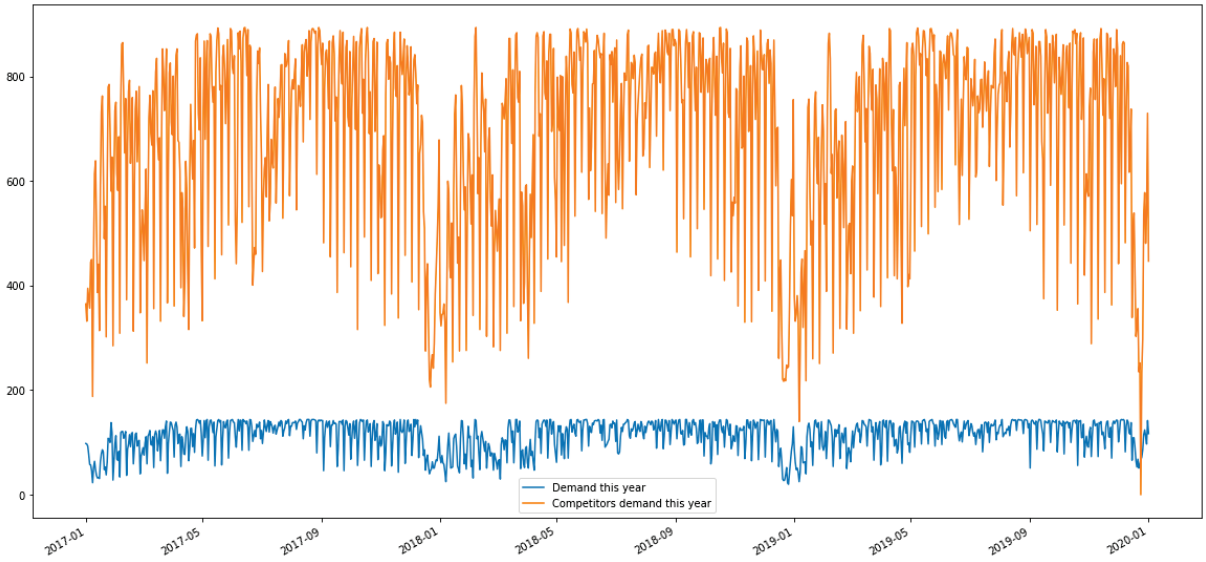


Figure 6: Demand over the past few years vs the competitors

In figure 6 we can see both trends look quite similar, but it is difficult to compare. This is because in the competitors case, it concerns the demand of five hotels compared to only one hotel. Thus we need to average the demand of the hotels. But the average supply of the competitors is higher, thus it might still not be accurate. Therefore, the best way to compare them might be to use the percentage of difference of supply and demand. Then comparing this to the percentage of average difference of supply and demand of the competitors. This results in figure 7. This graph is already a lot clearer and it is clear there is a similar trend between the two with sometimes the concerning hotel having a higher percentage of difference and sometimes its competitors having a higher percentage of difference. One thing we can see is that it looks quite like a stationary time series trend. The amount of data we have for this set, dates back further then data set 1 thus we can see and model a time series trend more clearly in this case.

Because only the demand and not the demand compared to its competitors needs to be predicted it does not matter if its competitors are taken into account at this point. Thus, the data about its competitors is omitted for now. First, the demand of the hotel until 01-01-2020 is plotted to see the trend more clear, as seen in figure 10. It is possible

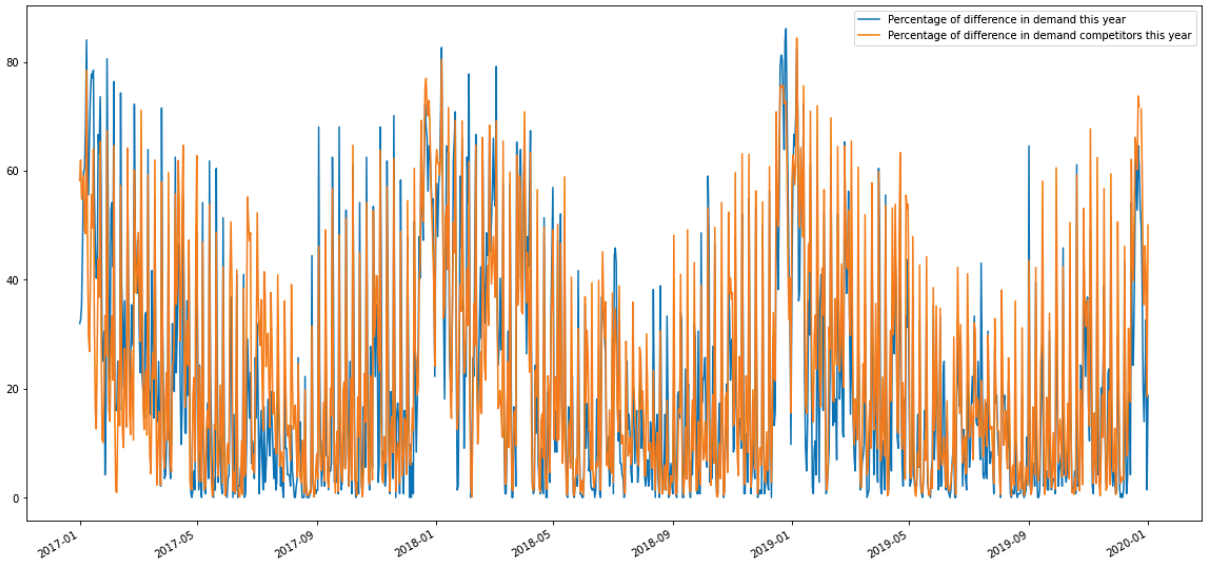


Figure 7: Difference of demand and supply in percentage vs the competitors

to observe a seasonality in this graph. But it is not clear if the data is stationary as seasonality does not always imply non-stationarity. In some cases more tests need to be done to determine (non) stationarity. The basic idea of stationarity is that the probability laws that govern the behavior of the process do not change over time. In a sense, the process is in statistical equilibrium [12]. As mentioned in 4.3 it is important to determine the stationarity when using the SARIMA model to be able to choose the right settings for the integrative part of the model.

To get a clearer view of the trends graph 8 shows a trend of only the year 2019. Here it is visible that over the year the trend stays somewhat the same, except for a few months during the summer. From early May until the end of August the valleys are much higher overall then the rest of the year. From this, we can conclude there is an off-season and an on-season for the hotels, between May and September each year. Figure 9 shows two graphs. On the left we can see the demand over 2 weeks in January in 2019 and on the right one we can see the demand over a couple months in 2019. It is clearly visible here that each month has 4 similar trends (weekly trends) that have one valley per week. In the weekly graph we determine that the weekly valleys are Sundays. Thus we can conclude from this that in general the demand is lower on Sundays and the trend follows weekly seasonality.

Next is to look into each of the columns of the data set and see if they are correlating with the demand data column. This could help in future forecasting. Table 4 and table 5 give all of the correlations of the columns using spearman's correlation test.

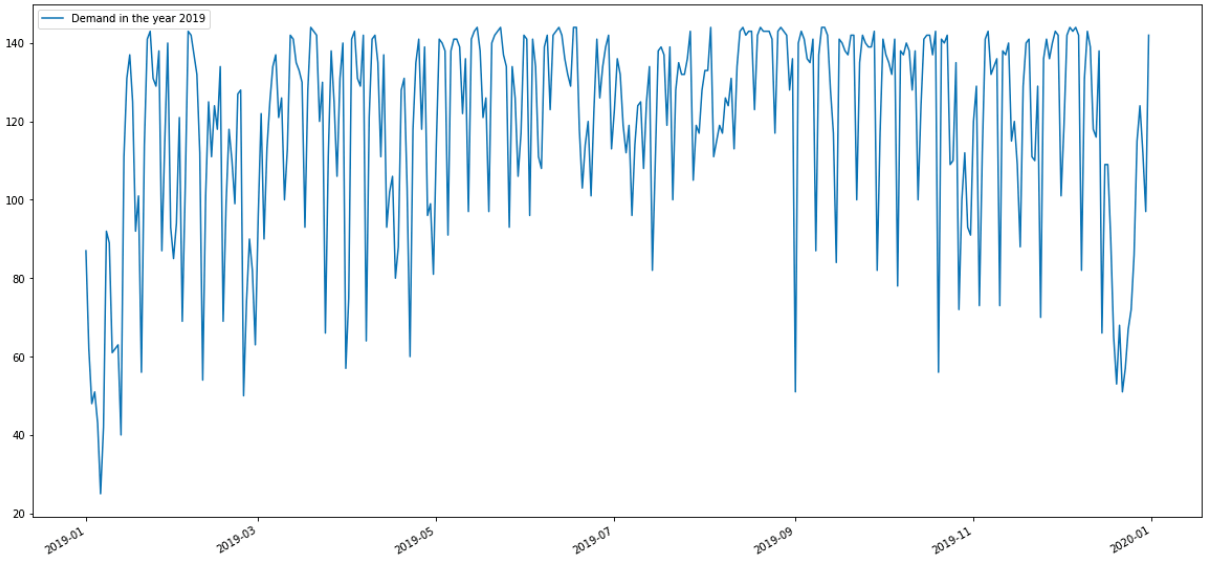


Figure 8: Demand over the year 2019

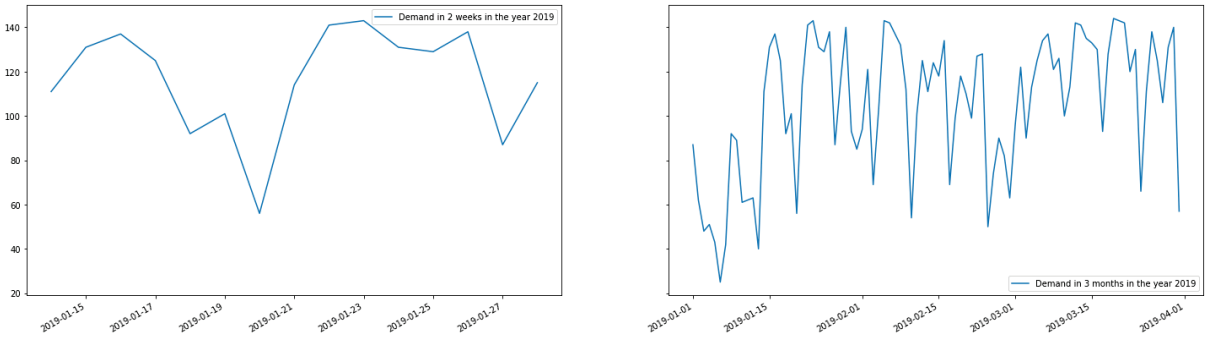


Figure 9: Demand over a few weeks in January 2019 (left) and a few months in early 2019 (right)

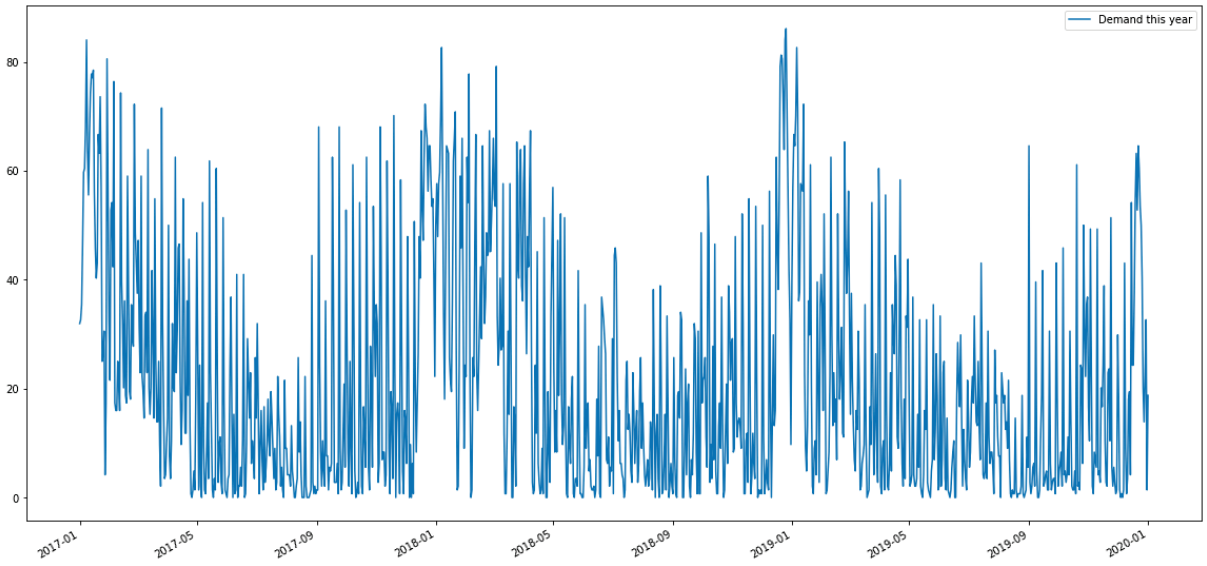


Figure 10: Demand over the last few years until 2020

Correlation	PropDemTY	PropRevTY	PropDemLY	PropRevLY	CompSupTY
PropDemTY	1.000000	0.870771	0.412756	0.404225	0.208648
PropRevTY	0.870771	1.000000	0.418537	0.546604	0.181302
PropDemLY	0.412756	0.418537	1.000000	0.822828	0.028711
PropRevLY	0.404225	0.546604	0.822828	1.000000	0.062224
CompSupTY	0.208648	0.181302	0.028711	0.062224	1.000000
CompDemTY	0.905050	0.876771	0.391931	0.394134	0.294007
CompRevTY	0.835934	0.963654	0.410889	0.520695	0.223014
CompSupLY	-0.125426	-0.175180	0.040978	0.003434	-0.084739
CompDemLY	0.480813	0.491116	0.875582	0.824020	0.040140
CompRevLY	0.431847	0.562953	0.768899	0.947674	0.052684

Table 4: Correlation of the all the data columns together part one

Correlation	CompDemTY	CompRevTY	CompSupLY	CompDemLY	CompRevLY
PropDemTY	0.905050	0.835934	-0.125426	0.480813	0.431847
PropRevTY	0.876771	0.963654	-0.175180	0.491116	0.562953
PropDemLY	0.391931	0.410889	0.040978	0.875582	0.768899
PropRevLY	0.394134	0.520695	0.003434	0.824020	0.947674
CompSupTY	0.294007	0.223014	-0.084739	0.040140	0.052684
CompDemTY	1.000000	0.928255	-0.231434	0.457258	0.418342
CompRevTY	0.928255	1.000000	-0.210546	0.484184	0.542055
CompSupLY	-0.231434	-0.210546	1.000000	0.044112	0.034284
CompDemLY	0.457258	0.484184	0.044112	1.000000	0.889165
CompRevLY	0.418342	0.542055	0.034284	0.889165	1.000000

Table 5: Correlation of the all the data columns together part two

4.4 Modeling Benchmark Data

An important econometric task is determining the most appropriate form of the trend in the data [37]. It was already clear (section 4.3) the data follows a time series trend with signs of seasonality, which could imply non-stationary (but this is not always the case). Thus, it is good to test that assumption before continuing. Therefore, we use the Augmented Dickery Fuller test (ADF), which is a unit-root test that will allow us to test if the time series is non-stationary:

Hypothesis H_0: The time series is stationary.

Hypothesis H_1: The time series is non-stationary.

After running the test it yields the following results (table 6).

ADF Statistic:	-3.289702
p-value:	0.015345
lags:	24.0000
Critical value 10%:	-2.568
Critical value 5%:	-2.864
Critical value 1%:	-3.435

Table 6: ADF test, to check for stationarity

The lower the p-value the more likely the time series is stationary. The test statistic is lower in the case of the critical values of 10% and 5%. The p-value is lower then 0.05. From this we can conclude that we accept the null hypothesis and thus the time series is stationary. If we want to be stricter we would have to look at the 1%. To ascertain the model fits stationarity with a 1% critical value we could use differencing in the SARIMA model. That would mean setting the Integrated part of the model to 1. This is however not necessary as the 10% and the 5% are already satisfactory enough to run the model with the assumption the time series is stationary.

4.5 SARIMA Model 1

Now that we have determined seasonality and stationarity the next step is trying to fit a model and start predicting. A good fit for seasonal time series data would be a SARIMA model. Because we only have 4 years of data the model is influenced heavily by any year that is following a different trend or behaves like an outlier. In the benchmark data set the data of 2020 is corrupted in the time Covid19 broke out. Table 7 shows the correlation between the last four years and clearly shows bad correlation for the year 2020. Trial and error analyses showed the data has outliers between the 50th and 230th day of the year 2020. To solve this issue it is best to leave out the data from the 50th until the 230th day of the year and do further forecasting without that part of the data.

Corr/P-val	2017	2018	2019	2020
2017	X	0.36/0.00	0.18/0.00	0.06/0.25
2018	0.36/0.00	X	0.38/0.00	0.03/0.57
2019	0.18/0.00	0.38/0.00	X	0.02/0.67
2020	0.06/0.25	0.03/0.57	0.02/0.67	X

Table 7: Spearmans Correlation test Correlation statistic and P-value in two decimals

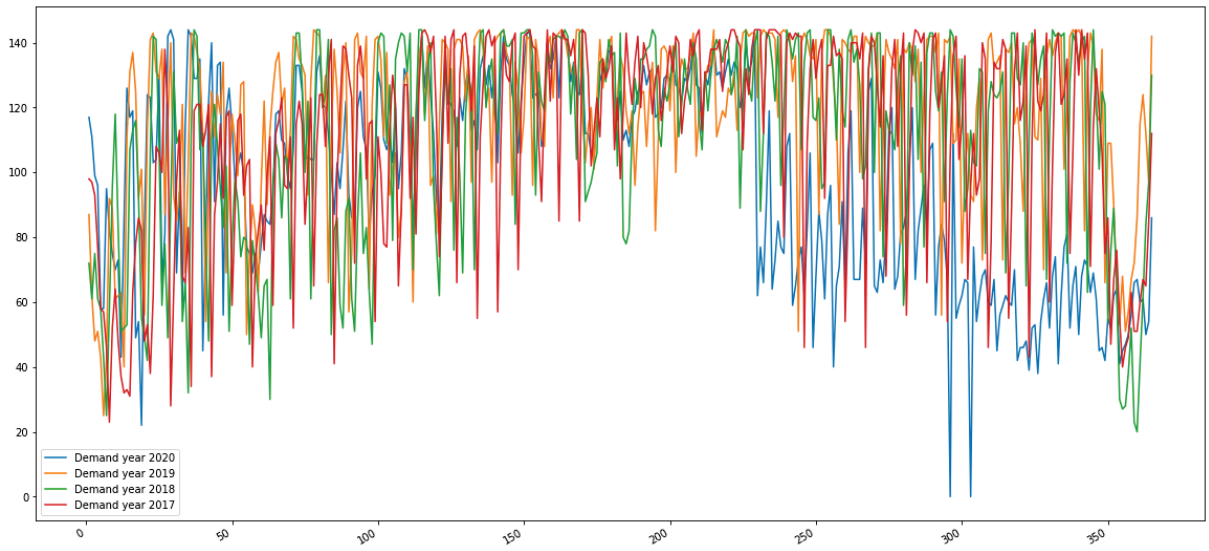


Figure 11: Demand per day over the last few years with adjusted data for 2020

If we would also look at the stationarity again with leaving out the data for 2020. The time series becomes stationary even for the 1%. See table 8.

To decide inputs for the AR and MA parameters of the model we will plot PACF and ACF graphs. In the graphs in figure 12 we can see in the ACF there is a geometric decay at each 7th lag and in the PACF we can see it is significant at each 7th after the first

ADF Statistic:	-4.426629
p-value:	0.000266
lags:	22.0000
Critical value 10%:	-2.568
Critical value 5%:	-2.864
Critical value 1%:	-3.435

Table 8: ADF test, to check for stationarity. Without the data of 2020

lag. Thus significant at (1,7,14,21,28). The lag at 7 tells us there is a weekly pattern (7 days) present in the data. As presented earlier there is stationarity thus making I = 0. For AR we use the PACF and can clearly see that within 1 lags the AR is significant thus we choose AR = 1.

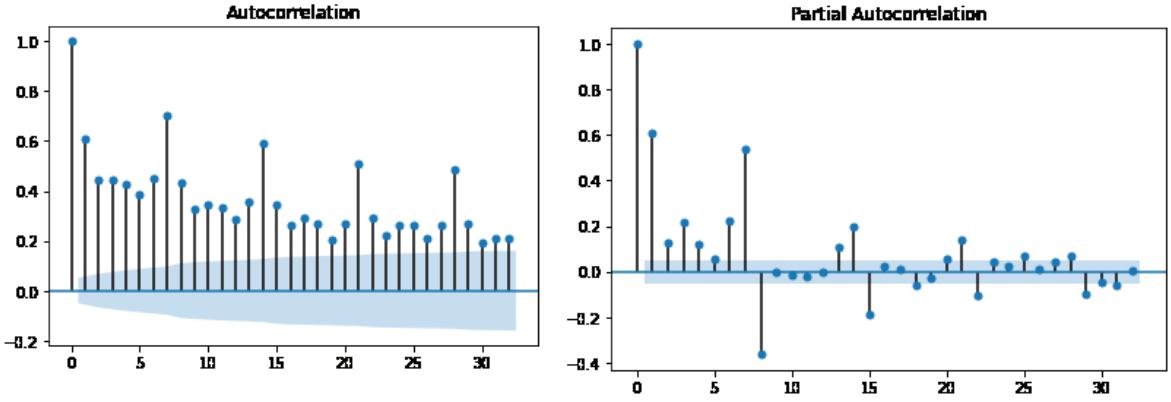


Figure 12: ACF and PACF.

Next is running the SARIMA(1,0,0)(1,0,1,7) model. Figure 13 gives 4 graphs. The first shows the predicted values and the actual values in one line plot. The second shows the plotted errors. The third shows the theoretical quantiles and the last shows the autocorrelation plot. The RMSE for this model is: 12.664.

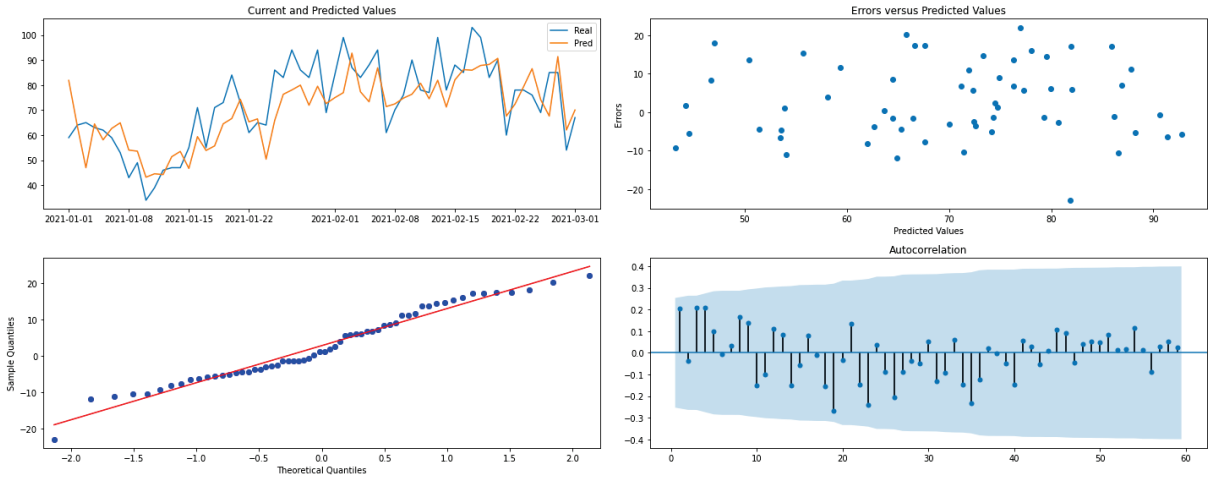


Figure 13: SARIMA results 1

4.6 LSTM Model

The second model which is fitted on the data set is an LSTM (long short-term memory) model. Before it is possible fit this model the data set must be transformed into a supervised learning problem. There are two options to explore. The first one is using the already existing data set as a supervised problem. This is possible as there is already a column with the exact data of a year prior. This would suggest we take lag 365 as we look 365 days previous to the current day. The second option is by only using the current data and adding a column with data a week prior to each day. A week, because if we look back at the ACF and PACF the determined lag is 7, furthermore in section 4.3 it also became clear there is a weekly seasonality. Thus, if we add this column it will help restructure the data as a supervised learning problem.

As we saw before, the data set is stationary if we look at the critical values of 5% and 10%. If we take into account that we are leaving out the data of 2020 the data set is stationary even for the 1% critical value.

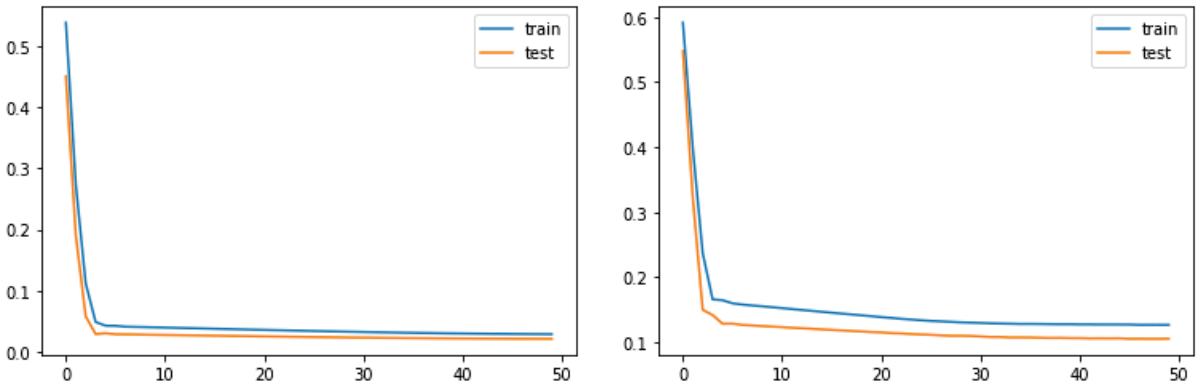
The next transformation on the data set would be scaling. We need to scale the time series in order for us to use LSTM. There are multiple options to scale our data (see chapter 3), but we will use Min-Max Scaling as we will end up with smaller standard deviations, which can suppress the effect of outliers [15]. LSTM works better if the corrupted data of 2020 is not taken into account when creating the model, therefore the model will train and predict with the data of 2017, 2018 and 2019.

Both of the models are tested with two different loss functions to see which one fits the model better. The first loss function is the mean squared error 5 and the second loss function is the mean absolute error 6. Both of these functions work better than other loss functions when the data concerns a regression model, which is clear in this case. The squared error is usually easier to solve for the model, but the absolute error function is more robust to outliers. Both of the loss functions can be found in section 3.3.2. The assumption would be that the MAE does not defer much from the MSE, because we only use data from 2017 until 2019. Thus, the data does not contain many outliers. Secondly, because the forecasting is about hotel room demand the forecast has a maximum (the maximum number of hotel rooms in a hotel) and a minimum (there cannot be less then 0 hotel room demand) thus this limits the outliers to 0 and 144. The first LSTM models which are run have only 1 hidden layer. This hidden layer has 50 neurons. Furthermore, it is not a stacked LSTM model. This means multiple LSTM are stacked in sequence to make the model better. Another influence on the performance of the LSTM model would be the optimizer algorithm. A few of these algorithms were already mentioned earlier in 3.3.2. Too see which of these optimizer algorithms perform better in this particular case they are all shown below in the first LSTM Model. The last step before running the algorithm is setting a preferred batch size. The batch size determines the number of iterations each epoch will run. The batch size is the number of inputs the neural network takes a time. For example, if we take a batch size of 32 each iteration will have 32 inputs thus the number of iterations will be 25 for each epoch. A batch size of 32 is the preferred

standard and therefore used in this model as well.

4.6.1 LSTM supervised model 1, choosing the loss function

The first model which is implemented is the one which compares the current values of each day for the past years (2017, 2018, 2019) with the same values one week prior. This means the supervised values will go back t-7 steps. The LSTM model will use 7 extra columns as the input for this particular set-up. The model is run with 50 epochs (figure 14a with MSE and figure 14a with MAE). With trial and error it was clear this was enough for the model to converge without over fitting. Lastly, the model uses most of the last year (2019) as validation set (288 days total) and the rest of the data as training set. Figure 15 and 16 show the predicted values and their actual values. The RMSE for this forecast was 18.151 with the MSE. The RMSE for this forecast was 18.127 with the MAE. This confirms the earlier made assumption that the loss function does not influence the outcome of the model too much. Thus, for the next LSTM models only the MAE is used as loss function, because this loss function performs slightly better.



(a) Convergence of epochs MSE (b) Convergence of epochs MAE

Figure 14: Epochs of MSE and MAE

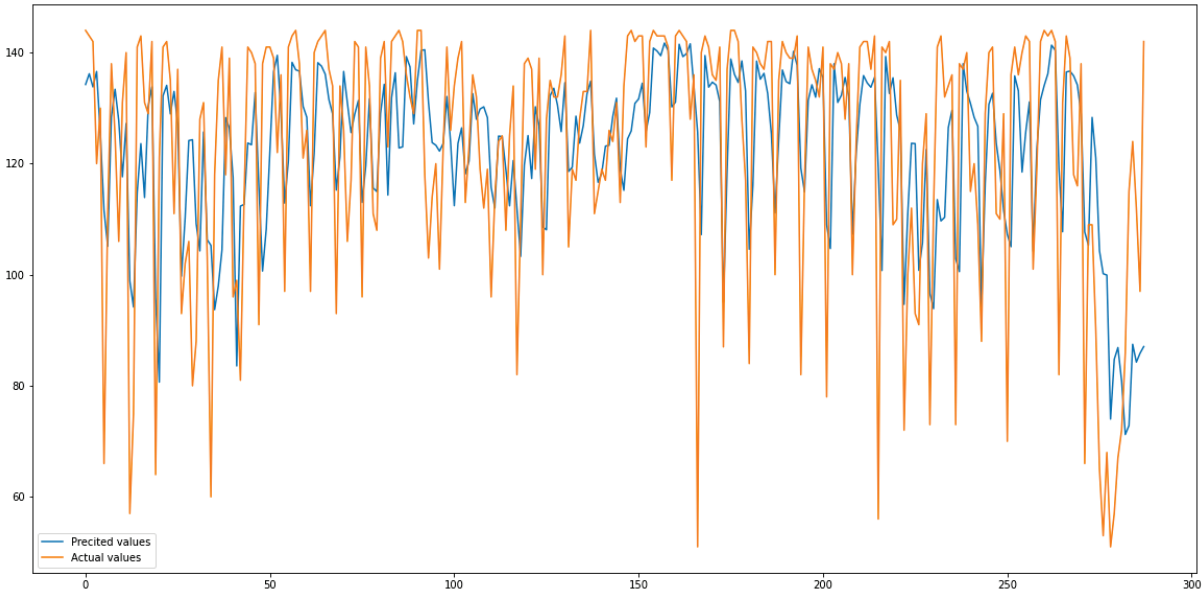


Figure 15: LSTM model 1 with MSE, predicted and actual values

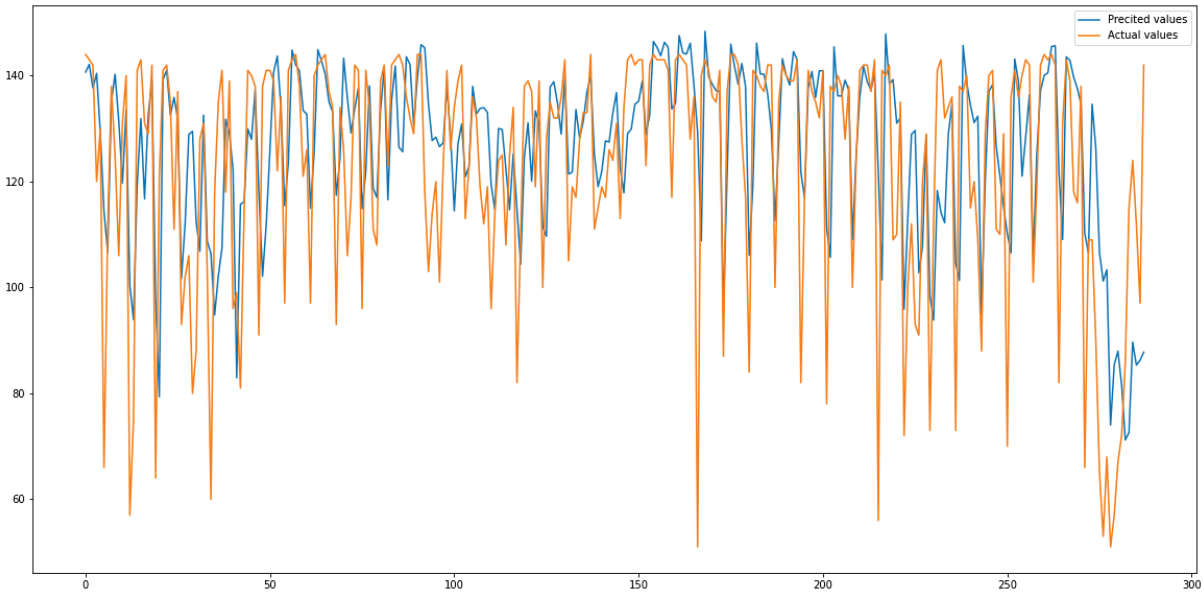


Figure 16: LSTM model 1 with MAE, predicted and actual values

4.6.2 LSTM supervised model 1, choosing the optimizer

The next model uses the MAE as loss function, determined in the previous section. All the other parameters stay the same. Three optimizer algorithms are tried: RMSProp, Adam and Nadam. Adam was the optimizer which was used in the precious section thus the RMSE for this optimizer is 18.127. In figures 17 and 18 are the resulting predictions from RMSProp and Nadam with RMSE of 18.333 and 18.087 respectively.

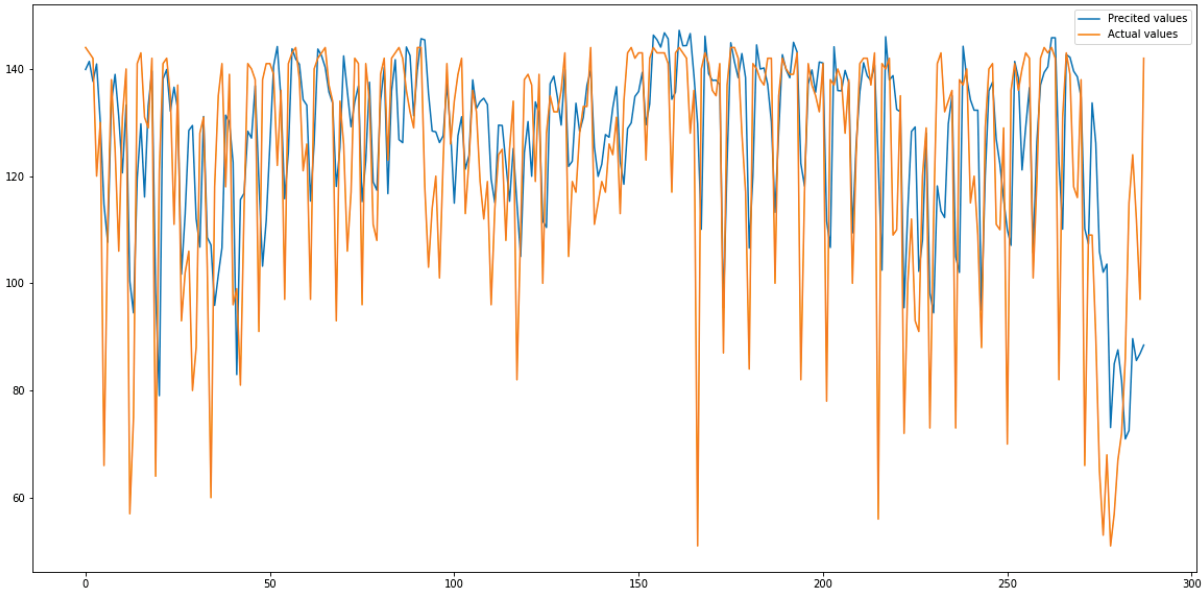


Figure 17: LSTM model 1 with RMSProp, predicted and actual values

From these results it is possible to assume the optimal combination of loss function and optimization algorithm for this data set is MAE with Nadam. To be sure this assumption is correct table 9 shows the RMSE of each of the combinations. It was wrong to assume that the MAE would be optimal in all cases, as seen in the table, it is clear that the optimal parameters are MSE and RMSProp. Figure 19 shows the predicted results of this combination.

RMSE	MSE	MAE
Adam	18.151	18.127
RMSProp	17.706	18.333
Nadam	17.938	18.087

Table 9: RMSE for each combination of loss function and optimization algorithm

4.6.3 LSTM model 2

The next step in optimizing the LSTM model is seeing wether or multiple hidden layers work better. The number of neurons within each hidden layer also play a part in this

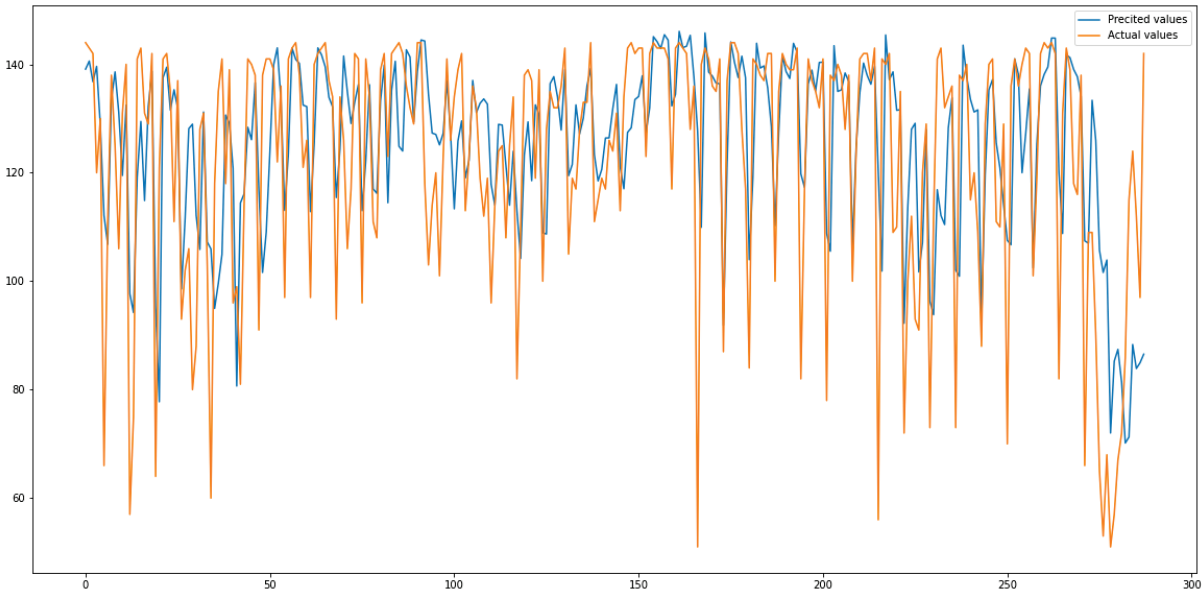


Figure 18: LSTM model 1 with Nadam, predicted and actual values

part of the optimization. There is no "one way" of determining the optimal amount of hidden layers and neurons in a neural net. There is no proof or equation which applies to every algorithm. So how can anyone determine their optimal parameters? The only way to do this is by trial and error. Furthermore, there are some rule of thumbs which you can follow. One of these is a generally accepted formula 13 to calculate the maximum number of neurons in a hidden layer. The number of hidden layers should not be to large to prevent overfitting. Therefore, it is decided that the maximum number of hidden layers tried on this algorithm will be 10.

$$Neurons = \frac{Numberoftrainingsamples}{(\alpha * (Numberofinputneurons + Numberofoutputneurons))} \tag{13}$$

Where α represents an arbitrary number between 5 and 10. TO find the optimal settings the above equation is used with every α between 5 and 10 and with hidden layers 1-10. The algorithm is run until all options have been tried and returns the best RMSE outcome of all options with the respective number of neurons and hidden layers. After running the algorithm the optimal setting is found with 3 hidden layers and 58 neurons per hidden layer. This yields an RMSE of 16.546. The predicted and actual values of this model are shown in figure 20.

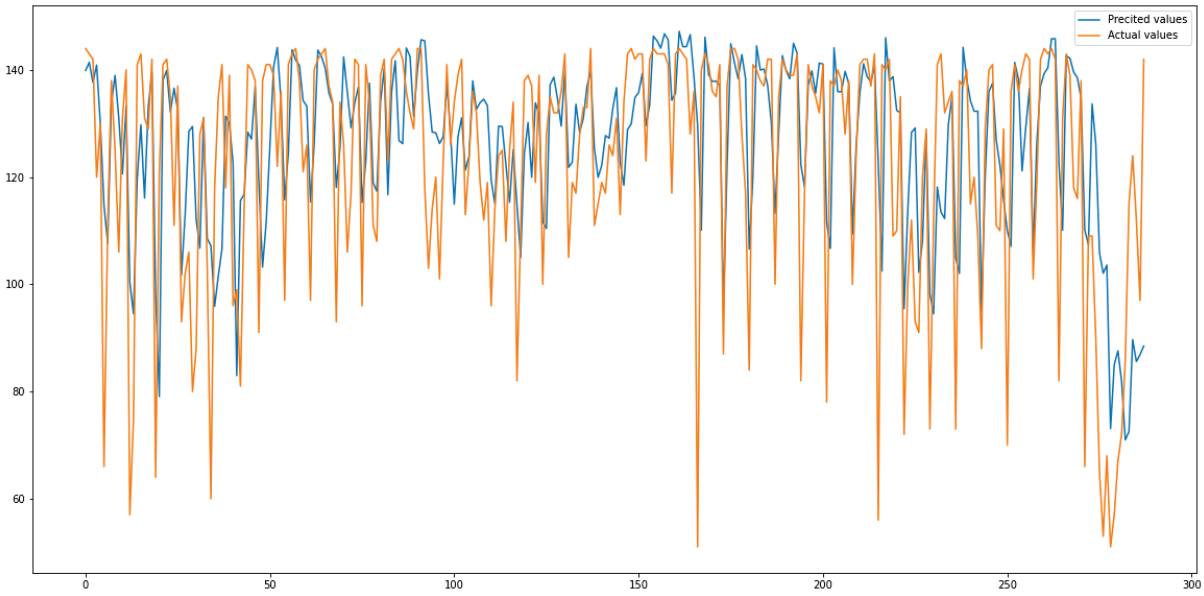


Figure 19: Predicted and actual values of the LSTM model with optimal parameters for the loss function and the optimization algorithm.

4.6.4 LSTM model 3

The final step in optimizing the LSTM model is looking into whether or not a stacked LSTM causes better performance. A stacked LSTM means the model has an extra LSTM layer as one of the hidden layers. Graves, et al. were the first to introduce stacked LSTMs in 2013. They mentioned the following: *"RNNs are inherently deep in time, since their hidden state is a function of all previous hidden states. The question that inspired this paper was whether RNNs could also benefit from depth in space; that is from stacking multiple recurrent hidden layers on top of each other, just as feedforward layers are stacked in conventional deep networks."* [40]. In this paper they discovered that the depth of the network was more important then the number of memory cells in a hidden layer. Thus, the next model is a double stacked LSTM model with still the same parameters previously stated, except for one hidden layer being changed to a LSTM layer. The actual and predicted values are seen in figure 21. The extra stacked layer resulted in a RMSE of: 17.492

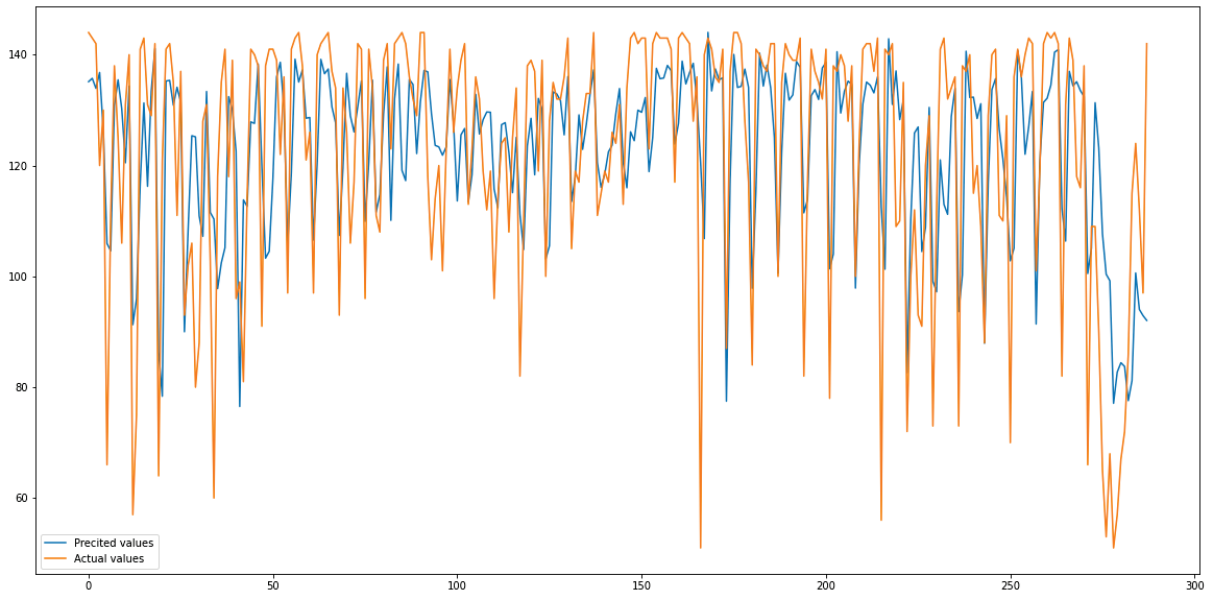


Figure 20: Actual and predicted values with 58 neurons and 3 hidden layers.

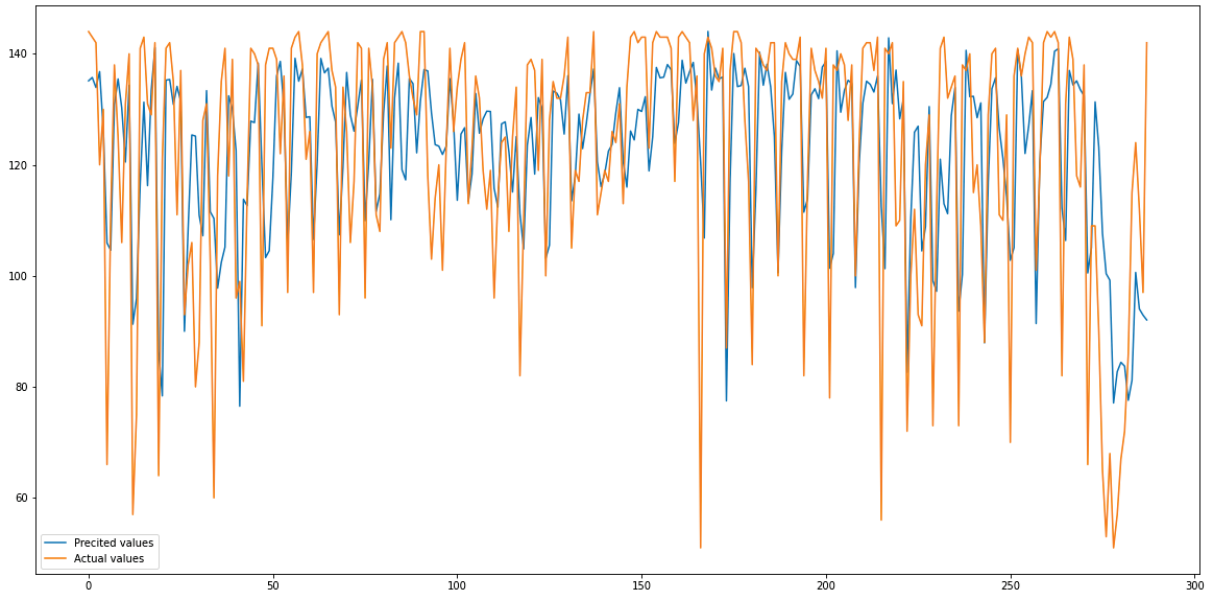


Figure 21: Actual and predicted values with 58 neurons and 3 hidden layers and one stacked LSTM.

4.7 Prophet

The next section will show the implementation of the prophet model. The first model used is a simple modular regression model using only the demand of the last few years as input. The settings are determined by the model itself. The first forecast is done over a period of 365 days in the future. Figure 22 shows the results of the first forecast. In this graph it is visible that the model takes into account the contaminated data of 2020. This results in a lower, yet gradually increasing, trend for 2021. What immediately stands out in the resulting prediction from Prophet is the increasing trend of 2021, even though the model does include the corrupted data of 2020. The interesting part about this is that the previous models did not take into account this data, because it would throw the prediction off.

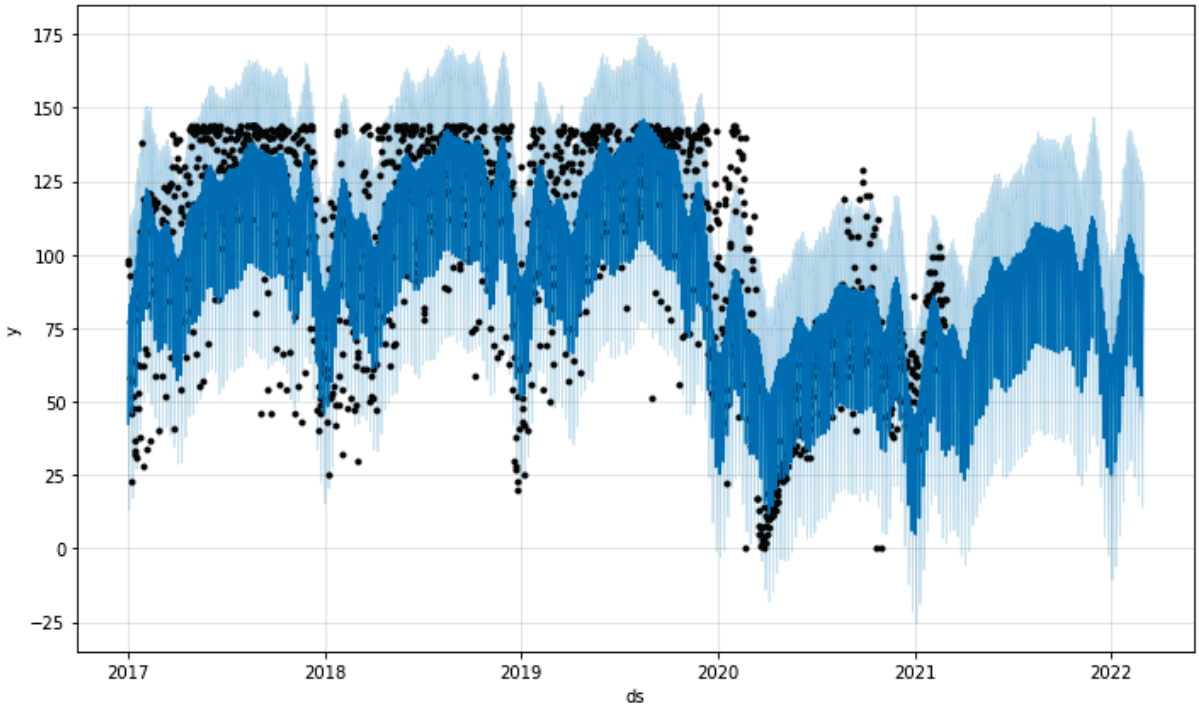


Figure 22: Results of 1 year forecast using Prophet

In figure 23 is the trend of the data and the forecast components. The trend is shown weekly and yearly. In the weekly trend the earlier assumption in section 4.3, about the weekly seasonality and Sundays having less demand is proven. This helps in further fine tuning of the Prophet model.

4.7.1 Prophet model 1

This forecast was made for a year in the future, to value the model the forecast needs to be until the last day of the observations. Next, a validation set needs to be selected. Time

series forecasting models work better in general if they are not predicting to far into the future, especially considering the simple modular regression of Prophet and the corrupted data of 2020 due to Covid19. Thus, the model will predict 30 days into the future until the last day of observations. This makes the validation set 30 days which will be used to calculate the RMSE to give an estimation of the performance of the model. This yields the following forecast (see figure 24) with a RMSE of 32.695.

4.7.2 Prophet model 2

In section 4.3 it was determined there was weekly seasonality with an on and off season each year between May and September. It is possible to include this in the model. To do this a column is added to the existing data which gives the number of the month. In case of the month being lower then 5 or higher then 8 the hotel demand is in off-season and in the other cases the demand is in on-season. This results in the following trends, see figure 26. And the resulting predictions in figure 25. This resulted in a RMSE of 32.631 which is slightly better then the previous RMSE.

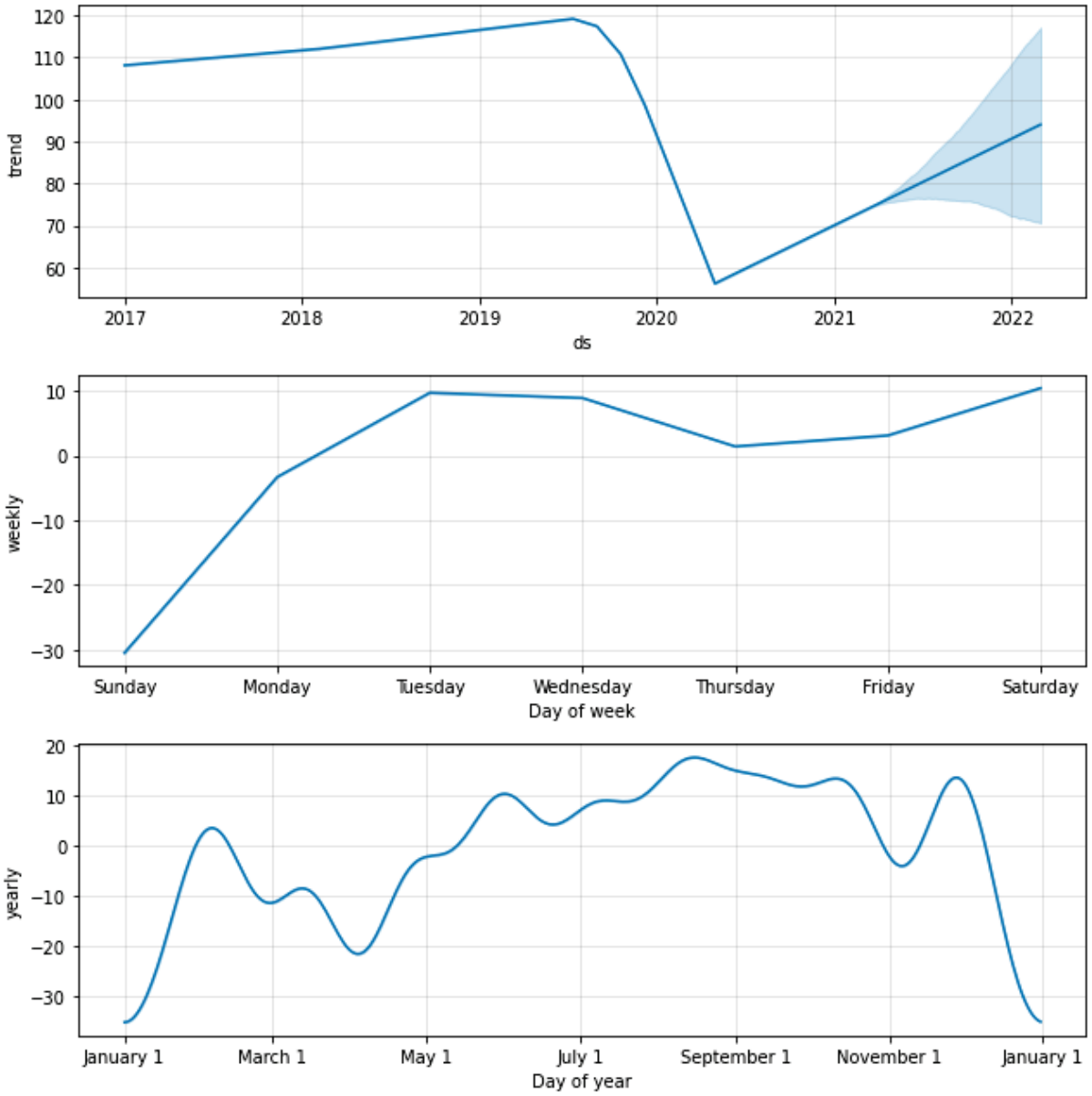


Figure 23: Trends of the forecast.

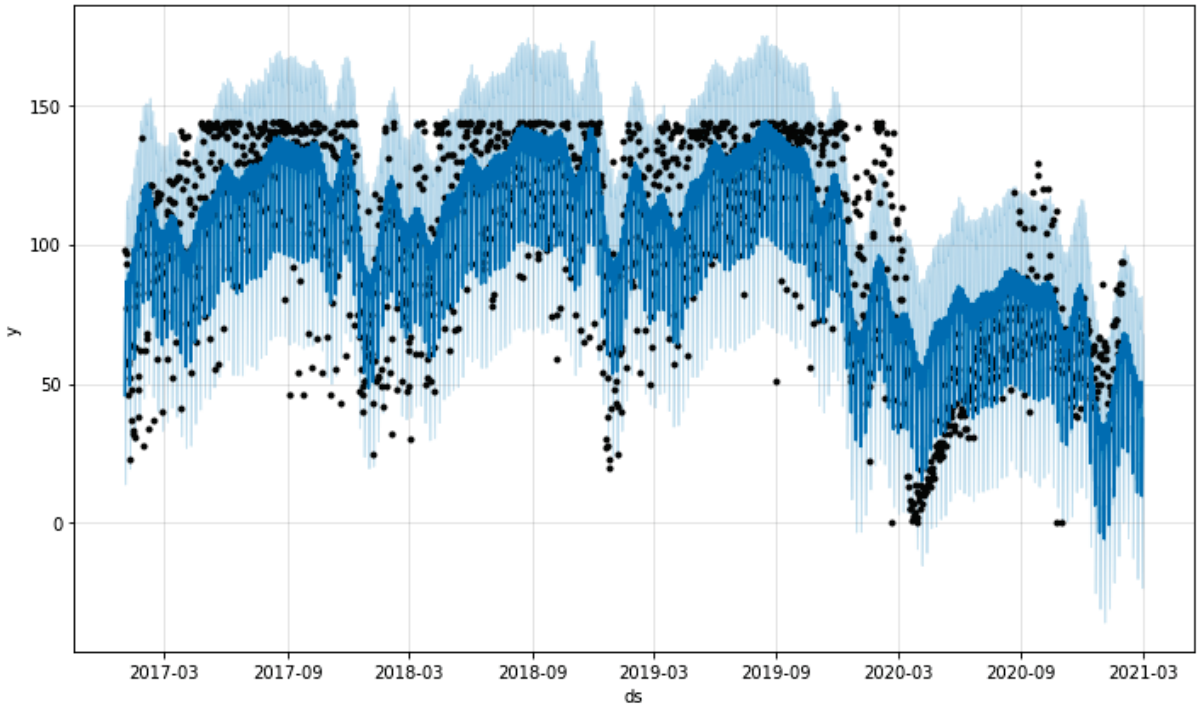


Figure 24: Prophet forecast for the last 30 days of the data set.

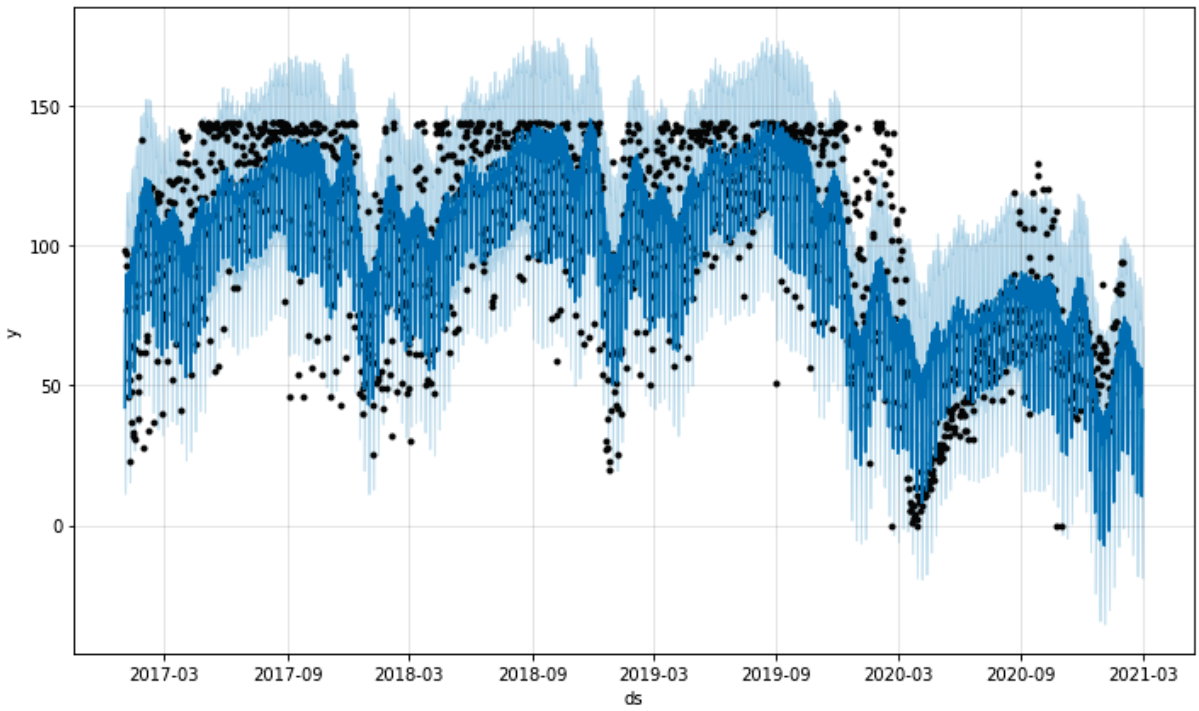


Figure 25: Forecast with on and off seasonal occurrence

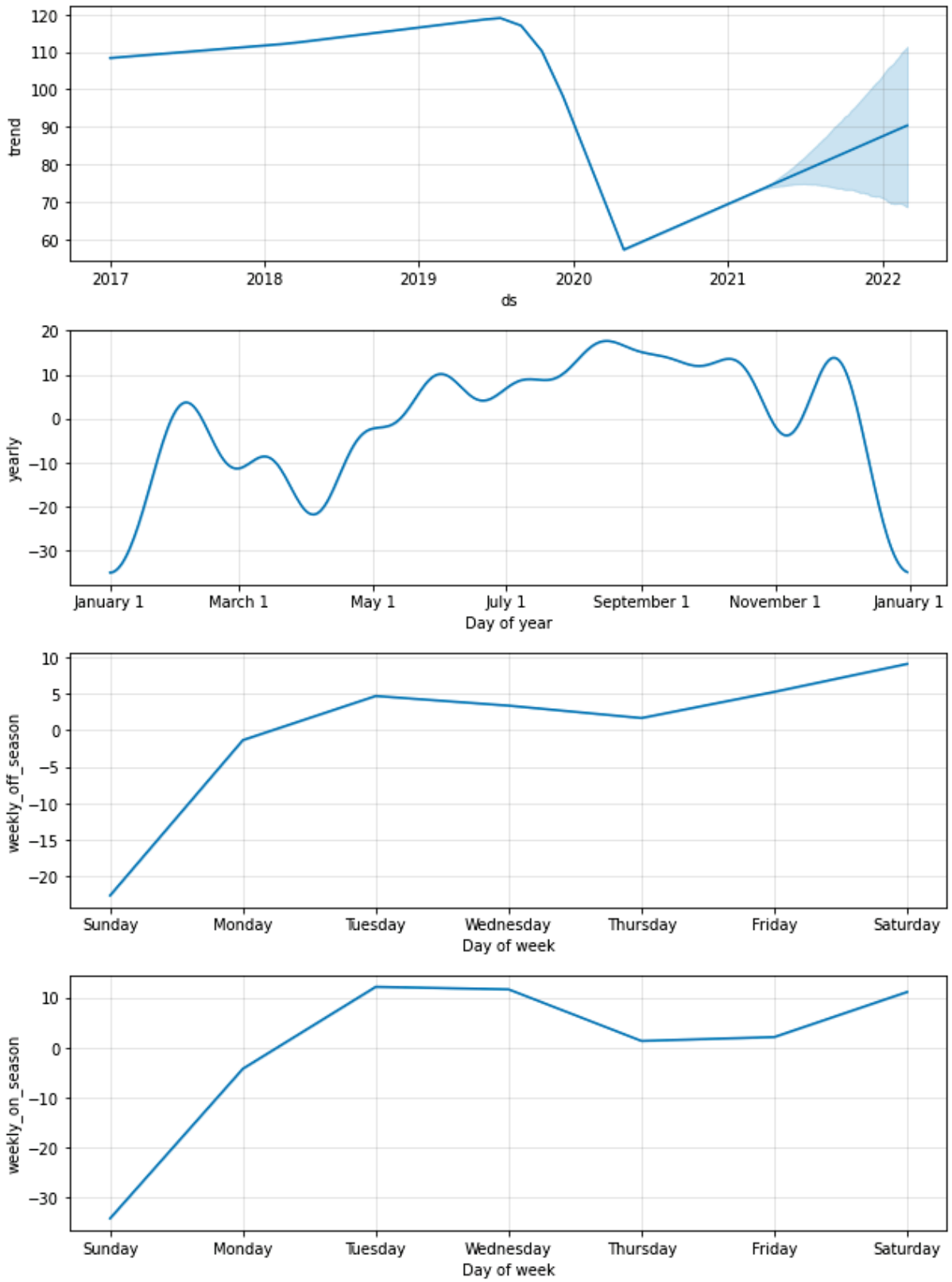


Figure 26: Trends of the forecast with on and off seasonal occurrence

4.7.3 Prophet model 3

In both forecasts it is visible that the predictions made reach below 0 at certain points. Of course this is not possible in the hotel demand industry as there can not be a negative need for hotel rooms. Thus, the next step is to implement a minimum in the model to ensure the model does not predict below this threshold. There are two ways of doing this in Prophet. The first is by applying a cap and floor to set saturated minimum in the logistic growth model. Although this is not recommended as it will corrupt your predictions. The second way is to first take the natural log of each of the input values before fitting the model, than fit the model and make the forecasts and lastly take the exponent of each of the predicted values to get the actual predicted values above 0. It is also important that when using this second step all the values need to be higher then 0 to avoid the log(0). This can be done by adding 1 to all the values before the prediction and subtracting 1 after the predictions. The results for applying both of the above mentioned methods are in the following figure 27. With an RMSE of 35.106. If we only follow the second option then the results are what can be seen in figure 28. With an RMSE of 33.935.

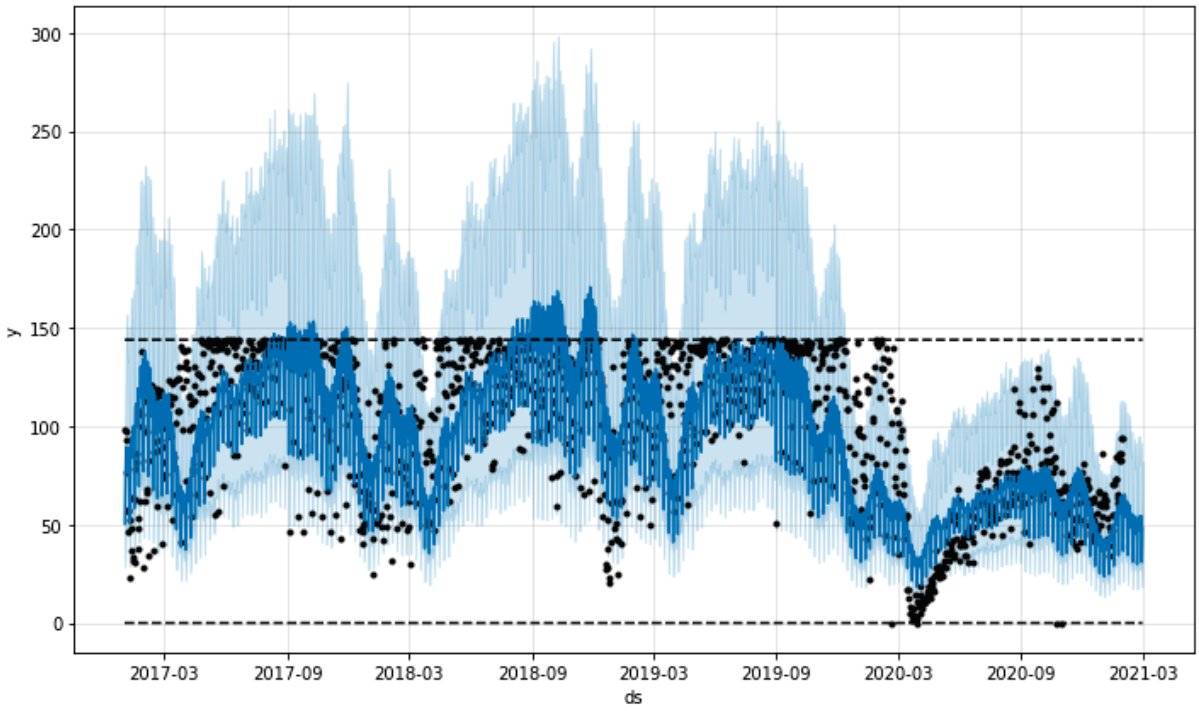


Figure 27: Predictions with a minimum of 0 with saturated growth minimum

4.7.4 Prophet model 4

The last step in implementing the prophet model is making sure it is comparable with the other models. Prophet models 1 to 3 are not comparable to the SARIMA and LST models, because in that case the data of 2020 is not used. Thus in order to make sure the comparison can be made the final Prophet model will only use the same data as the other two models. This results in the graph 29 and a RMSE of 37.672.

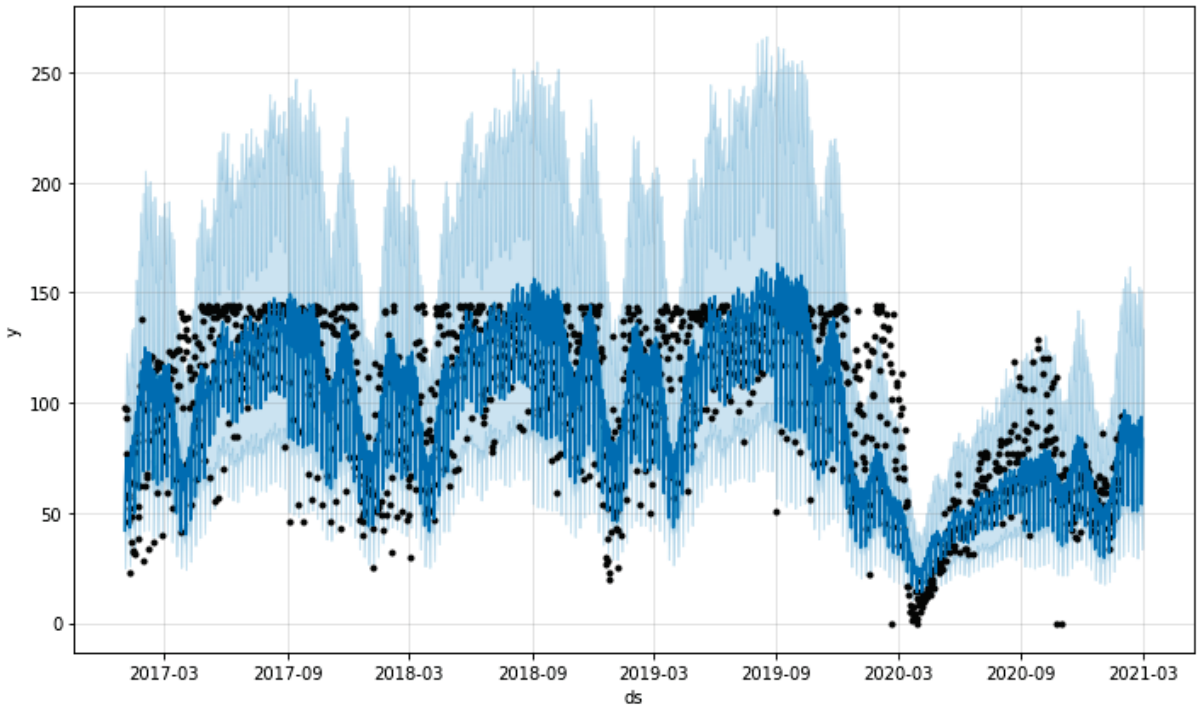


Figure 28: Predictions with a minimum of 0 without saturated growth minimum

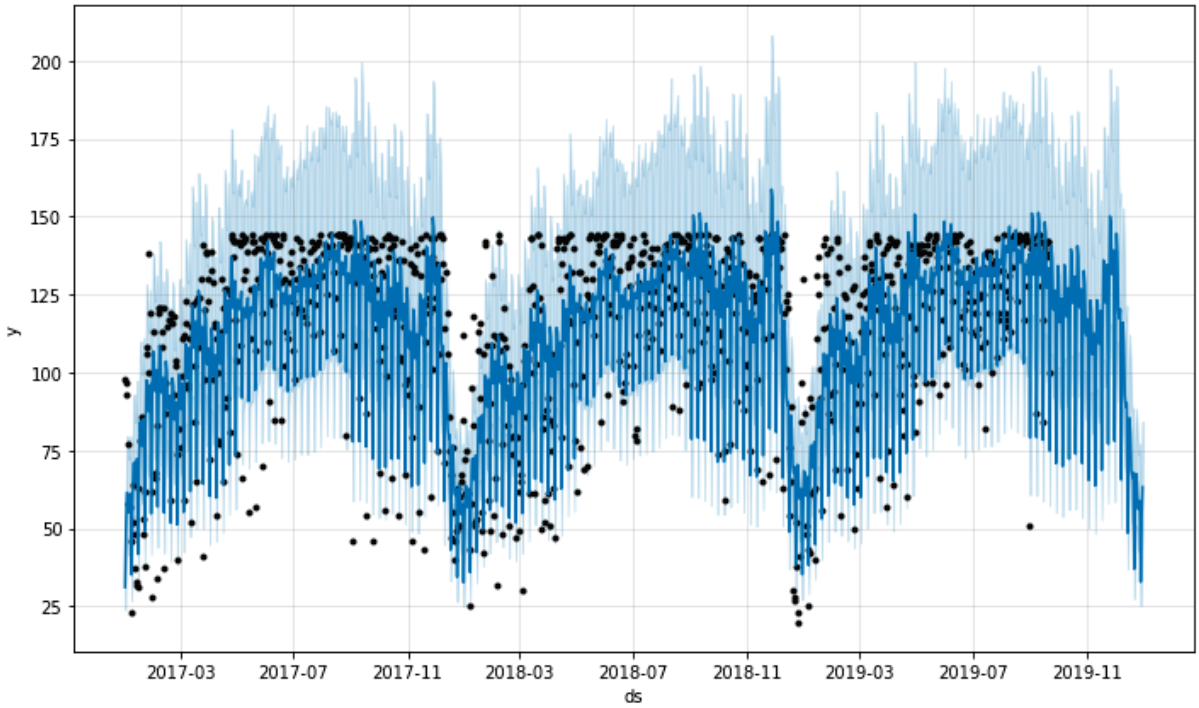


Figure 29: Predictions with a minimum of 0 without saturated growth minimum with data up until 2020

5 results

To compare the models, table 10 gives the RMSE for each of the models. The RMSE is a method often used for measuring the performance of a forecasting model [38]. The RMSE is calculated using the following equation 14:

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(x_i - \hat{x}_i)^2}{N}} \tag{14}$$

Where N is the number of non-missing data points, x_i are actual observations time series and $\hat{x}_i$ the estimated time series.

Table 10 gives an overview of the performed models and their respective RMSE. With LSTM model 1 being the first implemented LSTM model with optimized loss function and the best optimizer (MSE and RMSProp). LSTM model 2 is the same model with optimal number of hidden layers and neurons within this hidden layer (3 hidden layers with 58 neurons). LSTM model 3 being the same model with an extra stacked LSTM model as second hidden layer. Prophet model 1 is the first implemented Prophet model. Prophet model 2 is the last implemented Prophet model which takes into consideration a predefined weekly seasonality based on an on and off season. Prophet model 3 has a saturated growth minimum. Lastly, Prophet model 4 has all of the earlier aspects, but less data input to get a better comparable RMSE.

Model:	RMSE:
SARIMA	12.664
LSTM model 1	17.706
LSTM model 2	16.546
LSTM model 3	17.492
Prophet model 1	32.695
Prophet model 2	32.631
Prophet model 3	33.935
Prophet model 4	37.673

Table 10: Results of each of the algorithms

6 Conclusion and discussion

In this section will be the conclusion of this thesis, furthermore in section 6.1 will be suggestions and limitations for future research. The objective of this thesis was answering the previously stated (section 1) research question:

Which algorithm between SARIMA, LSTM and Prophet performs best in forecasting hotel room demand in Stockholm.

In summary **it can be concluded the LSTM model performs best compared to the other algorithms.** In section 5 it can be seen that SARIMA has the lowest RMSE and based on just this would be the best performing model. However, this is based on the data between 2017 and 2019 and the model can not take into account the data of 2020. Both the models LSTM and Prophet are able to accurately take this data into account without having heavy repercussions on the performance of the model. Thus, taking this into consideration a data scientist in a the hotel revenue management industry should choose one of the latter models. Given that in the results the RMSE of LSTM was close to the RMSE of SARIMA and therefore the performance on this particular data was similar, it is concluded that the LSTM model is the best performing model.

The main implications of this thesis

To conclude, this work has implemented three different algorithms on hotel data from Stockholm. First the workings of each of these algorithms have been explained (section 3) to understand how and why they can be used on the data and what their limitations might be. Next analysis is performed (section 4)on the data regarding the hotel market in Stockholm. Using this analysis optimal parameters can be chosen for the algorithms, including which data can be used and why. Then the three algorithms were implemented. Firstly SARIMA was used and returned a good RMSE. This goes to show SARIMA is already a good algorithm which can be used by the company for predicting hotel room demand. The second algorithm LSTM performed nearly as good as the SARIMA model in terms of RMSE. The model has been tuned as best as possible and the chosen parameters have been proven to be optimal for this data. The company can choose this algorithm as well, especially if other variables need to be included in the forecasting. Lastly, the Prophet model is shown to under perform compared to the other two models in terms of RMSE. This helps the company in making the decision of choosing the right model easier.

6.1 Limitations and future research

In section 3 all of the algorithms were explained including some of their limitations. One of the limitations of the SARIMA model was the use of corrupted and missing data of 2020 due to Covid19. This results in bad performance of the model in predicting future demand. The way to solve this might be to either leave this data out or to adjust the data manually to follow the same trend is the years before. A limitation of the LSTM model was that the training time was longer then the other algorithms and the model needs more data to outperform the other models more significantly. The prophet model performed poorly compared to the other models, this was mostly because the data had

many outliers especially considering the Covid19 data of 2020. Lastly the automatic changepoint detection of Prophet might be the cause for creating wrong changepoints which resulted in bad predictions. To solve this manual changepoints could be set. The data in the hotel industry will continue to grow. This makes it easier for algorithms which need more and longer time series data to become better. Further research is suggested to continue develop these algorithms and maybe try some different algorithms as well. It is also suggested that further research looks into combining external data and using this to improve the algorithms.

7 References

[1] *Models and techniques for hotel revenue management using a rolling horizon.*, Goldman, P., Freling, R., Pak, K. & Piersma, N., 2002.

[2] *Human factors in the design of revenue management systems in multinational corporations.*, Zarraga Oberty, C. & Bonache, J., 2007.

[3] *Hospitality and COVID-19: How long until 'no vacancy' for US hotels?"*, Vik Krishnan, Ryan Mann, Nathan Seitzman, and Nina Wittka, 2020.

[4] *Forecasting uncertain hotel room demand*, Rajopadhye, Mihir and Ghalia, Mounir Ben and Wang, Paul P and Baker, Timothy and Eister, Craig V, 2001.

[5] *An application of yield management to the hotel industry considering multiple day stays*, G.R. Bitran, S.V. Mondschein, 1995.

[6] *Managing hotel reservations with uncertain arrivals*, G.R. Bitran, S.M. Gilbert, 1996.

[7] *Dynamic operating rules for motel reservations*, S.P. Ladany, 1976.

[8] *Modeling and forecasting hotel room demand based on advance booking information*, Lee, Misuk, 2018.

[9] *A new approach to modelling and forecasting monthly guest nights in hotels*, Brännäs, Kurt and Hellström, Jörgen and Nordström, Jonas, 2002.

[10] *Stockholm HotellRapporten*, Annordia, 2020.

[11] *The Visual Display of Quantitative Information*, E.R. Tufte, 2001.

[12] *Time Series Analysis*, Jonathan D. Cryer Kung-Sik Chan, 2008.

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

[14] *Time series forecasting using a hybrid ARIMA and neural network model* , G.Peter Zhang, 2003.

[15] *About feature scaling and standardization and the effect of standardization for machine learning algorithms* , Sebastian Raschka, 2014.

[16] *Time series forecasting of petroleum production using deep LSTM recurrent networks* , Alaa Sagheer and Mostafa Kotb, 2019.

[17] *Learning Sequence Representations*, Bayer and Justin Simon, 2015.

[18] *Long short-term memory.*, Hochreiter, S, and J Schmidhuber, 1997.

[19] *Employing long short-term memory and Facebook prophet model in air temperature forecasting*, Toharudin, Toni and Pontoh, Resa Septiani and Caraka, Rezzy Eko and Zahroh, Solichatus and Lee, Youngjo and Chen, Rung Ching, 2020.

[20] *Forecasting at scale*, Taylor, Sean J and Letham, Benjamin, 2018.

[21] *Prediction of covid-19 cases in India using prophet*, Indhuja, M and Sindhuja, PP, 2020.

[22] *Time series analysis of monthly rainfall data for the Gadaref rainfall station, Sudan, by SARIMA methods*, Etuk, Ette Harrison and Mohamed, Tariq Mahgoub, 2014.

[23] *Comparative study of four time series methods in forecasting typhoid fever incidence in China*, Zhang, Xingyu and Liu, Yuanyuan and Yang, Min and Zhang, Tao and Young, Alistair A and Li, Xiaosong, 2013

[24] *Introduction to artificial neural network*, Dongare, AD and Kharde, RR and Kachare, Amit D,2012.

[25] *The vanishing gradient problem during learning recurrent neural nets and problem solutions*, Hochreiter, Sepp, 1998.

[26] *A deep learning integrated lee–carter model.*, Andrea Nigri, Susanna Levantesi, Mario Marino, Salvatore Scognamiglio, and Francesca Perla, 2019.

[27] *Gradient-based learning algorithms for recurrent*, Williams, Ronald J and Zipser, David, 1995.

[28] *Optimal hyperparameters for deep lstm-networks for sequence labeling tasks*, Reimers, Nils and Gurevych, Iryna, 2017.

[29] *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization.*, John Duchi, Elad Hazan, and Yoram Singer, 2011.

[30] *ADADELTA: an adaptive learning rate method*, Matthew D. Zeiler, 2012.

[31] *Neural Networks for Machine Learning - Lecture 6a - Overview of mini-batch gradient descent*, Geoffrey Hinton, 2012.

[32] *Adam: A Method for Stochastic Optimization*, Diederik P. Kingma and Jimmy Ba, 2014.

[33] *Incorporating Nesterov Momentum into Adam*, Timothy Dozat, 2015.

[34] *Time Series Forecasting Based on Augmented Long Short-Term Memory[J]*, Hsu D., 2017.

[35] *Economic Nowcasting with Long Short-Term Memory Artificial Neural Networks (LSTM)*, Hopp, Daniel, 2021.

[36] *The utility driven dynamic error propagation network*, Robinson, AJ and Fallside, Frank, 1987.

[37] *Augmented dickey fuller test*, Mushtaq, Rizwan, 2011.

[38] *. Error measures for generalizing about forecasting methods: Empirical comparisons*, Armstrong, J. S., and F. Collopy, 1992.

[39] *Employing long short-term memory and Facebook prophet model in air temperature forecasting*, Toharudin, Toni and Pontoh, Resa Septiani and Caraka, Rezzy Eko and Zahroh, Solichatus and Lee, Youngjo and Chen, Rung Ching, 2020.

[40] *Speech Recognition with Deep Recurrent Neural Networks*, Alex Graves, Abdel-rahman Mohamed, Geoffrey Hinton , 2013.