|  |
| --- |
| IU |
| Unit 1:Newsvendor Model |
| DLMAIEECMDF02  Mojtaba Nabipour |

# Newsvendor model

**Study Goals**

Upon completing this unit, you will be able to

* define the optimal inventory stocking decision.
* explain the classical newsvendor model.
* understand various demand models.
* identify important distributions.
* understand important extensions of the classical newsvendor model.

## Introduction

The newsvendor problem is a well-known inventory management dilemma that entails determining the most advantageous quantity to order for a time-sensitive item with uncertain demand. The problem is named by analogy with a newsvendor who must decide how many newspapers to order for selling the next day. Given demand uncertainty, the newsvendor model finds the optimal stocking level to minimize the total cost. The newsvendor problem is common in real life; examples include perishable goods such as blood products that spoil quickly or fashion consumables or goods that become obsolete after a certain time. The term can also be applied to products that rely on technology, such as cell phones. Considering the uncertainty of the demand level, this problem is not trivial. In the following, we begin with a simple form of this model and use demand modeling methods to address demand uncertainty. From there, we explore the complex model of multiperiod newsvendors.

## 1.1 Single-Period Newsvendor Model

The single-period newsvendor model operates under the constraint that the inventory of goods can only be accessed for a limited time, and retaining stock between consecutive periods is impossible. These systems are inherently linked to one-time decision models, which pertain to decision-making processes involving a singular choice that is not expected to be replicated in subsequent instances. The goal of a one-time decision model is to make the best possible decision given the available information and resources available at the time. Here, the objective is to determine the order quantity that maximizes a product's profit when the demand is probabilistic (Chopra et al., 2007). If the order level is below the demand level, some profit is lost (understocking situation). Moreover, overstocking costs reduce profits if the anticipated demand is not satisfied. Thus, the problem is choosing the order level to maximize the profit or minimize the cost. To formulate the problem, we use the following notation:

|  |  |  |
| --- | --- | --- |
| Single-Period Model Notations | | |
| Notation | **Explanation** | **Simply speaking** |
| s | Unit price | The price at which a unit is sold |
| w | Wholesale cost per unit | Purchasing + transporting+… |
| Cu | Understocking cost per unit | If one unit is short the cost (losing the profit) will be Cu |
| Co | Overstocking cost per unit | If one unit is unsold the cost will be Co. usually it contains the original cost added by the inventory cost |
| f ( ) | Probability density function of demand | Likelihood of a certain demand |
| F ( ) | Cumulative density function of demand | Probability of realization of less than a certain demand level |
| Q | Order level | Number of stocked units with Q\* as optimum |
| x | Demand level |  |

Source: [Mojtaba Nabipour], (2023).

Given the order level Q, the total cost is calculated as follows:

Cu(x-Q) x>Q (Eq.1)

C(Q) =

Co(Q-x) x<Q

Since x is a random variable, the expected cost is

(Eq.2)

Leibniz’s rule proves that e(Q) is concave. Therefore, we use the optimality condition to find the minimum (Reid & Sanders, 2023):

(Eq.3)

In the literature, this fraction is called the “critical ratio” and represents the relative importance of overstocking cost versus understocking cost. To obtain Q\*, we take the inverse of the cumulative function:

(Eq.4)

 . (Eq.5)

The critical ratio indicates the level of uncertainty in the stocking decision. For example, stocking highly perishable goods is associated with considerable uncertainty, forcing Q to be reduced and vice versa, see Figure.

Probability Distribution Of Demand For Two Perishable Goods



Newspaper

Fashion goods

Order level

Order level

Source: [Mojtaba Nabipour], (2023).

**Example 1**: Use the tables below for the following tasks:

1. Calculate the expected profit for an order quantity of 3.
2. Calculate the expected profit for an order quantity of 4.
3. Given the normal probability distribution of a demand with a mean of 7 and a standard deviation of 1, calculate the optimal order quantity Q\*.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Demand quantity** | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| **probability** | 0.04 | 0.06 | 0.13 | 0.17 | 0.19 | 0.16 | 0.13 | 0.07 | 0.05 |

|  |  |  |  |
| --- | --- | --- | --- |
| **w** | 120$ | **Cu** | 150-120=30$ |
| **s** | 150$ | **Co** | 120-100=20$ |

**Answer**:

1. The profit can be calculated by subtracting the cost from the raw revenue, but in the case of probabilistic demand, we calculate the expected profit:

Since *F(3) = 1* we have

*E[Profit] = 3×150 -3×120 = 90*

1. If *demand = 3,* then *revenue = 3×150* with the probability of *0.04*.

If *demand ≥ 4,* then *revenue = 4×150* with the probability of *0.96 [i.e., F(4)=0.96]*. The expected profit is

*E[Profit] = 3×150×0.04 + 4×150×0.96 – 4×120 = 114*

1. F(Q\*) = *30 / (30+20) = 0.6*

NORM.INV *(0.6, 7, 1) = 7.25*

It is possible to use the Excel function NORM.INV (x, mean, std).

### Self-check Questions

1. What is the objective of the classical newsvendor problem?

*The classical newsvendor problem aims to determine the order quantity that maximizes profit or minimizes cost by balancing the trade-off between overstocking and understocking inventory levels.*

2. How does the classical newsvendor problem differ from the standard inventory management problem?

*The classical newsvendor problem differs from the standard inventory management problem in that demand for the product is uncertain and cannot be forecasted accurately. Additionally, the cost of overstocking and understocking may not be equal.*

## 1.2 Demand as a Stochastic Quantity

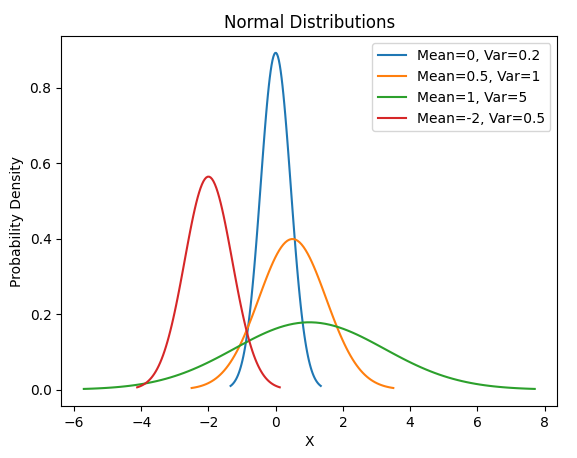
So far, we have introduced the demand quantity as an uncertain variable and implicitly used statistics to handle it in our cost function. This section assumes demand is a stochastic quantity that follows a statistical distribution. First, we briefly review probability distributions and the relevant statistical concepts, and then we use these concepts to describe the demand variable.

Deterministic demand in inventory management refers to a situation where the demand for a product or item is known with certainty. In other words, the quantity of items needed or sold is predetermined and does not vary or fluctuate. Inventory management would be straightforward if the received demand were deterministic. However, in reality, demand is a random variable with an uncertain nature. Practically, the probability distribution helps us to handle the associated uncertainty. Here, we first concentrate on continuous distributions and then discuss discrete distributions. The normal distribution is well suited to describe events that (i) are more likely to occur around a specific mean value and (ii) have an almost constant average dispersion. Such a distribution *f*N(*x*) is symmetric around the mean and can be defined by two parameters: mean µ and standard deviation σ:

. (Eq.6)

The figure below shows four different normal distributions. A normal distribution is bell-shaped. Notice how the parameters affect the shape of the distribution.

Normal Distribution



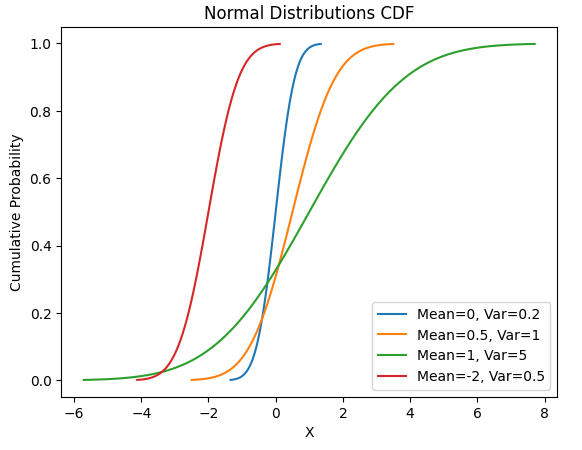
Source: [Mojtaba Nabipour], (2023).

The Central Limit Theorem is based on the normal distribution model. This theorem says that averages derived from independent, identically distributed random variables have approximately normal distributions, no matter how they are sampled (provided the distribution has a finite variance) (Ross, 2017). The cumulative distribution function (CDF) estimates the probability that a random distribution will produce a value less than or equal to a threshold value. Thus, the CDF is the integral of the probability of the random variable occurring between zero and the threshold value Z. This is expressed as

(Eq.7)

The CDF of the normal distribution is shown in the figure below.

CDF of Normal Distribution



Source: [Mojtaba Nabipour], (2023).

**Example 2:**

A weekly product demand follows a normal distribution with a mean of 1500 and a standard deviation of 300. There will be no deliveries in the next two weeks; the current on-hand inventory is 3200. Given the independent nature of the demand in different weeks:

1. In each of the next two weeks, what is the probability that demand will be less than *1600*?

2. What is the probability that the total demand in the next two weeks exceeds *3200*?

**Answer:**

1. Define x1 as the demand of week 1 (where the current week is week 0) and x2 as the demand of week 2.

According to the assumptions, the answer is

*where a is the demand probability in section a* and *p(a)* is its probability.

Since x1 and x2 are independent, we have

**P(a) = 0.3976**

2. Any linear combination of normal variables has a normal distribution with the same linear combination of means and variances.

If we define *x = x1+x2*, then

That is, we have

So, the requested probability is

where *b* is the demand probability in section b.

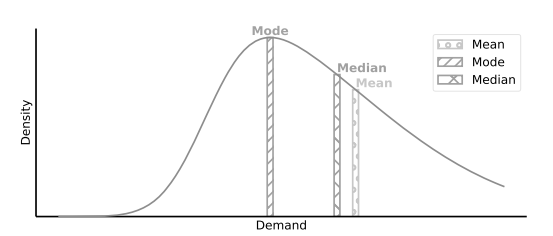
**P(b) = 0.068.**

Current on-hand inventory is often referred to as “safety stock.” It helps the vendor cover demand during supply shortages.

To this point, we have assumed that demand follows a normal distribution. As the name suggests, in normal situations, the demand for certain items often fluctuates around an average with a high probability, so it makes sense to use the normal distribution to describe the stochasticity of such demand.

Sometimes small variations such as exceptional weather, product improvement, or understocking by competitors generate unexpected demand for items that are usually in low demand. For example, the demand for toilet paper increased dramatically during the Covid pandemic. Thus, the normal distribution with a symmetrical spread around a mean value may not be optimal. Normal distributions have the same mode (the value that occurs the most often), median (the value in the middle of a dataset), and mean and are symmetrically distributed around them. However, real supply chains often have higher exceptions (values other than the mode) than the normal distribution predicts. As the figure below illustrates, the real distribution is right-skewed, and the demand fluctuates around the mode value rather than the mean (Vandeput, 2020). Furthermore, in normal distributions, the abscissa indicates demand, and negative demand does not conform with reality. Negative demand is only applicable when the mean is sufficiently high to make the negative demand negligible. As a rule of thumb, the mean must be five standard deviations from the origin (Snyder, 1984). Studies such as that of Burgin and Wild (Burgin & Wild, 1967) considered these difficulties and showed that, in such cases, the demand characteristics are better described by the gamma probability distribution.

**Demand Distribution in Real Supply Chain**



Source: [Mojtaba Nabipour], (2023).

The gamma distribution can be defined by two parameters α and β. The first parameter is called “shape” and the second is called “scale.”

These parameters can be derived as follows from the mean and standard deviation of the data (Vandeput, 2020):

(Eq.8)

(Eq.9)

The general form of the density function of the gamma distribution is

(Eq.10)

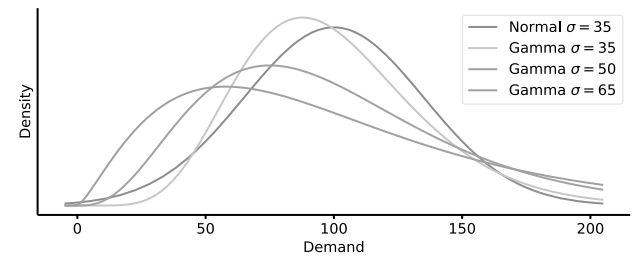
The figure below shows that the gamma distribution approaches the normal distribution upon decreasing the standard deviation, so the skew shrinks. Another characteristic of the gamma distribution is that it is strictly positive. The same figure shows that, near the origin, the probability density function is quite low.

The skewness of the distribution is directly related to the standard deviation. The skewness of the distribution is given by

(Eq.11)

The skewness is a reliable and straightforward tool to assess whether the gamma distribution or the normal distribution is better suited to a particular set of demand data.

**Gamma Distribution With Different Standard Deviation**



Source: [Mojtaba Nabipour], (2023).

The skewness of an observed set of demand is calculated by using the Excel function SKEW (range of data):

. (Eq.12)

Thus, if the skewness is close to zero, the distribution is normal, and if it is close to 2σ/μ, the distribution is a gamma distribution. The following rule of thumb can be used to choose the correct distribution (Vandeput, 2020):

**Example 3.1**

The following table contains a 94-day demand history. Which distribution best fits the data?

Table Title

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2.15259 | 3.78 | 1.84023 | 2.32288 | 2.70754 | 3.27141 | 3.80136 | 4.38438 | 5.40975 | 5.9123 |
| 0.82956 | 1.25938 | 1.99574 | 2.4638 | 2.81085 | 3.26863 | 3.8726 | 4.41528 | 5.72096 | 5.532 |
| 0.89594 | 3.8 | 2.0645 | 2.45825 | 2.89762 | 3.33974 | 3.95324 | 4.69969 | 5.82157 | 5.6732 |
| 1.92093 | 1.35397 | 2.01937 | 2.46988 | 2.86425 | 3.35425 | 4.0478 | 4.77041 | 6.17035 | 1.2346 |
| 0.98322 | 4.01 | 2.12213 | 2.5635 | 2.93938 | 3.43649 | 4.23609 | 4.79774 | 6.26586 |  |
| 1.08231 | 1.36602 | 2.13441 | 2.58187 | 2.95057 | 3.5293 | 4.2524 | 4.80418 | 6.35507 |  |
| 1.07008 | 1.45362 | 2.19559 | 2.5467 | 3.10426 | 3.54447 | 4.30449 | 4.99675 | 6.75722 |  |
| 1.15259 | 1.63576 | 2.20841 | 2.6403 | 3.10062 | 3.54565 | 4.31569 | 5.03056 | 7.0478 |  |
| 1.22441 | 3.48 | 2.28921 | 2.6036 | 3.13624 | 3.60547 | 4.32002 | 5.11493 | 8.05613 |  |
| 3.45 | 1.75526 | 2.30481 | 2.66168 | 3.25755 | 3.82151 | 4.37879 | 5.35323 | 8.09159 |  |

**(Table Source)**

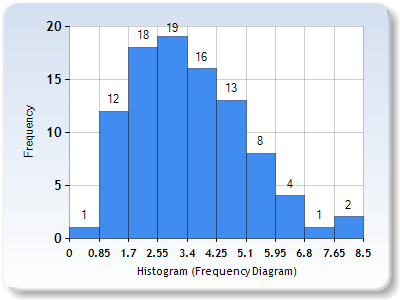
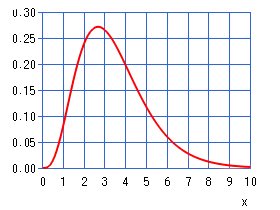
**Solution 3.1:**

Sd = 0.6468 , µd = 3.4417 , σd = 1.6277

)

The figure below shows the histogram of the table (blue bins) and the fitted gamma distribution (red line).

**The Fitted Gamma Distribution on the Demand Data**



Source: [Mojtaba Nabipour], (2023).

In the example above, we use the histogram method to identify the distribution. This method is simple but has limitations, such as limited accuracy in capturing fine-grained variations or irregularities in the data. The most prominent limitation is that histograms only provide a visual representation of the data but do not provide additional information about the underlying distribution. Students interested in detailed explanations can turn to Vandeput (2020).

**Discrete probability distributions**

A type of probability distribution that models the probability of specific outcomes in a discrete set of values.

**Discrete probability distributions** are used to model situations involving a finite number of possible outcomes, each with a specific probability of occurring.

In demand modeling, discrete probability distributions are commonly used to model the uncertainty in customer demand for a product. The probability distribution is used to estimate the likelihood of different levels of demand, which can be used to make decisions about inventory levels, pricing, and other aspects of supply chain management.

Several types of discrete probability distributions are commonly used in demand modeling, each with its own characteristics and applications. Some of the most used distributions are introduced in the following paragraphs.

#### Poisson Distribution

The Poisson distribution is commonly used to model demand for products or services that occur randomly over time, such as customer arrivals or machine breakdowns. The Poisson distribution is characterized by a single parameter lambda that represents the average number of occurrences per unit time. The Poisson distribution often sees use in inventory management and service operations, where demand is unpredictable and sporadic.

#### Binomial Distribution

The binomial distribution is commonly used to model demand for products or services with a fixed number of possible outcomes, such as the success or failure of a marketing campaign. The binomial distribution is characterized by two parameters, *n* and *p*, which represent the number of trials and the probability of success, respectively. The binomial distribution is often used in pricing and promotion decisions, where the goal is to estimate the probability of different levels of demand under different marketing scenarios.

#### Geometric Distribution

The geometric distribution is commonly used to model demand for products or services that have a fixed probability of success in each trial, such as the probability of a customer purchasing a product on each visit. The geometric distribution is characterized by a single parameter *p*, which represents the probability of success. The geometric distribution is often used to model customer behavior, where the goal is to estimate the probability of different levels of demand based on the probability of a customer purchasing a product on each visit or interaction.

#### Negative Binomial Distribution

The negative binomial distribution is similar to the geometric distribution but allows for a variable number of trials before success occurs. The negative binomial distribution is characterized by two parameters, *r* and *p*, which represent the number of failures before the first success and the probability of success, respectively. The negative binomial distribution is often used in inventory management and service operations, where demand can be unpredictable and sporadic.

#### Hypergeometric Distribution

The hypergeometric distribution is commonly used to model demand for products or services that have a fixed number of possible outcomes and that draw from a limited population, such as the demand for a specific product in a particular market segment. The hypergeometric distribution is characterized by three parameters, *N*, *n*, and *k*, which represent the total population, the sample size, and the number of successes in the population, respectively. The hypergeometric distribution is often used in market research and product development, where the goal is to estimate the demand for a product in a specific market segment.

We now introduce the Poisson distribution.

The Poisson distribution is a probability distribution that models the number of rare events occurring in a fixed interval of time or space. It is named after the French mathematician Siméon Denis Poisson, who introduced it in 1837.

The Poisson distribution is particularly useful in situations where events occur randomly and independently and where the average rate of occurrence is known or can be estimated. The distribution is characterized by a single parameter λ (lambda) that represents the average number of occurrences in the interval of interest.

The probability mass function of the Poisson distribution is given by (Walpole et al., 2011)

where *X* is the random variable representing the number of occurrences, *k* is a nonnegative integer, *e* is the base of the natural logarithm (approximately 2.718), and *k*! is the factorial of *k*.

Suppose a customer service center receives an average of 12 calls per hour. We can use the Poisson distribution to calculate the probability of receiving a certain number of calls in a given hour. For example, the probability of receiving exactly 10 calls in an hour is

This means that there is a 9.4% probability of receiving 10 calls in one hour, assuming that the number of calls follows a Poisson distribution with a mean of 12 calls per hour.

In a retail store, demand can be modeled with the Poisson distribution by treating each purchase as an event that happens randomly in time. The Poisson parameter λ is the average number of purchases per unit time, such as per hour or per day.

For example, suppose a retailer needs to predict the number of purchases in a one-hour period at a particular store. The retailer can collect historical data on the number of purchases made during one-hour periods at that location on previous days. The table below contains typical data for this situation.

Table Title

|  |  |  |
| --- | --- | --- |
| Day of Week | Hour of Day | Number of Purchases |
| Monday | 12:00 PM | 45 |
| Tuesday | 1:00 PM | 50 |
| Wednesday | 2:00 PM | 35 |
| Thursday | 3:00 PM | 60 |
| Friday | 4:00 PM | 75 |
| Saturday | 5:00 PM | 90 |
| Sunday | 6:00 PM | 80 |

Source: [Mojtaba Nabipour], (2023).

Using these data, the retailer can estimate the Poisson parameter λ for purchases during a one-hour period at the given store. The Poisson parameter λ can be estimated by calculating the average number of purchases per one-hour period over all days of the week. In this case, the average number of purchases per one-hour period is

lambda = (45 + 50 + 35 + 60 + 75 + 90 + 80) / 7 = 63.6

The retailer then uses the Poisson distribution to estimate the probability of different numbers of purchases during a one-hour period at the given store. For example, the retailer might want to estimate the probability of having more than 70 purchases during a one-hour period, which would require the store to be prepared to handle a higher volume of customers than usual.

Using the Poisson distribution with lambda = 63.6, the probability of having more than 70 purchases during a one-hour period is

P(X>70)=1-P(X<=70)= 1 - P(X = 0) - P(X = 1) - ... - P(X = 70) ≈ 0.004

This calculation shows that the probability of having more than 70 purchases during a one-hour period is quite low, approximately 0.4%. This information can help the retailer decide about staffing levels, inventory management, and other aspects of store operations.

### Self-check questions

1. Why is the Gamma distribution more suitable for demand description than the normal distribution?

*The gamma distribution allows for a wider range of shapes and has a longer tail than the normal distribution, which makes it more suitable for describing demand patterns that may be skewed.*

## 1.3 Demand Models

As discussed in the previous section, demand is considered a stochastic quantity. An inventory is often subjected to a broad range of demand patterns. Without identifying and distinguishing these patterns, an inventory management policy would probably fail. In many publications, inventory models are classified rather than demand models. The figure below presents a proposed classification of inventory demand patterns based on a study of demand for inventory items (Kobbacy & Liang, 1999).

**Types of Demand**

Source: [Mojtaba Nabipour] (2023), based on [Khairy A.H. Kobbacy] (1999, pg 358).

Demand patterns that are statistically predictable have relatively smooth and repetitive fluctuations and are analyzed using statistical forecasting techniques. Demand patterns can vary over time and are classified into two categories: time-dependent and time-independent. For the former, the quantity to replenish is determined by the time at which the decision to replenish is taken, which is not the case with the latter. A time-independent demand may be assumed to be constant if it is reasonably stable and can be accurately forecasted. Time-independent demand has two subcategories: absolutely constant demand (with minimal variation) and quasi-constant demand (with variation below a specified threshold). Whenever demand has substantial uncertainty, it is called “stochastic,” which can be expressed by theoretical or empirical probability distributions. Time-dependent demand is categorized based on the change factor.

A seasonal variation, trend, or both may be responsible for the time dependency of demand. Due to lumpy or uneven demand, statistically unpredictable demand patterns undergo sudden fluctuations. Such demand may follow approximative or nonapproximative demand patterns. With an approximative demand pattern, an item's demand goes through severe but regular peaks and dips that do not recur annually but at predictable subannual intervals. In contrast, nonapproximative demand patterns are characterized by severe and unpredictable peaks and dips in demand that do not occur predictably. Such demand patterns cannot be predicted by conventional forecasting techniques, including statistical techniques, but can be approximated within a tolerable margin of error by a statistically predictable pattern.

Determining the demand rate in a purely objective manner is rarely feasible. Instead, one may exploit the knowledge of inventory managers. By closely monitoring inventory levels and sales trends, they can identify patterns in customer behavior and adjust inventory levels accordingly. The knowledge and expertise of inventory managers play a crucial role in effectively managing demand rates. By closely monitoring inventory levels and sales trends, inventory managers gather valuable insights into customer behavior and purchasing patterns, which allows them to make informed decisions regarding inventory management and adjust stock levels accordingly.

The knowledge of inventory managers is useful in several ways. By closely monitoring inventory levels, they identify patterns and trends in customer demand. They can analyze sales data to determine which products are popular, which are declining in demand, and any seasonal or cyclical patterns that may influence customer purchasing behavior. This knowledge enables them to anticipate fluctuations in demand and to efficiently adjust inventory levels to meet customer needs.

Moreover, inventory managers can use their knowledge to optimize inventory-replenishment strategies. By analyzing sales trends and monitoring inventory turnover rates, they can determine the appropriate timing and quantities for replenishing stock, which helps avoid stock-outs and excess inventory, both of which can lead to financial losses and customer dissatisfaction. Thus, the knowledge of inventory managers is invaluable in understanding customer behavior, optimizing inventory replenishment, improving forecasting accuracy, and implementing effective inventory control strategies.

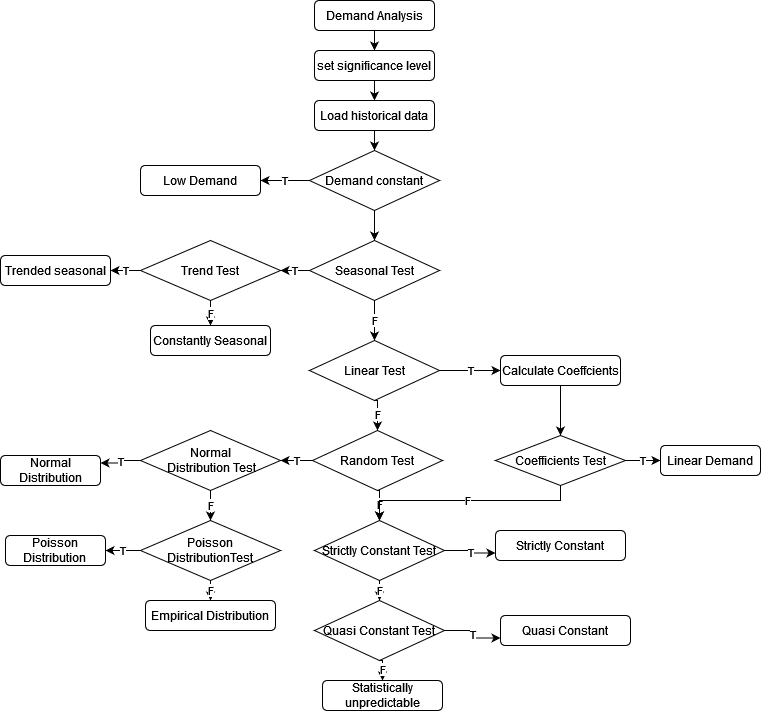
### Identification Flowcharts

To determine the pattern of historical demand, we can perform some statistical tests to distinguish between the different types of demand shown in the figure above. To begin, we should look for any seasonal movement in historical demand and separate that from basic demand. The stationary, linear, and probabilistic demand types should then be identified. This process is shown in the flowchart in the figure below.

Statistical tests can be used to detect seasonal variations in demand. The Kruskal–Wallis test is a nonparametric rank-based test that compares two or more samples to determine whether they belong to the same distribution (Ostertagova et al., 2014). The mathematical expression of the test

(Eq.13)

The Identification Flowchart



Source: [Mojtaba Nabipour] (2023), based on [Eva Ostertagová] (2014, 3).

where *n* is the sum of sample sizes for all samples, *g* is the number of samples, *Ti* is the sum of ranks in sample *i*, and *ni*= size of sample *i*.

The following steps allow one to test the historical demand:

1. Sort data from different samples into ascending order.
2. Compute the summation of rank for each sample separately.
3. Compute the H test (test-statistic from Equation 13).
4. Find the [critical chi-squared (](https://www.statisticshowto.com/how-to-find-a-critical-chi-square-value/)*[α](https://www.statisticshowto.com/how-to-find-a-critical-chi-square-value/)*[) value](https://www.statisticshowto.com/how-to-find-a-critical-chi-square-value/) with *L* degrees of freedom (*L* = *g* − 1, where *g* is the number of seasons).
5. If *H* > *α*, then the demand is seasonal.

Statistical tests are also available for detecting trends in data collected over time. The Mann–Kendall test determines whether a trend in data is statistically significant or whether the data are increasing or decreasing. Another similar test is the Spearman-rank correlation, which is used interchangeably with the Mann–Kendall test (note that the Mann–Kendall test is more robust) (Libiseller & Grimvall, 2002).

### Self-check Questions

1. Explain a stationary demand pattern.

*A stationary demand pattern is a type of demand pattern that remains constant over time. In other words, the mean and variance of the demand remain the same over time. This means that the demand pattern has no trend or seasonality.*

2. What exactly does the Kruskal–Wallis test do?

*The Kruskal–Wallis test is a statistical test that compares two or more samples to determine whether they belong to the same distribution. It is a nonparametric rank-based test, so it does not assume that the data follow a normal distribution.*

3. How is the seasonality of a demand pattern defined?

*To define the seasonality of a demand pattern, look for patterns that repeat over a fixed period of time. Seasonality refers to regular and predictable fluctuations in demand that occur at fixed intervals, such as daily, weekly, monthly, or yearly. Using statistical methods such as autocorrelation analysis or spectral analysis, identify seasonality in a demand pattern. These methods can help identify periodic patterns in the data and determine whether they are statistically significant.*

## 1.4 Multiperiod Newsvendor Model

A multiperiod newsvendor model is an extension of the classical newsvendor problem in which unsold items or unfulfilled demand are carried over to deal with in the next period. Thus, the inventory level of each period, instead of the first, does not equal the ordering level. The problem is finding the optimal inventory level for each period to maximize profit (Matsuyama, 2006). Several models and assumptions have been proposed to achieve this. In this section, we use the approach proposed by Matsuyama (Matsuyama, 2006). For convenience, we use the previously listed notation instead of the original notation. We also ignore some of the theorems discussed and their proofs. More details are available in the original paper. The extra notation is introduced in the following.

### Model Consideration

The model defines how to handle unsold commodities or unfulfilled demand for the next period. Quantity ordering for period *i* is planned according to the following rules:

* Buying items is always accomplished so that the inventory level *li* satisfies *l*min ≤ *li* ≤ *l*max.
* If the inventory level is greater than the demand during the current period, the difference *li* − *xi* indicates the unsold stock. Some fraction of the unsold items are carried over to the next period according to *α*(*li* − *xi*), where 0 ≤ *α* ≤ 1. The holding cost of this fraction would be *hiα*(*li* – *xi*), where *hi* is the holding cost of unsold items for period *i*. Carrying this portion over to the next cycle gives the ordering quantity for period *i* + 1 as *li*+1 − *α*(*li* – *xi*).
* Unsatisfied demand remaining at the end of the current period cost a penalty *Cu*(*xi* – *li*). If this period is not the last, the fraction *β*(*xi* – *li*) of the demand is fulfilled in the next period (0 ≤ *β* ≤ 1). In this case, *li*+1 + *β*(*xi* – *li*) units are sold at the beginning of the next period with the selling price *δsi* + (1 − δ)*si*+1  for each unit (0 ≤ *δ* ≤ 1).

### Profit function

To define the profit function for the multiperiod newsvendor model, we introduce four auxiliary functions:

(Eq.14)

(Eq.15)

(Eq.16)

(Eq.17)

Function *gi* is the profit from selling items during period *i* and function *di* is what we consider doing with the unsold items or unfulfilled demand at the end of period *i*and the start of period *i* + 1. The function *di* returns zero for the terminal period. Therefore, function *di* for a nonterminal period is

(Eq.18)

Assuming no setup cost if we have just one period, the profit is

. (Eq.19)

In the case of two periods, the profit is

(Eq.20)

The general recursive definition of the profit function for an *N*-period newsvendor model is

. (Eq.21)

Defining the profit of an inventory planning during an *N*-period interval gives a nonrecursive form and leads to

(Eq.22)

Since *x* is a random variable, the profit becomes a random variable, and the expected value of *P* is

(Eq.23)

where *f*(*xi*) is the density function of the random variable *xi*.

The problem of determining the optimal inventory level for period *i* can be solved by differentiating the expected value of the profit, setting it equal to zero, and solving the equation:

(Eq.24)

**Example:** Compute the optimal inventory level for a retailer during a period consisting of 20 cycles with the following assumptions:

**α** = 0.8, **β** = 0.7, **δ** = 0.5, **Cu** = 20

1/20 50 ≥ xt ≥ 30

ft(xt) =

0 else

st = 200 + 0.8t

wt = 80 + 0.6t

ht = 2.5 + 0.03t

**Answer**

To simulate the inventory, the random generation of the demand level *xt* populates the following table for 20 cycles.

The inverse of the uniform cumulative probability is

*I*(*F(xt)*) = *30* + *F(xt)*(*50* − *20*)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| t | s | w | h | x | F(L\*) | L\* |
| 1 | 200.8 | 80.6 | 2.53 | 39.4 | 0.760858 | 45.21716 |
| 2 | 201.6 | 81.2 | 2.56 | 34 | 0.759572 | 45.19145 |
| 3 | 202.4 | 81.8 | 2.59 | 34.9 | 0.758294 | 45.16588 |
| 4 | 203.2 | 82.4 | 2.62 | 39.6 | 0.757022 | 45.14045 |
| 5 | 204 | 83 | 2.65 | 48.6 | 0.755758 | 45.11516 |
| 6 | 204.8 | 83.6 | 2.68 | 47 | 0.7545 | 45.09001 |
| 7 | 205.6 | 84.2 | 2.71 | 42.4 | 0.75325 | 45.06499 |
| 8 | 206.4 | 84.8 | 2.74 | 32.8 | 0.752006 | 45.04011 |
| 9 | 207.2 | 85.4 | 2.77 | 35 | 0.750768 | 45.01537 |
| 10 | 208 | 86 | 2.8 | 48.8 | 0.749538 | 44.99075 |
| 11 | 208.8 | 86.6 | 2.83 | 35.4 | 0.748314 | 44.96627 |
| 12 | 209.6 | 87.2 | 2.86 | 35.3 | 0.747096 | 44.94192 |
| 13 | 210.4 | 87.8 | 2.89 | 35.5 | 0.745885 | 44.91771 |
| 14 | 211.2 | 88.4 | 2.92 | 49.1 | 0.744681 | 44.89362 |
| 15 | 212 | 89 | 2.95 | 33.9 | 0.743483 | 44.86966 |
| 16 | 212.8 | 89.6 | 2.98 | 48.6 | 0.742291 | 44.84582 |
| 17 | 213.6 | 90.2 | 3.01 | 48.3 | 0.741106 | 44.82211 |
| 18 | 214.4 | 90.8 | 3.04 | 32.7 | 0.739927 | 44.79853 |
| 19 | 215.2 | 91.4 | 3.07 | 34 | 0.738754 | 44.77507 |
| 20 | 216 | 92 | 3.1 | 43 | 0.419941 | 38.39882 |

### Self-check Questions

1. Explain the difference between the classical newsvendor and the multiperiod models.

*The classical newsvendor model assumes only one sales period and that any unsold items cannot be carried over to the next period. The goal of this model is to find the optimal order quantity that maximizes profit given a fixed selling price and a fixed cost per unit. This model assumes that demand is known and follows a specific probability distribution. Conversely, the multiperiod newsvendor model is an extension of the classical newsvendor problem in which unsold items or unfulfilled demand is carried over to be dealt with in the next period. Thus, the inventory level of each period, instead of just the first period, differs from the ordering level. The goal of this model is to find the optimal inventory level for each period to maximize profit. This model considers demand uncertainty and allows more flexibility in managing inventory levels over time.*

## 1.5 Extensions

Section 1.4 extends the classical newsvendor model to a multiperiod problem. Different variables of the classical newsvendor model may be extended to adapt to different real-world scenarios. This section reviews two important extensions to the newsvendor model.

## 1.6 Multiproduct Newsvendor Model

Suppose a retailer has expanded her business from selling only one product to selling a set of products and wants to define an optimal stocking quantity for every product. The problem can be formulated as follows (Choi et al., 2011): To determine the stocking policy or order quantity of each product, the same approach can be used as in the classical newsvendor model. The only difference is that we calculate the profit probability function *πi* separately for each product. Additionally, leftover stock is salvaged for a price *g* per unit to provide more diversification. Thus, the profit probability function is

(Eq.25)

*πi* calculates the profit for a specific inventory item based on different conditions. It incorporates several variables and parameters to capture different scenarios in inventory management. Integrating *πi* gives the expectation of profit for each product. The optimal order quantity Q\* maximizes the profit and can be computed by differentiating the expectation equation, which gives

. (Eq.26)

### Stocking Constraints

If inventory space is limited, the newsvendor model must be reformulated subject to space constraints, which is expressed as follows:

,

where *ai* is the space required for a unit of product *i* and *A* is the total inventory space. We must solve the optimal stocking problem with these constraints. Hadley et al. (Hadley & Whitin, 1963) 10.1016/j.ejor.2005.11.002proposed a dynamic programming solution based on Lagrange multipliers if the optimal solution violates a constraint (Abdel-Malek & Areeratchakul, 2007). The Lagrangian is

(Eq.27)

Since the Lagrangian is concave, the first-order derivative suffices to determine the optimal *Q*:

(Eq.28)

Yao et al. (2006) examine the stocking policy under the assumption that pricing strategies affect demand levels. Considering risk and disruptions in the supply chain makes inventory policies robust and effective. Garvey and Carnoval (2020) proposed an inventory model that considers the propagation of disruptions in a supply chain. It is an updated version of the single-period newsvendor model and is based on Bayesian networks, which consider the propagation of risk throughout the supply chain. This model allows supply chain managers to minimize supply chain risk by directing their attention to the factors that have the most impact on overall risk in the network.

The newsvendor model is the foundation of many existing supply-chain and inventory-optimization models. Stakeholders such as managers, retailers, and producers extend the basic model based on their specific needs and conditions.

### Self-check Questions:

1. Explain several extensions of the classical newsvendor problem.

*1. Multi-period newsvendor model: This extension considers a scenario where unsold items or unfulfilled demand is carried over to the next period. The goal is to maximize profit by finding the optimal inventory level for each period.*

*2. Newsvendor problem with price-dependent demand: This extension assumes that demand is a function of price. The goal is to maximize profit by finding the optimal price and inventory level.*

*3. Newsvendor problem with lead time: This extension considers the time required for an order to be filled and delivered and searches for the optimal order quantity that minimizes expected cost while satisfying demand requirements.*

*4. Newsvendor problem with quality-dependent demand: This extension assumes that product quality affects demand and searches to maximize profit by finding the optimal inventory level and quality level.*

*5. Newsvendor problem with multiple products: This extension considers a scenario with multiple products with different demand patterns and production costs and searches to maximize overall profit by finding the optimal inventory level for each product.*

*These extensions permit more realistic modeling of real-world scenarios and can provide insights into how different factors affect inventory management decisions.*

Summary

Although the economics literature has studied the newsvendor model as a fundamental problem in stochastic inventory control since the eighteenth century, much work remains to be done. This model has been used extensively to analyze supply chains for trendy and perishable products. In recent decades, operational research has focused on the newsvendor model and extended it to model a variety of real-world problems.

Unit 2 – Traditional Methods of Demand Forecasting

**Study Goals**

Upon completing this unit, you will be able to

* define demand forecasting.
* identify the traditional methods for demand forecasting.
* determine the suitable method for a specific demand type.
* apply the methods discussed to predict future demand.

Basic Readings

Hyndman, R. J., & Athanasopoulos, G. (2018). Forecasting: Principles and practice.

Peixeiro, M. (2022). Time series forecasting in Python. Manning Publications.

Shumway, R. H., & Stoffer, D. S. (2017). Time series analysis and its applications: With R examples (3rd ed.). Springer.

Bisgaard, S., & Kulahci, M. (2011). Time series analysis and forecasting by example (1st ed.). John Wiley & Sons.

Further Reading

Lindsey, J. K. (2004). Statistical analysis of stochastic processes in time (Vol. 14). Cambridge University Press.

Capasso, V., & Capasso, V. (2021). Introduction to Continuous-Time Stochastic Processes. Springer.

Bartlett, M. S. (1978). An introduction to stochastic processes: With special reference to methods and applications. Cambridge University Press.

Sepehri, A. (2021). Inventory management under carbon emission policies: A systematic literature review. In Decision Making in Inventory Management (pp. 187-218).

2. Traditional methods of Demand Forecasting

Introduction

Any process dealing with predicting the future of uncertain events can be called forecasting. Demand is an uncertain quantity that may follow a probabilistic distribution and can be impacted by certain factors such as seasonality and trends. Manufacturing managers must forecast future product demand to ensure they have the materials, labor, and capacity to meet them. Resource planning and scheduling occur well before the firm faces product demand. Since retailers and even producers need to minimize their stocking costs, inventory control relies heavily on demand forecasting.

Any supply chain system needs to be resilient to change to be effective. Therefore, managing complex supply chains requires sophisticated forecasting techniques. A historical flow of demand serves to make the forecasts. The assumption behind quantitative forecasting is that historical data adequately describes the scenario. These data are gathered in a successive measurement process from the same source and are called “time series data.” Demand for an item over time can be collected and analyzed as a time series to predict future demand, which helps improve planning. The following is an explanation of traditional methods of demand forecasting. Throughout this unit, historical demand data analysis is explained by using traditional time series analysis methods. These methods rely on statistical concepts, which are briefly introduced as needed.

Traditional demand-forecasting methods span across diverse industries and sectors, offering wide-ranging applications. These methods find relevance in key areas such as production planning, supply chain management, pricing and promotion strategies, financial planning and budgeting, and capacity planning. By employing traditional demand forecasting techniques, organizations are empowered to make well-informed decisions, optimize resource allocation, and effectively navigate dynamic market conditions.

2.1 Exponential Smoothing

Exponential smoothing is a popular method for forecasting time series in a wide range of industries. It is a statistical technique that uses a weighted average of past observations to predict future values. The method assumes that recent data points are more relevant than older ones. It assigns exponentially decreasing weights to past observations. This approach smooths irregularities in the data and provides more accurate forecasts.

Since exponential smoothing belongs to the category of average-based algorithms, we start by briefly describing these algorithms. Generally, they average historical data to forecast future data. Average-based methods offer advantages and disadvantages. The advantages include simplicity, speed, and low computational costs. However, they may not work well for time series with complex patterns or irregular fluctuations.

Exponential smoothing has proven particularly valuable in the retail industry, where precise demand forecasting is vital for inventory optimization and efficient supply chain management. This technique enables retailers to accurately anticipate customer demand, align their inventory levels accordingly, and streamline their operations to meet market requirements. Similarly, the financial sector uses exponential smoothing techniques for market analysis and forecasting stock prices. Applying these algorithms to historical stock data smooths the inherent fluctuations in the data, allowing investors to generate forecasts that provide valuable insights for making well-informed decisions regarding stock trading and investment strategies.

Cumulative average

A time series is a sequence of data points collected over time, typically at regular intervals. It is a set of chronologically ordered observations that can be used to analyze trends, patterns, and behavior over time. Thus, the time series *A* can be defined as a set of observations *Ai* so that .

In its simplest form, forecasting the next data point *Ak+1* is an average of all past data, which is expressed as

(Eq.1)

If the data pattern is horizontal, with data fluctuating randomly over time around a constant mean, then the mean of the historical data can be used for the next forecast.

Average-based methods are applicable to stationary time series data where the statistical properties of the generating process do not change over time.

To better account for abrupt changes in time series data, we include the last *M* observations when taking an average. This method is called “moving average” and involves a window of size *M* that slides forward in time and averages the last *M* observations, as depicted in the figure below. This is an effective way to smooth out short-term fluctuations and requires less memory than the full average method.

**Moving Average Method**

Ein Bild, das Reihe, Text, Diagramm, Screenshot enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

Simple moving average

A simple moving average is expressed as

(Eq.2)

The moving average method treats all data within a window equally, disregarding the fact that more recent observations hold greater significance than older observations. This limitation motivates the use of exponential smoothing, which assigns exponentially decreasing weights to the observations.

Exponential Smoothing

The recursive equation of the exponential smoothing method is

. (Eq.3)

where *α* is the smoothing constant, *yk* is the most recent observation, *Ak* is the most recent forecast, and *Ak+1 is the next forecast.*

Expanding Equation 3 reveals that the method uses exponentially decaying weights:

Continuing this substitution process gives

(Eq.4)

Accordingly, *Ak*+1 is the exponentially weighted average of all past observations. Using *α* = 1/*k* gives the full average (Equation 1). We must have 0 < *α* < 1 and not 0 ≤ *α* ≤ 1. A smaller *α* provides more stable predictions with smoothed random variation, whereas a larger *α* provides the most rapid response to real changes in the time series. Beyond that, no guidelines exist for choosing a value for alpha for a specific scenario. A straightforward method to estimate *α* is to compute the forecast for *α* = 0.1, 0.2, …, 0.9 and to choose the *α* that gives the highest accuracy based on the minimum forecasting error *et*. The forecasting error is the difference between *Yt* and *Ft*, which are the actual and forecasted values of the time series for period *t*: *et* = *Yt* − *Ft*.

A common error-measuring technique is the root-mean-square error (RMSE), which is based on the forecasting error. The RMSE is given by

(Eq.5)

where *ek* is the difference between the predicted and observed values for each individual observation or time period.

Example 1

The table below gives the demand quantities of a product over two years. What is the optimal value of *α*?

|  |  |  |  |
| --- | --- | --- | --- |
| Demand history for two years | | | |
| DATE | DEMAND | DATE | DEMAND |
| Jan(Year1) | 97.6 | **Jan(Year2)** | 89.3 |
| Feb(Year1) | 95.1 | **Feb(Year2)** | 88.5 |
| Mar(Year1) | 90.3 | **Mar(Year2)** | 93.7 |
| Apr(Year1) | 92.5 | **Apr(Year2)** | 92.7 |
| May(Year1) | 89.8 | **May(Year2)** | 94.7 |
| Jun(Year1) | 92.7 | **Jun(Year2)** | 95.3 |
| Jul(Year1) | 94.4 | **Jul(Year2)** | 94.7 |
| Aug(Year1) | 96.2 | **Aug(Year2)** | 95.3 |
| Sep(Year1) | 88.9 | **Sep(Year2)** | 94.7 |
| Oct(Year1) | 90.2 | **Oct(Year2)** | 96.5 |
| Nov(Year1) | 88.2 | **Nov(Year2)** | 99.2 |
| Dec(Year1) | 91 | **Dec(Year2)** | 96.9 |

Source: [Mojtaba Nabipour], (2023).

The forecasting values for *α* = 0.1, 0.3, 0.6, and 0.9 are computed and listed in Table 2 (*Y*^ denotes the forecasted value). The results in Table 2 indicate that the forecasting series with *α* = 0.6 is more consistent with the original data. The computed RMSE confirms this hypothesis.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Forecasted value with different values of α | | | | | |
| Date | Demand | Y^(α=0.1) | Y^(α=0.3) | Y^(α=0.6) | Y^(α=0.9) |
| Jan(Year1) | 97.6 | 97.6 | 97.6 | 97.6 | 97.6 |
| Feb(Year1) | 95.1 | 97.6 | 97.6 | 97.6 | 97.6 |
| Mar(Year1) | 90.3 | 97.35 | 96.85 | 96.1 | 95.35 |
| Apr(Year1) | 92.5 | 96.645 | 94.885 | 92.62 | 90.805 |
| May(Year1) | 89.8 | 96.2305 | 94.1695 | 92.548 | 92.3305 |
| Jun(Year1) | 92.7 | 95.58745 | 92.85865 | 90.8992 | 90.05305 |
| Jul(Year1) | 94.4 | 95.29871 | 92.81106 | 91.97968 | 92.43531 |
| Aug(Year1) | 96.2 | 95.20883 | 93.28774 | 93.43187 | 94.20353 |
| Sep(Year1) | 88.9 | 95.30795 | 94.16142 | 95.09275 | 96.00035 |
| Oct(Year1) | 90.2 | 94.66716 | 92.58299 | 91.3771 | 89.61004 |
| Nov(Year1) | 88.2 | 94.22044 | 91.86809 | 90.67084 | 90.141 |
| Dec(Year1) | 91 | 93.6184 | 90.76767 | 89.18834 | 88.3941 |
| Jan(Year2) | 89.3 | 93.35656 | 90.83737 | 90.27533 | 90.73941 |
| Feb(Year2) | 88.5 | 92.9509 | 90.37616 | 89.69013 | 89.44394 |
| Mar(Year2) | 93.7 | 92.50581 | 89.81331 | 88.97605 | 88.59439 |
| Apr(Year2) | 92.7 | 92.62523 | 90.97932 | 91.81042 | 93.18944 |
| May(Year2) | 94.7 | 92.63271 | 91.49552 | 92.34417 | 92.74894 |
| Jun(Year2) | 95.3 | 92.83944 | 92.45687 | 93.75767 | 94.50489 |
| Jul(Year2) | 94.7 | 93.08549 | 93.30981 | 94.68307 | 95.22049 |
| Aug(Year2) | 95.3 | 93.24694 | 93.72686 | 94.69323 | 94.75205 |
| Sep(Year2) | 94.7 | 93.45225 | 94.1988 | 95.05729 | 95.2452 |
| Oct(Year2) | 96.5 | 93.57702 | 94.34916 | 94.84292 | 94.75452 |
| Nov(Year2) | 99.2 | 93.86932 | 94.99441 | 95.83717 | 96.32545 |
| Dec(Year2) | 96.9 | 94.40239 | 96.25609 | 97.85487 | 98.91255 |

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE(α=0.1) | RMSE(α=0.3) | RMSE(α=0.6) | RMSE(α=0.9) |
| 3.698793311 | 2.939379223 | 2.642176214 | 2.656275036 |

Source: [Mojtaba Nabipour], (2023).

Despite the computational simplicity and memory efficiency of exponential smoothing, this method does not detect trends or seasonality patterns of time series, as is made clear by the figure below.

**Chart of original data and forecasted values**

Source: [Mojtaba Nabipour], (2023).

Holt’s Exponential smoothing

To enhance the capability of the simple exponential smoothing method to detect trends, we add a new parameter called the “trend factor” to the original method to extend the exponential smoothing. The resulting method is called Holt’s exponential smoothing method (Hyndman & Athanasopoulos, 2018). The assumption here is that the series has an additive linear trend and is the sum of its components, so it has a level *L* and a trend *T*. This method uses three equations and two smoothing parameters *α* and *β* to build a forecast model.

In this model, the next value in the series is estimated by

(Eq.6)

Thus, Holt’s exponential smoothing method is like the simple exponential smoothing method with the trend factor accounted for. To estimate the trend factor we use

(Eq.7)

The initial values for *L* and *T* are essential for starting the forecasting process. The specific values for *L*0 and *T*0 depend on the approach used and the available historical data. The method permits the trend to fluctuate and alter its shape as time progresses. The smoothing constant *β* regulates the pace at which the trend adjusts, so if the trend changes rapidly throughout the series, increasing *β* allows the user to catch up with the series. With this particular arrangement, the trend can fluctuate and alter its shape as time progresses. The smoothing constant *β* allows for rapid adaptation to sudden changes in the series. For instance, if the trend changes rapidly, a larger smoothing constant may be necessary to learn the new trend more rapidly.

Finally, we add the level function and trend factor to reach the final estimation:

(Eq.8)

where m is the number of time steps in the future for the estimate.

**Example 2**

Table 3 gives the results of applying Holt’s exponential smoothing method to forecasting the annual ticket demand for an airline between 1989 and 2016 (Hyndman & Athanasopoulos, 2018). The smoothing parameters and initial values for level and trend are estimated by minimizing the sum of the error squared. The parameters are *α* = 0.8321 and *β* = 0.0001. The very small β indicates that the slope is roughly constant over time. From 2016 we are out-sampled and continue with the last row of values and steps *m* to predict the future. The results in the figure below show that the method indeed captures the demand trend.

**Holt's method for estimating the trend pattern**

Source: [Mojtaba Nabipour], (2023).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| The historical data of ticket demand and forecasting values | | | | | |
| Year | Time Step K | Observation(1000) | Level | Trend | Forecast |
| 1989 | 0 |  | 15.57 | 2.102 |  |
| 1990 | 1 | 17.55 | 17.57 | 2.102 | 17.67 |
| 1991 | 2 | 21.86 | 21.49 | 2.102 | 19.68 |
| 1992 | 3 | 23.89 | 23.84 | 2.102 | 23.59 |
| 1993 | 4 | 26.93 | 26.76 | 2.102 | 25.94 |
| 1994 | 5 | 26.89 | 27.22 | 2.102 | 28.86 |
| 1995 | 6 | 28.83 | 28.92 | 2.102 | 29.33 |
| 1996 | 7 | 30.08 | 30.24 | 2.102 | 31.02 |
| 1997 | 8 | 30.95 | 31.19 | 2.102 | 32.34 |
| 1998 | 9 | 30.19 | 30.71 | 2.101 | 33.29 |
| 1999 | 10 | 31.58 | 31.79 | 2.101 | 32.81 |
| 2000 | 11 | 32.58 | 32.8 | 2.101 | 33.89 |
| 2001 | 12 | 33.48 | 33.72 | 2.101 | 34.9 |
| 2002 | 13 | 39.02 | 38.48 | 2.101 | 35.82 |
| 2003 | 14 | 41.39 | 41.25 | 2.101 | 40.58 |
| 2004 | 15 | 41.6 | 41.89 | 2.101 | 43.35 |
| 2005 | 16 | 44.66 | 44.54 | 2.101 | 44 |
| 2006 | 17 | 46.95 | 46.9 | 2.101 | 46.65 |
| 2007 | 18 | 48.73 | 48.78 | 2.101 | 49 |
| 2008 | 19 | 51.49 | 51.38 | 2.101 | 50.88 |
| 2009 | 20 | 50.03 | 50.61 | 2.101 | 53.49 |
| 2010 | 21 | 60.64 | 59.3 | 2.102 | 52.72 |
| 2011 | 22 | 63.36 | 63.03 | 2.102 | 61.4 |
| 2012 | 23 | 66.36 | 66.15 | 2.102 | 65.13 |
| 2013 | 24 | 68.2 | 68.21 | 2.102 | 68.25 |
| 2014 | 25 | 68.12 | 68.49 | 2.102 | 70.31 |
| 2015 | 26 | 69.78 | 69.92 | 2.102 | 70.6 |
| 2016 | 27 | 72.6 | 72.5 | 2.102 | 72.02 |
|  | m |  |  |  |  |
|  | 1 |  |  |  | 74.6 |
|  | 2 |  |  |  | 76.7 |
|  | 3 |  |  |  | 78.8 |
|  | 4 |  |  |  | 80.91 |
|  | 5 |  |  |  | 83.01 |

Source: [Mojtaba Nabipour], (2023).

Holt–Winters smoothing method

Holt’s exponential smoothing method captures the trend in time series forecasting. However, it does not reveal the seasonality patterns, which is where the Holt–Winters smoothing method comes in. Holt (1957) worked with Winters (1960) to extend Holt's exponential smoothing method to include seasonality (Hyndman & Athanasopoulos, 2018). The Holt–Winters method can be applied in two different ways depending on the type of seasonality. If the seasonality is additive, the additive version must be used, whereas, if the seasonality is multiplicative (i.e., when the time series is the product of its components), the multiplicative version must be used. The three components required to describe the model are level *L*, trend *T*, and season *S*.

The additive Holt–Winters method is expressed as follows:

(Eq.9)

Eq.10)

(Eq.11)

(Eq.12)

Equations 9–11 show how to update the level, trend, and seasonality (with *h* seasons). The seasonal smoothing factor *γ* acts just like *α* and *β*. In an additive series, we use subtraction to decompose the different components, whereas in a multiplicative series, we use division for decomposition. Here, the level component is subtracted to update the seasonal component (Equation 11). Equation 12 makes the forecast with the additive seasonality and additive trend taken into account.

If the series has a multiplicative seasonal component, Equations 9, 11, and 12 take the form (Makridakis et al., 2008)

(Eq.13)

(Eq.14)

(Eq.15)

The only difference is how seasonality composes (multiplication instead of addition) and decomposes (division instead of subtraction). Moreover, to update the seasonality in Equation 14, the level component is removed by division. The adjustments made to the composition (multiplication) and decomposition (division) of seasonality in the Holt–Winters smoothing method are valuable for capturing and accommodating seasonal fluctuations, resulting in a more inclusive and precise forecast. By excluding the level component during the seasonality update, the method ensures that the estimation of seasonality is independent and accurately reflects the inherent seasonal patterns in the data.

One season’s data are required to initialize the level, seasonal, and trend indices. This initialization is performed as follows:

(Eq.16)

(Eq.17)

(Eq.18)

**Example 3:**

The seasonal demand for an item in a hypothetical hardware shop is recorded quarterly (see Table 4). The model assumes a combination of multiplicative seasonality and additive trend. Therefore, Equations 13–15 are used to estimate the trend and forecast future demand. The forecasting process begins in the second season because it requires data from a complete season before it can commence.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Holt–Winters method to capture the seasonality of the quarterly demand data | | | | | | | |
| Year | Quarter | t | Demand | Lt | Tt | St | At+h |
| 1994 | 1 | 1 | 500 |  |  | 1.333333 |  |
|  | 2 | 2 | 350 |  |  | 0.933333 |  |
|  | 3 | 3 | 250 |  |  | 0.666667 |  |
|  | 4 | 4 | 400 | 375 | -12.5 | 1.066667 |  |
| 1995 | 1 | 5 | 450 | 396.9667 | -9.05333 | 1.273412 | 483.3333 |
|  | 2 | 6 | 350 | 372.3747 | -10.6072 | 0.935307 | 362.0524 |
|  | 3 | 7 | 200 | 296.7938 | -17.1046 | 0.668827 | 241.1783 |
|  | 4 | 8 | 300 | 287.3869 | -16.3348 | 1.059833 | 298.3352 |
| 1996 | 1 | 9 | 350 | 302.1219 | -13.2278 | 1.23893 | 345.161 |
|  | 2 | 10 | 200 | 252.9623 | -16.821 | 0.891905 | 270.2048 |
|  | 3 | 11 | 150 | 201.4173 | -20.2934 | 0.691596 | 157.9377 |
|  | 4 | 12 | 400 | 268.2504 | -11.5807 | 1.189227 | 191.9611 |
| 1997 | 1 | 13 | 550 | 373.5062 | 0.102908 | 1.309011 | 317.9958 |
|  | 2 | 14 | 350 | 363.8087 | -0.87713 | 0.912946 | 333.2237 |
|  | 3 | 15 | 250 | 317.4823 | -5.42206 | 0.720351 | 251.002 |
|  | 4 | 16 | 550 | 406.7605 | 4.047961 | 1.238103 | 371.1103 |
| 1998 | 1 | 17 | 550 | 465.9614 | 9.563264 | 1.270414 | 537.7528 |
|  | 2 | 18 | 400 | 444.9496 | 6.505758 | 0.908756 | 434.1286 |
|  | 3 | 19 | 350 | 410.5851 | 2.418728 | 0.759978 | 325.2062 |
|  | 4 | 20 | 600 | 487.3071 | 9.84905 | 1.236049 | 511.3412 |
| 1999 | 1 | 21 | 750 | 597.7855 | 19.91199 | 1.265679 | 631.5942 |
|  | 2 | 22 | 500 | 570.255 | 15.16774 | 0.899169 | 561.3363 |

Source: [Mojtaba Nabipour], (2023).

In this scenario *α*, *β*, and *γ* are set to 0.4, 0.1, and 0.3, respectively. The figure below shows that the method reasonably captures the seasonality of the demand data.

**Capturing Seasonality Component of the Time Series Using Holt–Winters Technique**

Source: [Mojtaba Nabipour], (2023).

Self-check Questions:

1. Given the following table as a 6-month demand history, calculate the RMSE for an exponential smoothing model with *α* = 0.54.

|  |  |
| --- | --- |
| Date | Demand |
| Jan | 89.5 |
| Feb | 95.6 |
| Mar | 94.3 |
| Apr | 92.1 |
| May | 95.8 |
| Jul | 88.7 |

2. In Example 3, use the multiplicative trend with additive seasonality model and then compare the RMSE of the resulting model that of the original model.

2.2 Autoregressive Integrated Moving Average (ARIMA)

A different approach to time series forecasting is provided by autoregressive integrated moving average (ARIMA) models (Box et al., 2015). The ARIMA model is a statistical technique developed in the 1970s that is widely used to forecast future values of a time-series variable based on its past values. The ARIMA model assumes that time-series data can be decomposed into three components: trend, seasonality, and randomness. The trend component represents the long-term direction of the time-series data, the seasonality component represents the recurring patterns in the data, and the randomness component represents the unpredictable fluctuations in the data. In time series forecasting, exponential smoothing and ARIMA models are complementary approaches. Exponential smoothing uses trend and seasonality to describe data, whereas ARIMA uses autocorrelations to describe data (Hyndman & Athanasopoulos, 2018). Mathematically, autocorrelation gives the statistical relationship or correlation between a variable and its lagged values. As the name suggests, ARIMA uses autoregressive and moving average processes to model the time series, so we begin by explaining these two processes. However, we first consider the concept of stationarity.

Stationarity holds when the statistical properties of a stochastic process, such as the one that generates a time series, are constant in time. Therefore, time series with trends cannot be stationary because the mean of the data is not constant over time. In another example, a seasonal time series cannot be stationary because it lacks constant variance.

Autoregressive process

In an autoregressive model, the variable of interest is expressed as a linear combination of its lagged values when the values of that variable occur at previous time points. The formal definition of an autoregressive process of order *p* is

(Eq.19)

where AR is the autoregressive process and *εt* is white noise or an error term with zero mean and a variance of σ2. A wide range of different time series patterns can be handled by autoregressive models (Hyndman & Athanasopoulos, 2018). Φ coefficients are the process parameters, which determine the degree to which the current value depends on the lagged values. The parameter *C* is the constant of the process; it can be considered as the noise average and sometimes does not appear separately.

**Example:**

The figure below shows the series generated from the AR(2) process expressed by

(Eq.20)

**An Autoregressive Time Series Generated by Equation 20**

Source: [Mojtaba Nabipour], (2023).

The variance of white noise does not change the pattern but only changes the scale (see the following figure).

**Increasing the White Noise Variance to 1.5**

Source: [Mojtaba Nabipour], (2023).

The series generated by autoregressive processes are stationary by construction. However, the coefficients *φ* are constrained to make the process stationary (Hyndman & Athanasopoulos, 2018).

For an AR(1) process, .

For an AR(2) process, .

At higher orders, the constraints become more complicated

Moving Average Process

Instead of using past values to estimate the current value, a moving average (MA) process uses linear combinations of past forecast error terms to forecast future values. The moving average process (the name is similar to what we saw before, but the concept differs significantly) assumes that a stationary process is a random fluctuation around a forward moving average of the series (white noise process):

. (Eq.21)

Equation 21 describes the general form of an MA process. The depth of the lagged errors is determined by *q*. Like AR processes, moving average processes can only model stationary time series. To model nonstationary time series with these two methods, we transform the original series to obtain a stationary time series, often through differencing, which essentially calculates the difference between consecutive observations, allowing us to remove the change between them.

Differencing can be done multiple times; the number of times is the “order.” For example, first-order differencing gives

(Eq.22)

Second- and third-order differencing give

(Eq.23)

(Eq.24)

For example, the following figure shows a time series of Dow Jones closing averages. Since the data trend upward, the series cannot be stationary. The following figure shows the results of applying first-order differencing to this time series.

**Dow Jones Time Series**

Source: [Mojtaba Nabipour], (2023).

This transformation eliminates the upward trend in the time series, leaving it with a random pattern with constant mean and variance, which convinces us to consider it a stationary time series.

**First Order Differencing**

Source: [Mojtaba Nabipour], (2023).

Nonseasonal ARIMA Process

As we saw in the previous sections, MA and autoregressive processes can model stationary time series. Differencing allows us to transform nonstationary time series with trend components into stationary time series. These methods may be integrated to produce a pervasive time-series model.

A nonseasonal ARIMA model is produced by combining differencing with autoregression and moving averages. The model is expressed as follows:

(Eq. 25)

The variables *p* and *q* again indicate the lag order in the autoregression and MA processes, respectively (Hyndman & Athanasopoulos, 2018). The order of differencing is given by *d*. ARIMA(*p,0,0)* is an autoregressive process [AR(*p*)] and ARIMA (*0, 0, q*) is an MA process [MA(*q*)].

The following table lists examples of ARIMA processes.

|  |  |
| --- | --- |
| Process | Equation |
| ARIMA(1,1,2) |  |
| ARIMA(1,2,1) |  |
| ARIMA(2,2,0) |  |
| ARIMA(1,0,2) |  |

The differencing order *d* is thus the number of differencing steps required to transform a nonstationary time series into a stationary time series. To determine the appropriate differencing order, we typically assess the stationarity of the time series at different levels of differencing. If the initial time series is highly nonstationary, we start with a first-order differencing (*d* = 1) and check if the resulting time series is stationary.

Below, we discuss the following main stages of ARIMA modeling according to Peixeiro (2022):

* identification test to determine whether the time series is stationary
* choosing appropriate model parameters (i.e., *p, d*, and *q*)
* fitting the best model to the observations
* estimating the coefficients *φ* and *ϴ*
* validating the model

The residuals (i.e., forecasting errors) should form a random pattern; if they are correlated, the model must be changed (Ramos & Oliveira, 2016). Forecasting thorough the model. Moreover, we can use accuracy metrics such as the RMSE. To make the concepts easy to use, we avoid complicated statistical formulas and explain the stages above in Python codes.

To start, we must determine whether a time series is stationary. If not, we must determine the differencing order to apply to obtain stationarity. The time series plot might show if there is a strong upward or downward trend. However, if the time series trend is difficult to judge just by looking at the plot, more sophisticated tools are required.

Some statistical tests can be used to examine the stationarity of time series. One of them is the augmented Dicky–Fuller (ADF) test in which the null hypothesis holds that the series is nonstationary and, if *p* is greater than or equal to the significant value (0.05), the null hypothesis is not rejected and the time series is not deemed stationary (Dickey, 2015).

To get the *p* value, the Python programming language provides the adfuller(time-series) function from the statsmodels package*.*

For example, let us determine the stationarity of the time series of Figure 6.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

The Phython output gives p-valuu = 0.36133347874971455

Since the *p* value exceeds 0.05, the null hypothesis is not rejected, so the time series is not stationary. Subsequently, we ascertain whether the first-order differencing process engenders stationarity within the time series under consideration. To calculate the difference, we use the diff method, which takes the data and number of steps as parameters. The following code snippet shows how to use it.

Ein Bild, das Text, Schrift, weiß, Screenshot enthält.

Automatisch generierte Beschreibung

*p* value = 0.0034683081217674982

The *p* value rejects the null hypothesis, so first-order differencing suffices. Actually, by applying differencing and the ADF test, we can determine the value of *d* in the ARIMA model. The next step is to define the values of *p* and *q*. Parameter *p* is the order of the autoregressive model; it determines how lag correlates with current. Thus, we must analyze the autocorrelation of the time series.

Autocorrelation analysis provides the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF), which can be of significant help not only for determining stationarity but also for model identification (*p, q*). The ACF determines the correlation of the time series with its past values:

(Eq.26)

Note that a lag extending beyond one raises the question of how much information we can gain from *yt*−*k* about *yt* that we cannot obtain from in-between observations. This question guides us to the PACF, which does the same thing as the ACF but first removes the effects of in-between observations. For example, PACF(*yt*, *yt*−3) is the correlation that is not explained by *yt*−1 and *yt*−2. The PACF makes it possible to determine the number of observations required to describe the current value, so it is not surprising that it sees use in ARIMA models to determine the value of *p.* Therefore, the most significant lags in a partial autocorrelation plot decrease the correlation, so the index of the last lag is used to determine *p*.

Likewise, the ACF plot gives similar information about the MA in the ARIMA model. The significant contribution from the past error terms causes the curve to drop below the confidence region, which is typically represented as a band around the predicted values in a time series plot. The width of the band depends on the level of confidence chosen by the analyst, which is often set at 95% or 99%. Thus, we consider *q* to be the last significant lag index.

**Example:**

We are given a data set that records the usage amount of a web server over several minutes. What is the appropriate ARIMA model to describe the data?

We start by plotting the time series:

**Time Series Plot of the Example**

Ein Bild, das Diagramm, Reihe enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

Clearly, the time series is not stationary. Let us start with first-order differencing:

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

The test shows that we cannot reject the null hypothesis. Thus, we continue with second-order differencing. The next figure shows the *p* value of the ADF test and plots the time series after second-order differencing.

**The Plot of the Second Order Differenced Time Series**

Ein Bild, das Text, Schrift, Reihe, Diagramm enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

The plot looks like a stationary time series, and the ADF test suggests that *d* = 2.

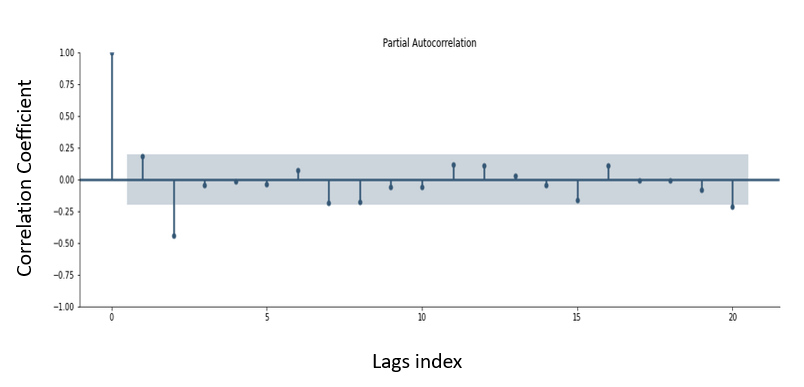
To obtain the *p* value, we use the PACF plot.

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

The PACF plot indicates (see figure below) that there is no significant autocorrelation (shaded area) after the second observation. Thus, we go no further than the second observation for the autoregressive part (*p* = *2*)

**Partial Autocorrelation of the Time Series**



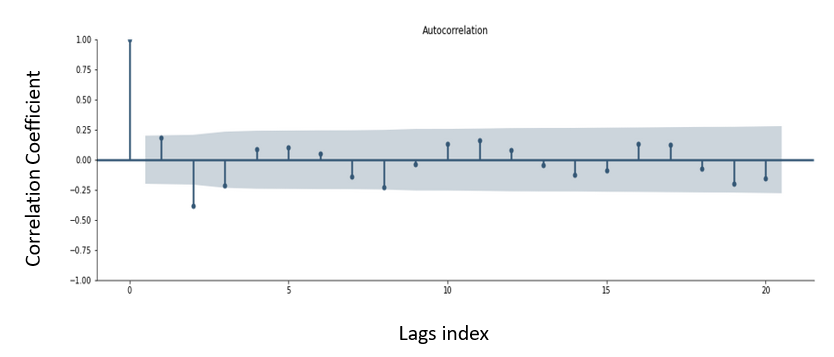
Source: [Mojtaba Nabipour], (2023).

Finally, looking at the last observation that crosses the threshold in the ACF plot (see figure below) allows *q* to be estimated.

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

**Autocorrelation of the Time Series**



Source: [Mojtaba Nabipour], (2023).

The plot above indicates that *q* = 2, so the final ARIMA model is ARIMA(2,2,2).

To determine the coefficients, we fit the model to the data, which is done by using the ARIMA class from the module **statsmodels.tsa.arima.model**.

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

The following figure summarizes the results of the model fitting.

**Summary of ARIMA Fitting Results**

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

We can see how well the model fits by plotting the actual time series alongside the predicted time series (see the figure below).

To predict future values using autocorrelation measures within a time series, ARIMA incorporates autocorrelation measures into the model. It measures the dependence of a particular sample on a few previous observations through autoregression. By measuring and integrating these differences, data patterns become stationary, or the correlation with past data is minimized (linearity and noncollinearity are fundamental assumptions of the linear regression model). Data are then condensed, and significant features are brought out through an MA.

Self-check Questions:

1. Compute the average value of an autoregressive process described by the following equation for 50 iterations:
2. Briefly explain the procedure of ARIMA modeling.
3. *Stationarity: Check if the time series is stationary (i.e., that its statistical properties, such as mean and variance, do not vary over time). If it is not stationary, transform the data to achieve stationarity.*
4. *Differencing: If the data are not stationary, take data differences until they become stationary.*
5. *Autocorrelation and Partial Autocorrelation Analysis: Analyze the autocorrelation and partial autocorrelation functions of the differenced data to determine appropriate values for p, d, and q in the ARIMA model.*
6. *Model Selection: Select an appropriate ARIMA model based on the results of step III.*
7. *Model Fitting: Fit the selected ARIMA model to the data.*
8. *Model Validation: Validate the fitted model by checking its residuals for* randomness and independence.
9. What does the Akaike information criterion (AIC) criterion tell us about the ARIMA model?

*The AIC is a statistical measure that evaluates the goodness of fit of a model by considering the number of parameters used in the model. In ARIMA modeling, the AIC criteria can be used to compare different models and select the model with the lowest AIC value. The lower the AIC value, the better the model fits the data while using fewer parameters. Therefore, AIC criteria can help us determine which ARIMA model is best suited for a given time series data.*

**Predicting Time Series Using the Fitted ARIMA Model**

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

Automatisch generierte Beschreibung

Observations

Time Steps

Source: [Mojtaba Nabipour], (2023).

2.3 State-Space Models

Any time series that we want to model and predict is the result of consecutive observations of a system’s output. A system can be implemented as a finite set of states, some that can be measured and some that cannot – the so-called “hidden states.” As a system transitions between different states, it produces observations and outputs. The state-space model is like a statistical model but with two fundamental differences: The first difference is that observations in state-space models may be noisy or difficult to measure, so state-space model assumptions are more realistic because they explicitly model the underlying system dynamics that generate the observed data. The second difference is the model updating characteristic. As the model runs, it incorporates prior knowledge to improve the prediction. The more interesting advantage the model offers, especially for time series analysis, is its focus on the process that generates noisy data instead of the noisy data itself. In other words, these types of models try to understand the dynamics behind the process rather than describing the relationships between the outcomes of the process.

The general form of output generation (the linear form with additive noise) and state evolution can be expressed as

(Eq.26)

(Eq.27)

where *Zt* describes the linear model of the system that generates the observations, *α* is the state vector through which the system evolves, *T* is the transition function that controls the state evolution (it describes the system dynamics), *d* and *c* are the intercepts, *ε* is the measurement noise, and *η* is the disturbance noise of the system. If the system is Gaussian, then we have

(Eq.28)

(Eq.29)

**Time-series example**

Consider an additive time series where the observations contain seasonal and trend components. We write it as

(Eq.30)

where *μ*, *γ*, and *ε* are the trend, seasonality, and error components, respectively. Suppose we can model the trend component as a simple random-walk process as follows:

(Eq.31)

If *μt* = *αt*, *γt* = 0, and the error is a white noise process, we can write the time series as

(Eq.32)

. (Eq.33)

The time series thus takes the form of a state-space model where Equation 32 is the measurement model that relates observation *y* to state *α*. Equation 33 describes the dynamics of the state evolution.

In the following, we explore the Kalman filter, which is the most used method for state-space models.

The Kalman Filter

The Kalman filtering method is the most developed and deployed method for state-space models. It uses an iterative process to minimize prediction error by incorporating new measurements. Thus, the Kalman filtering process contains two iterative steps (see figure below).

**Kalman Iterative Process**

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

Building on the previous equations for state-space models, the prediction step is expressed as

(Eq.34)

(Eq.35)

The prime (*'*) notation indicates the predicted term prior to observation or measurement, and the superscript *T* indicates the transposition operation and defines a linear relationship between the current state vector and the previous state vector. *P* is the error covariance, which expresses the uncertainty in the state. In the first phase, the next state and error covariance are predicted. Upon receiving a new observation, the correction or filtering phase will begin according to Equations 36–38, where R is the variance in measurement error:

(Eq.36)

Equation 36 computes the Kalman gain, which is used to correct the estimate of the next state by

(Eq.37)

Intuitively speaking, the Kalman gain regulates the impact of prediction error on the state correction part of the estimation. As time advances, the Kalman gain should decrease. Finally, the following equation updates the error covariance:

. (Eq.38)

Kandananond (2014) provides an example of Kalman filtering in demand forecasting.

The first-order autoregressive model is commonly used to explain stationary demand processes:

(Eq.39)

Equation 39 is linear in white noise, so we use Kalman filtering to predict the time series.

The output can be written as

(Eq.40)

where v is the measurement noise. Considering the given system, the Kalman filter equations are

**The time update equations**

(Eq.41)

(Eq.42)

**Measurement update equations**

(Eq.43)

(Eq.44)

(Eq.45)

As an illustration, Kandananond (2014) shows the result for *φ* = 0.9.

**Self-check Questions:**

1. What are the differences between state-space modeling and statistical modeling?

* *Observations: In state-space models, observations may be noisy or difficult to measure, whereas statistical models assume that observations are precise and error-free.*
* *Model Updating: State-space models are characterized by model updating, meaning that, as new data become available, the model is updated to incorporate the new information.*

*Statistical models do not have this characteristic. State-space models also offer several advantages over classical difference-in-differences schemes, including inferring the temporal evolution of attributable impacts, incorporating empirical priors on parameters into fully Bayesian models, and accommodating multiple sources of variation.*

2. Explain the two main stages of Kalman filtering.

* *Prediction: In the prediction step, the Kalman filter uses the previous state estimate and its covariance to predict the current state and its covariance. This prediction is based on the system dynamics model, which describes how the state evolves.*
* *Update: In the update step, the Kalman filter incorporates new measurements to improve its estimate of the current state. The update is based on a comparison of the predicted measurement with the actual measurement, as well as their respective uncertainties.*

*These two steps are repeated iteratively as new measurements become available, with each iteration improving the accuracy of the state estimate. The Kalman filter is widely used in various fields, such as engineering, economics, and finance, because it provides accurate estimates even in noisy environments.*

2.4 Bayesian Structural Time Series Analysis

Analysis of time-series data presents some of the most difficult analytical challenges. A limited amount of data is available to inform some of the most important decisions (e.g., corporate planning), so time series analyses are commonly used to forecast demand, which requires understanding seasonality and trends and quantifying how they affect business. In most cases, however, the lack of historical data makes it difficult to estimate these components accurately. Uncertainty can be handled transparently by Bayesian structural time series models. The time series forecasting, feature selection, and other applications are all done using the Bayesian structural time series (BSTS) method. The BSTS method is a statistical method used in various applications, including feature selection and time series prediction. It is specifically designed to be used with time-series data. Time-series analysis using the Bayesian method has the following advantages:

* Versatility: Bayesian methods offer the flexibility to model complex systems by incorporating prior knowledge and assumptions, which enables more precise and reliable predictions.
* Quantifying Uncertainty: Bayesian methods allow uncertainty in both model parameters and predictions to be quantified. This is particularly valuable in time-series analyses where data can be noisy and subject to uncertainty.
* Model Comparison: Bayesian methods provide a framework for comparing different models and selecting the most suitable model based on the available data.
* Integration of External Information: Bayesian methods facilitate the integration of external information, such as expert knowledge or additional data sources, into the modeling process.
* Real-time Adaptation: Bayesian methods can be updated in real time as new data become available, allowing for more accurate and timely predictions.

Overfitting is a problem with time series models, especially when estimating models with many parameters over a short period. Despite not being a problem in this case, it can be problematic when dealing with multiple variables, such as in economic forecasting. To solve the overfitting problem, we need to take a Bayesian approach, which allows us to prioritize our variables. As a result, it can be used to assess the extent to which different marketing campaigns contribute to changing product sales, brand popularity, and other relevant indicators related to product sales and brand popularity. Several factors can be considered in state-space models as opposed to classical difference-in-differences schemes, including inferring the temporal evolution of attributable impacts, the incorporation of empirical priors on parameters into fully Bayesian models, and the ability to accommodate multiple sources of variation, including contemporaneous covariates that change over time.

A Bayesian structural time series model with unobserved components is used. Compared with ARIMA models, this technique is more transparent and deals elegantly with uncertainty. Due to its lack of differencing, observations, and moving averages, it is also more transparent (model components can be visually inspected). It handles uncertainty better by quantifying posterior uncertainty, controlling variance, and imposing prior beliefs on the model. In addition, any ARIMA model can be recast as a structural model. The Bayesian structural model can be expressed as follows:

(Eq.46)

where *xt* is a set of regressors, *St* is seasonality, and *μt* is the local level term. Latent state evolution over time is referred to as an unobserved trend at the local level. For instance, it can represent underlying growth in brand value or external factors that are hard to track. It can also absorb short-term fluctuations that should be explicitly controlled. As a result, we can avoid strange estimates of coefficients due to spurious relationships. If the historical data cannot yield meaningful estimates, the outside priors for the means can be specified by the Bayesian nature of the model.

BSTS uses linear Gaussian models that are similar to Kalman filtering to provide more insight into the constituent components of series such as seasonality, trend, and regression (Fildes, 1991).

The Bayesian approach tries to estimate the distribution of future states with the help of previous observations and previous state distributions. This approach is based on the Bayes theorem:

. (Eq.47)

Here, θ is the hidden state, *p*(θ|*y*) is the posterior distribution given observation *y*, and *p*(y) is the marginal probability of *y* and is given by

. (Eq.48)

The state evolution can be expressed as

(Eq.49)

where *T* is the transition function and can include all the components mentioned. Given the normal distribution of the error terms (*η* and *ɛ*), the posterior distribution of the states is always less uncertain (Mathys et al., 2011).

The state transition is usually modeled by a Markov chain. Using Markov chain Monte Carlo sampling, one can simulate complicated distribution functions. We can forecast the time series by using the general model (Equation 46) and Bayesian inference.

To clarify, we provide an example of using the BSTS method to forecast time-series data. Suppose we want to forecast the monthly sales of a retail store for the following year. We have historical sales data for the past five years, and we want to use these data to build a forecasting model. We can use the BSTS method to model the time series data. The BSTS method decomposes the time series into several components: trend, seasonality, and other relevant factors. The method then uses Bayesian inference to estimate the parameters of each component. The following steps show how we can apply the BSTS method to our example:

* Data preparation: We start by preparing the data by converting it into a time-series format. We then visualize the data to identify any trends, seasonality, or other patterns.
* Model specification: Next, we specify the model by defining the components used to model the time series. For example, we can include a linear trend component, a seasonal component, and other relevant factors such as promotions or holidays.
* Prior specification: We then specify the prior distributions for each parameter in the model, allowing us to incorporate prior knowledge or assumptions about the underlying system.
* Bayesian inference: We use Bayesian inference to estimate the posterior distribution of each parameter in the model. This strategy involves updating the prior distributions with the observed data to obtain the posterior distributions.
* Forecasting: Finally, we use the estimated model parameters to make forecasts for future time periods. The forecasts should include uncertainty intervals that reflect the uncertainty in the model parameters.

Self-check Questions:

1. How does Bayesian inference reduce the uncertainty of the forecasting procedure?

*Bayesian inference is a statistical approach that uses prior knowledge and data to update the probability of a hypothesis. In the context of time-series forecasting, Bayesian inference can reduce uncertainty by incorporating prior knowledge about the system being modeled. By using Bayesian methods, we can incorporate into our analysis prior beliefs about the model’s parameters,allowing us to make more informed predictions and reduce forecast uncertainty. Additionally, Bayesian methods allow us to update our beliefs as new data become available, which can further improve the accuracy of our forecasts. Overall, Bayesian inference provides a powerful tool for reducing uncertainty in time-series forecasting by allowing us to incorporate prior knowledge and update our beliefs as new data become available.*

**Summary**

Traditional methods of demand forecasting rely heavily on statistical analyses of time series. Such methods assume that the underlying demand pattern follows a certain pattern and that future demand will be similar to past demand. Statistical analysis allows for the identification of seasonality, trends, and other patterns in the data, which can provide insight into future demand. Methods such as exponential smoothing, moving average, and ARIMA models assume that the time series are stationary with fixed variance and the observations are available. In contrast, Kalman filtering and BSTS methods try to estimate hidden mechanisms that generate the observations. Thus, they can forecast variables that may not be observable. However, this approach requires much more information and computation than the first category of methods.

Unit 3 – Data-Driven Methods for Demand Forecasting

**Study Goals**

Upon completing this unit, you will be able to

* define data-driven methods.
* identify the data-driven methods for demand forecasting.
* determine which method is suitable for a specific demand type.
* apply the methods discussed to predict future demand.

Basic Reading

Blokdyk, G. (Year). Recurrent neural network: real life actions: practical tools for self-assessment.

Joseph, M. (2022). Modern Time Series Forecasting with Python: Explore industry-ready time series forecasting using modern machine learning and deep learning. Packt Publishing.

Further Reading

Dunis, C., Zhou, B., & Robinson, A. (Year). Applied Time Series Analysis with Neural Networks.

Akhundov, S., & Veys, Y. (Year). Time Series Forecasting with Deep Learning: A Survey.

Brownlee, J. (Year). Time Series Forecasting with Python and Deep Learning.

Smith, M. (Year). Time Series Forecasting: A Machine Learning Approach.

3. Data-Driven methods of Demand Forecasting

Introduction

Data-driven methods for demand forecasting use statistical and machine learning techniques to analyze historical data and predict future demand for a product or service. These approaches depend on having access to substantial quantities of data, which can be gathered from diverse sources such as sales logs, customer actions, and market patterns. By examining these data, companies can obtain valuable information about consumer behavior and use it to make well-informed choices regarding inventory control, production scheduling, and marketing tactics. Data-driven demand forecasting can help businesses optimize their operations, reduce costs, and improve customer satisfaction by ensuring that the right products are available at the right time and in the right quantities. Traditionally, demand forecasting is modeled as a time series consisting of linear components. Often, the real world involves more complex models. Artificial intelligence techniques such as supervised learning and neural networks can help us handle this complexity and do more accurate modeling. This unit thus familiarizes the reader with the concepts of these techniques and their application in demand forecasting.

3.1 Recurrent Neural Networks

Recurrent neural networks (RNNs) are a specific category of artificial neural networks that handle sequential data. Unlike traditional feedforward neural networks that process input data sequentially, RNNs can retain information from past inputs, enabling them to efficiently process input sequences. RNNs possess a memory that enables them to capture temporal relationships within data. This quality makes them highly suitable for tasks such as speech recognition, language translation, and time series prediction. The fundamental component of an RNN is the recurrent neuron, which takes an input and a hidden state and generates an output and a new hidden state. The hidden state functions as the network’s memory, enabling it to retain information from previous inputs. The output from the recurrent neuron can be reintroduced into the network as input for the subsequent time step, allowing the processing of input sequences. However, training RNNs is greatly hindered by the vanishing gradient problem, where the gradients used to update network weights diminish considerably as they propagate backward in time. (Hewamalage et al., 2019). This can make it difficult for the network to learn long-term dependencies in the data.

Numerous versions of RNNs have been created to tackle this issue, such as long short-term memory (LSTM) networks and gated recurrent units, both of which incorporate distinct gating mechanisms that regulate the information flow throughout the network. Overall, RNNs are a powerful tool for processing sequential data and have been used in a wide range of applications (Joseph, 2022). However, they can be computationally expensive to train and may require careful tuning of hyperparameters to perform well. The figure below shows the general structure of a feedforward neural network. A feedback path for neurons is provided in the hidden layer to add the memory capability to the network to reach RNNs.

**General Architecture of a Feedforward Neural Network**

Ein Bild, das Screenshot, Text, Kreis enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

To fully understand how the feedback structure works, focus on the hidden layers unfolding in time in the figure below.

**Recurrent Neural Network**

Ein Bild, das Diagramm, Screenshot, Reihe, Electric Blue (Farbe) enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

The figure below shows three time steps of the evolution of the hidden layers. The temporal input data *x* change the behavior of the hidden layers and of the output.

**Unfolding the hidden layers for 3 time steps**

Ein Bild, das Text, Schrift, Reihe, Diagramm enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

The functions *a*(*t*) and *y*(*t*) are expressed as follows:

|  |  |
| --- | --- |
|  | (Eq.1) |
|  | (Eq.2) |

where W*ij* are the weights determined by training, and *g*1 and *g*2 are the activation functions.

Depending on the number of *x* and *y*, there are five types of RNN (see the table below).

|  |  |
| --- | --- |
| Number of X and Y | Type of RNN |
| N(X)=N(Y)=1 | One-to-One |
| N(X)=1 , N(Y)>1 | One-to-Many |
| N(X)>1 , N(Y)=1 | Many-to-One |
| N(X)=N(Y)>1 | Many-to-Many |
| N(X)<>N(Y) | Many-to-Many |

Source: [Mojtaba Nabipour], (2023).

The loss function of an RNN network is the sum of all time-step loss functions:

|  |  |
| --- | --- |
|  | (Eq.3) |

The backpropagation is done at each point in time. The derivative of the loss function with respect to the weight matrix is

|  |  |
| --- | --- |
|  | (Eq.4) |

Ordinary recurrent neural networks have two shortcomings:

* Vanishing or exploding gradient problems
* Gradient clipping

The vanishing gradient issue in RNNs refers to the gradients becoming extremely small as they propagate through the network during backpropagation. The origin of this phenomenon is that, in each time step, the gradients are subject to the network weights. When these weights are small, the gradients become small. Consequently, the network has difficulty acquiring long-term dependencies because the insufficient gradients do not efficiently update the weights. This drawback can cause inadequate execution and a delayed convergence rate in RNNs (Hebbar et al., 2022).

Similarly, the predictions become unstable and inaccurate if the gradient update is too much. This issue is particularly common in RNNs because they have feedback loops that amplify the gradients over time, leading to an exponential increase in their magnitude. The exploding gradient problem can be addressed by using gradient clipping, which limits the maximum size of the gradients, or by using more stable RNN architectures such as LSTM or gated recurrent units.

Gradient Clipping

Gradient clipping is a technique to overcome the exploding gradient problem by preventing the gradient from exceeding a specified limit. The figure below illustrates the technique with a graph.

**Gradient Clip Illustration**

Ein Bild, das Text, Reihe, Screenshot, Diagramm enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

Some types of RNNs are designed to use specific gates to remedy the vanishing gradient problem in a specific way.

These gates are used to build LSTM network cells and will be described in detail in the following subsections.

Long-Short Term Memory (LSTM) networks

Long short-term memory neural networks are recurrent neural networks designed to handle long dependencies in sequential data. They are highly beneficial in various applications, including time-series forecasting, speech recognition, language translation, and image captioning. These tasks involve input data that consists of a sequence of words or images (Nithin et al., 2022). LSTM networks can selectively retain or discard information from previous time steps, which is achieved by using “memory cells” that store information over time, along with “gates” that regulate the flow of information in and out of these cells. These gates are implemented as sigmoid functions, which consider the current input and the previous hidden state as input and produce a value between zero and one. This value determines the extent to which the input and previous state are transmitted to the subsequent time step. The forget gate decides how much of the previous memory cell to retain, while the input gate determines how much current input should be added to the memory cell. Finally, the output gate determines the fraction of the current memory cell that should be passed on to the next time step (Abbasimehr et al., 2020). LSTM networks can preserve important information from past time steps while discarding irrelevant details, allowing them to handle long-term dependencies effectively and overcome the “vanishing gradient” issue commonly encountered in conventional RNNs. Consequently, LSTM networks are highly advantageous for sequential data modeling and consistently achieve cutting-edge performance across various tasks.

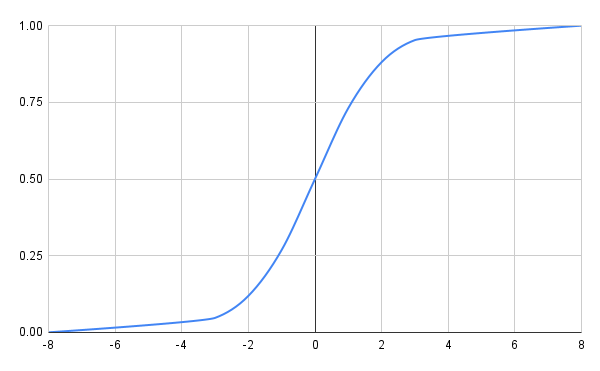
Prior to delving into the cell structure, we briefly revisit the sigmoid and hyperbolic tangent functions. These mathematical functions are extensively used in machine learning and artificial intelligence algorithms.

The sigmoid function is a type of activation function used in neural networks to introduce nonlinearity into the output of a neuron. It is defined as

|  |  |
| --- | --- |
|  | (Eq.5) |

where *x* is the input of the function. The sigmoid function transforms any input into a value ranging from zero to one, making it useful for binary classification tasks. As the input to the function increases, the output approaches one. For smaller inputs, the output approaches zero. The figure below shows a visual representation of the function.

**Sigmoid Graph**



Source: [Mojtaba Nabipour], (2023).

The hyperbolic tangent function is another type of activation function used in neural networks. It is defined as

|  |  |
| --- | --- |
|  | (Eq.6) |

The hyperbolic tangent function transforms its input into a range of −1 to +1 and is a valuable tool for regression tasks. As the input increases, the output gradually approaches +1; for small inputs, the output gradually approaches −1. Both the sigmoid and hyperbolic tangent functions are frequently used in machine learning algorithms to introduce nonlinear behavior into neuron outputs. This nonlinearity enables neural networks to grasp intricate patterns within the data. The figure below graphs the hyperbolic tangent function.

**Graph of TANH**

Ein Bild, das Reihe, Diagramm, parallel enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

LSTM Cell Structure

An LSTM cell can learn patterns over long data sequences by selectively retaining or discarding information. The functioning of an LSTM cell can be divided into four key components: the input gate, the forget gate, the output gate, and the cell state.

1. **Input Gate:** The input gate identifies the relevant information from the current input and previous hidden state that should be added to the cell state. Using a sigmoid activation function, the input gate calculates a value between zero and one for each input element, thereby determining the significance of incorporating the corresponding information into the cell state.
2. **Forget Gate:** The forget gate determines which information from the previous cell state should be retained or discarded. It takes the current input and previous hidden state as inputs and applies a sigmoid activation function, resulting in a value ranging from zero to one for each element in the cell state. This value signifies the degree to which each piece of information should be retained or discarded.
3. **Cell State:** The cell state serves as the memory of the LSTM cell. It stores information from previous time steps and updates it based on the input gate and forget gate. The input gate determines the information to be added to the cell state, and the forget gate determines the information to be retained or discarded. The cell state is updated by multiplying the previous cell state by the forget gate and adding the input gate multiplied by the current input.
4. **Output Gate:** The output gate determines which information within the current cell state is suitable for use as the hidden state. It takes inputs from the current input and previous hidden state and applies a sigmoid activation function to each element in the cell state, resulting in a value between zero and one. This value indicates the suitability of the data for use as the hidden state. The cell state then goes through a tanh activation function to generate the actual hidden state, which is multiplied by the output gate to produce the final output.

**Structure of LSTM Cell**

Ein Bild, das Text, Diagramm, Screenshot, Plan enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

The LSTM cell selectively retains or discards information based on the input and previous state and updates its memory accordingly (Brownlee, 2018). This approach allows it to handle long-term dependencies in sequential data and avoid the vanishing gradient problem that plagues traditional RNNs. The figure above shows the details of an LSTM cell with the input gate and forget gate. *C* is the cell state, and *h* is the hidden state.

**Implementation**

To implement an LSTM network, we use Python with the Keras library. The implementation steps are summarized as follows:

**Step 1: Import the necessary libraries**

import numpy as np

import pandas as pd

from keras.models import Sequential

from keras.layers import Dense, LSTM

**Step 2: Load the dataset**

data = pd.read\_csv('your\_dataset.csv')

**Step 3: Preprocess the data**

*Split the data into training and testing sets*

train\_data = data[:800]

test\_data = data[800:]

# Normalize the data

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature\_range=(0, 1))

train\_data\_scaled = scaler.fit\_transform(train\_data)

test\_data\_scaled = scaler.transform(test\_data)

# Create the input and output sequences

def create\_sequences(data, seq\_length):

X = []

y = []

for i in range(len(data)-seq\_length-1):

X.append(data[i:(i+seq\_length), 0])

y.append(data[i+seq\_length, 0])

return np.array(X), np.array(y)

seq\_length = 10

X\_train, y\_train = create\_sequences(train\_data\_scaled, seq\_length)

X\_test, y\_test = create\_sequences(test\_data\_scaled, seq\_length)

# Reshape the input sequences

X\_train = np.reshape(X\_train, (X\_train.shape[0], X\_train.shape[1], 1))

X\_test = np.reshape(X\_test, (X\_test.shape[0], X\_test.shape[1], 1))

**Step 4: Build the LSTM model**

model = Sequential ()

model.add(LSTM(units=50, return\_sequences=True, input\_shape=(X\_train.shape[1], 1)))  
model.add(LSTM(units=50))  
model.add(Dense(units=1))  
model.compile(optimizer='adam', loss='mean\_squared\_error')

**Example 1:**

We have monthly demand history data saved in a CSV file. An LSTM model has been implemented using Python. The steps coded in COLAB are described below.

The most difficult part is reformatting the data correctly to feed the model.

Read the dataset and store it in an array.

Ein Bild, das Text, Schrift, Screenshot enthält.

Automatisch generierte Beschreibung

The figure below plots the data.

**The Plot of Monthly Demand Dataset**

Ein Bild, das Text, Screenshot, Reihe, Diagramm enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

The time series seems to have seasonality and trend patterns. We can decompose them with the following code:

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

The following figure shows the decomposition results, including seasonality, trend, and noise patterns.

**Results of Demand Dataset Decomposition**

Ein Bild, das Text, Reihe, Schrift, Diagramm enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

The next step is to split the data set into train and test sets, as shown in the code below.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

The range of numbers should be limited to get better results. To do so, we use the MinMaxScaler function from the preprocessing library of the sklearn package.

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte BeschreibungEin Bild, das Text, Schrift, weiß, Screenshot enthält.

Automatisch generierte Beschreibung

The data now range from zero to one.

The following code reformats the training set data into batches ready to feed the model.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

We use batches of 12 months of demand data to train the model. The output of the model is the prediction of the following month.

The next step is defining the model, which is relatively straightforward.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

The network model created can be visualized with the following code:

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

The following figure shows the network diagram.

**The Network Created by the Model**

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

The model is trained by using the *fit* method with 50 epochs.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

The following loss graph shows how well the training has gone.

**The Loss Graph of the Training Procedure**

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

**The Predicted vs Realized Demand**

Ein Bild, das Diagramm, Reihe, Steigung enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

To test the model, the last 12 values of the training dataset are selected as a starting batch to predict the test dataset.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

After making the predictions, the scaled values must return to their original values.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung



The figure above plots the predictions.

LSTM is better suited for complex data with nonlinear relationships and long-term dependencies. Considering computational resources, statistical methods are better suited for simpler data with linear relationships and short-term dependencies.

Self-check questions:

1. How do RNNs handle sequential data, and what advantages does this give them over other models?

***Answer****:* *By sequentially analyzing individual data points and retaining an internal state that preserves previous information, RNNs generate predictions for the present data point. Updating the internal state at each time step enables RNNs to grasp both temporal and long-term dependencies within the input sequence. In contrast, conventional feedforward neural networks and other machine learning models cannot capture these dependencies, rendering them inappropriate for sequential data.*

2. What is the main advantage of LSTM networks over traditional RNNs?

***Answer:*** *The main advantage of LSTM networks over traditional RNNs is the former’s ability to handle the vanishing gradient problem, which is a common problem when training RNNs.*

3. How does an LSTM network handle vanishing gradients, which can be challenging when training RNNs?

***Answer:*** *LSTM networks solve this problem by introducing a memory cell and three gating mechanisms: the input gate, forget gate, and output gate. The memory cell is designed to retain information about the previous inputs and can control the flow of information through the gates.*

4. What are the main components of an LSTM cell, and how do they work together to process sequential data?

***Answer:***

1. *Memory Cell: The memory cell is the main component of the LSTM cell and is responsible for storing and updating information about the input sequence over time. It acts as a conveyor belt that passes information from one time step to the next.*
2. *Input Gate: The input gate controls the flow of information into the memory cell. It is a sigmoid function that takes the current input and the previous hidden state as input and decides how much new information should be added to the memory cell.*
3. *Forget Gate: The forget gate controls the flow of information out of the memory cell. It is a sigmoid function that takes the current input and the previous hidden state as input and decides how much old information should be removed from the memory cell.*
4. *Output Gate: The output gate controls the flow of information from the memory cell to the next hidden state. It is a sigmoid function that takes the current input and the previous hidden state as input and decides how much information from the memory cell should be output to the next hidden state.*

*The LSTM cell processes sequential data as follows:*

1. *At each time step, the current input and the previous hidden state are passed to the input gate and forget gate, respectively.*
2. *The input gate decides how much new information should be added to the memory cell, and the forget gate decides how much old information should be removed from the memory cell.*
3. *The memory cell updates its state based on the input from the input gate and the forget gate and stores the new information.*
4. *The output gate decides how much information from the memory cell should be output to the next hidden state.*
5. *The updated hidden state is returned as the output of the LSTM cell and is also used as input to the next time step.*

5. How can the forget gate in an LSTM network be adjusted to improve its performance on a given task?

***Answer:*** *The forget gate in an LSTM network controls the flow of information out of the memory cell, deciding how much old information should be removed from the cell. Adjusting the forget gate can improve the network's performance on a given task. Here are some ways to adjust the forget gate:*

1. *Change the Activation Function: The forget gate is typically implemented as a sigmoid function that takes the input, current hidden state, and previous cell state as input. By changing the activation function, we can modify the behavior of the forget gate. For example, using a rectified linear unit activation function can improve the network’s performance on tasks that require nonlinear transformations.*
2. *Use Different Initialization Methods: The forget gate’s weights are typically initialized randomly, which may lead to poor performance. Changing the initialization method to something like Xavier initialization or He initialization may improve the network’s performance by ensuring that the weights are initialized near the optimal values.*
3. *Add Regularization: Regularization techniques such as dropout or weight decay can prevent overfitting and improve network performance. By applying regularization to the forget gate, we can prevent it from overfitting the training data and improve its generalization ability.*
4. *Adjust the Learning Rate: The learning rate controls the step size of the gradient descent algorithm during training. By adjusting the learning rate, we can control the extent to which the network weights are updated during each training iteration. If the forget gate is not learning properly, we can adjust the learning rate to encourage more learning or less learning.*
5. *Change the Number of Units: The number of units in the forget gate can be adjusted to change the network size. Increasing or decreasing the number of units controls the complexity of the network and improves its performance on a given task.*

3.2 Supervised Learning

Supervised learning is a type of machine learning algorithm where the computer is trained on a labeled dataset, meaning that the data are already categorized or classified. The algorithm learns to recognize patterns in the input data and maps those patterns to the correct output or label. The goal of supervised learning is to train a model that accurately predicts the correct output for new, unseen data. In supervised learning, the input data are often referred to as the “features” or “predictors,” while the output data are referred to as the “targets” or “labels.” The labeled dataset is split into a training set, which is used to train the model, and a testing set, which is used to evaluate the model’s performance.

There are two main types of supervised learning algorithms: regression and classification. Regression algorithms are used when the output is a continuous variable, such as predicting the price of a house based on its features. Classification algorithms are used when the output is a categorical variable, such as predicting whether an email is spam based on its content.

Supervised learning has many applications, including image recognition, speech recognition, natural language processing, and fraud detection. It is a powerful tool for solving complex problems and making accurate predictions based on data. Supervised learning algorithms can be used for time series forecasting as well. One frequently employed method for time series forecasting with supervised learning involves utilizing regression algorithms, like linear regression or neural networks (discussed in the preceding section). The algorithm learns from a labeled dataset comprising past variable values and their corresponding future values. The input data consist of a sequence of past values, and the output data are the predicted future values.

Another approach entails employing classification algorithms such as decision trees or random forest. In this scenario, the output data are categorical variables that indicate whether future values will exceed a specific threshold. The algorithm is trained on a labeled dataset of past values and their corresponding binary labels, and it learns to classify new data based on the input sequence. Time series forecasting using supervised learning has many applications, including predicting stock prices, weather forecasting, and energy demand forecasting. It can help businesses make informed decisions based on future trends and improve operations by optimizing resource allocation and inventory management.

Demand forecasting using supervised learning involves the following steps:

* **Collect and preprocess historical demand data:** Collect and preprocess historical demand data, which includes the number of units sold or the revenue generated for a product or service over a period of time. Preprocessing may involve cleaning the data, handling missing values, and transforming the data if necessary.
* **Split the data into training and validation sets:** Split the historical demand data into training and validation sets. The training set is used to train the model, and the validation set is used to evaluate the model’s performance.
* **Choose a suitable model:** Choose a suitable model for demand forecasting. RNNs such as LSTM networks are commonly used for time series forecasting.
* **Train the model:** Train the model on the training set using an appropriate loss function such as the mean squared error (MSE) or mean absolute error. The model should learn to map past demand data to future demand data.
* **Evaluate the model:** To assess the effectiveness of the model, analyze its performance on the validation set using evaluation measures such as MSE or mean absolute error. If the model’s performance falls short of expectations, modify the model’s hyperparameters or try alternative architectural designs.
* **Make predictions:** Once the model is trained and validated, use it to predict new, unseen demand data. Overall, demand forecasting using supervised learning involves collecting and preprocessing historical demand data, splitting the data into training and validation sets, choosing a suitable model, training the model, evaluating the model’s performance, and making predictions on new data.

The following sections explain two major regression-based methods.

Support Vector Regression

Support vector regression (SVR) is a supervised learning technique for regression tasks. Its objective is to determine a function that estimates the continuous value of a target variable based on one or more input features. The underlying concept of SVR involves identifying a hyperplane within the given feature space that optimizes the distance between predicted and actual values. SVR strives to discover a line that maximizes the separation between data points while still capturing the overall data trend. To accomplish this, SVR uses a kernel function to transform input features into a higher-dimensional space, facilitating the identification of a hyperplane that effectively separates data points. The algorithm then strives to locate the hyperplane that maximizes the margin between predicted and actual values.

In SVR, a kernel is a function that maps the input features into a higher-dimensional space where it is easier to find a hyperplane that separates the predicted values from the actual values. The kernel function transforms the input data into a new representation that may be more suitable for linear separation. SVR uses several kernel functions, the most frequently used of which are the linear kernel, polynomial kernel, and radial basis function kernel. The selection of a kernel function is influenced by the characteristics of the data and the particular problem being addressed. The kernel trick allows the hyperplane to be computed in the high-dimensional feature space without actually computing the coordinates of the data points in that space. This approach makes SVR computationally efficient even for large datasets.

SVR distinguishes itself from other regression models by searching for the optimal line within a predefined threshold rather than minimizing the discrepancy between actual and predicted values. (Zhang & O'Donnell, 2020). Hyperplanes and boundary lines are separated by threshold values. The complexity of SVR fits increases superquadratically with sample number. This limitation means that scaling to datasets with more than 10 000 samples is challenging.

The Python implementation of SVR is given below.

First, we import the necessary libraries:

import pandas as pd

import numpy as np

from sklearn.svm import SVR

from sklearn.metrics import mean\_squared\_error

Next, we load the time series data into a Pandas DataFrame and split it into training and testing sets:

# Load data

data = pd.read\_csv('time\_series\_data.csv', index\_col='date')

# Split data into train and test sets

train\_data = data.iloc[:100]

test\_data = data.iloc[100:]

We then prepare the data by creating lag features. We create a function that takes a time series and a lag value as inputs and returns a DataFrame with the original time series and its lagged values:

def create\_lag\_features(data, lag):

df = pd.DataFrame(index=data.index)

df['y'] = data['y']

for i in range(1, lag+1):

df[f'lag\_(Hewamalage et al.)'] = data['y'].shift(i)

df.dropna(inplace=True)

return df

We apply this function to our training and testing sets to create lag features:

# Create lag features for training set

train\_df = create\_lag\_features(train\_data, lag=3)

# Create lag features for testing set

test\_df = create\_lag\_features(test\_data, lag=3)

Next, we split the lagged time series into *X* and *y* variables for training and testing, respectively:

# Split lagged time series into X and y variables for training set

X\_train = train\_df.drop('y', axis=1) y\_train = train\_df['y']

# Split lagged time series into X and y variables for testing set

X\_test = test\_df.drop('y', axis=1) y\_test = test\_df['y']

We then fit an SVR model to the training data:

# Fit SVR model to training data

svr = SVR(kernel='linear', C=1e3, epsilon=0.1)

[svr.fit](https://svr.fit" \t "_blank)(X\_train, y\_train)

Finally, we use the trained SVR model to make predictions based on the testing data and calculate the mean squared error:

# Make predictions on testing data

y\_pred = svr.predict(X\_test)

# Calculate mean squared error

mse = mean\_squared\_error(y\_test, y\_pred)

print(f'MSE: {mse:.2f}')

Some advantages and disadvantages of SVR are listed below.

**Pros of SVR:**

* handles nonlinear relationships between input and output variables
* handles high-dimensional data
* handles outliers in the data
* works for both regression and classification tasks
* can be trained with different kernel functions to handle different types of data

**Cons of SVR:**

* requires careful selection of hyperparameters, such as the kernel function and regularization parameter
* is sensitive to the choice of kernel function and hyperparameters, which can lead to overfitting or underfitting
* is computationally expensive for large datasets or complex kernel functions
* requires a large amount of data to train effectively
* may not perform well with noisy or sparse data
* model accuracy not maintained for large datasets

Self-check questions:

1. What is the purpose of supervised learning in machine learning?

***Answer****: Supervised learning is a machine learning technique that involves training a model to make predictions based on input and output pairs, where the output is a known label or target value corresponding to the input. The primary objective of supervised learning is to learn a mapping between the input and output variables so that the model can accurately predict the output for new, unseen input data. By developing a mapping function to predict the output for new input data, the labeled dataset teaches the algorithm how to make predictions.*

2. How does training a supervised learning model differ from testing it?

***Answer****: To summarize, training a supervised learning model involves teaching the model to make precise predictions based on labeled examples, whereas testing the model assesses how it performs on new and unseen data. Differences between the two processes include variations in data, objectives, evaluation metrics, procedures, and the risk of overfitting.*

3. What is the main difference between SVR and other regression techniques?

***Answer****: To summarize, SVR differs from other regression techniques in two main ways: First, it handles outliers by only considering the support vectors, which are the data points closest to the decision boundary. Second, it allows for highly nonlinear decision boundaries by using kernel functions, which capture complex shapes that other regression techniques may be unable to handle.*

4. How does SVR handle nonlinear relationships and outliers in the data?

***Answer****: Support vector regression addresses nonlinear relationships and outliers in the data through kernel functions that transform the input space into a higher-dimensional feature space. This is done by considering support vectors that are the data points closest to the decision boundary and by tuning parameters such as the regularization parameter and kernel function to optimize the model’s performance. These features make SVR a versatile and robust regression technique suitable for various applications.*

5. What techniques can determine the optimal value for the regularization parameter in SVR?

***Answer****: Finding the optimal value for the regularization parameter in SVR can be achieved through various techniques such as grid search, random search, cross-validation, or Bayesian optimization. The selection of the optimal approach depends on factors such as dataset size, available computational resources, and the desired level of accuracy.*

3.3 Effects of Correlation and Confounding

Correlation pertains to the connection between two variables, wherein alterations in one variable are linked to changes in the other. For time-series forecasting, correlation can help recognize patterns and trends within the data, enabling more precise predictions. Correlations can also lead to misleading associations, where two variables seem connected, but the relationship lacks causality. Not accurately understanding such a relationship can lead to inaccurate forecasts.

**Confounding**

This refers to a third variable that affects the relationship between two other variables.

In time-series forecasting, confounding can lead to inaccurate predictions if the effect of the confounding variable is not properly considered. For example, suppose a company’s sales are forecast based on historical data, but a major competitor enters the market during the forecast period. In that case, the competitor’s impact on sales must be considered to make accurate predictions. To mitigate the impact of correlation and confounding on data-driven time-series forecasts, the data must be carefully analyzed to identify potential relationships or confounding variables. This analysis can involve using statistical techniques such as regression analysis or time series decomposition to identify patterns and trends in the data. Additionally, it is important to consider external factors that may impact the forecast, such as changes in the market or industry. Properly accounting for correlation and confounding can make data-driven time-series forecasts more accurate and reliable.

Statistically analyzing the correlation and confounding effect in a time series before forecasting is an important step in ensuring the accuracy of the forecast. The steps for doing so are listed below:

* **Collect the data:** Collect the time-series data to analyze. These data should include the dependent variable (the variable you want to forecast) and any independent variables that may affect the dependent variable.
* **Plot the data:** Plot the time series data to visualize the trends and patterns in the data. This helps identify any outliers or anomalies in the data.
* **Calculate the correlation:** Calculate the correlation between the dependent variable and each independent variable. This helps identify any variables that correlate strongly with the dependent variable.
* **Check for confounding effects:** Ensure no confounding factors exist among the independent variables. Confounding factors arise when two independent variables are associated, which makes it challenging to identify the variable responsible for variations in the dependent variable.
* **Use regression analysis:** Use regression analysis to determine the relationship between the dependent variable and each independent variable. This analysis helps identify any significant predictors of the dependent variable.

For example, suppose you want to forecast the sales of a particular product over the next year. You collect data on product sales over the past five years, as well as data on product price, marketing spending, and product seasonality.

To statistically analyze the correlation and confounding effect in this time series, you plot the data to visualize any trends or patterns. You notice that sales tend to increase during certain months of the year, indicating seasonality. Next, you calculate the correlation between sales and each independent variable (price, marketing spending, and seasonality). You may find that sales correlate negatively with price, positively with marketing spend, and strongly with seasonality.

You then check for any confounding effects between the independent variables. For example, you may find that marketing spending and seasonality correlate strongly, which could make it more difficult to determine which variable changes sales.

Finally, you use regression analysis to determine the relationship between sales and each independent variable. This analysis helps identify the significant predictors of sales and uses them to forecast future sales.

**Example:**

We have an inventory cost for 19 months for three products. Determine the most significant predictors.

This is done by using a regression analysis.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Month | Inventory cost | Product A | Product B | Product C |
| 1 | 44439 | 515 | 541 | 928 |
| 2 | 43936 | 929 | 692 | 711 |
| 3 | 44464 | 800 | 710 | 824 |
| 4 | 41533 | 979 | 675 | 758 |
| 5 | 46343 | 1165 | 1147 | 635 |
| 6 | 44922 | 651 | 939 | 901 |
| 7 | 43203 | 847 | 755 | 580 |
| 8 | 43000 | 942 | 908 | 589 |
| 9 | 40967 | 630 | 738 | 682 |
| 10 | 78582 | 1113 | 1175 | 1050 |
| 11 | 45003 | 1086 | 1075 | 984 |
| 12 | 44303 | 843 | 640 | 828 |
| 13 | 42070 | 500 | 752 | 708 |
| 14 | 44353 | 813 | 989 | 804 |
| 15 | 45968 | 1190 | 823 | 904 |
| 16 | 47781 | 1200 | 1108 | 1120 |
| 17 | 43202 | 731 | 590 | 1065 |
| 18 | 44074 | 1089 | 607 | 1132 |
| 19 | 44610 | 786 | 513 | 839 |

Source: [Mojtaba Nabipour], (2023).

We use Microsoft Excel to do the analysis (a data analysis toolpack is in the data tab).

The dependent variable *Y* is inventory cost and the independent variables *X* form three product columns. The figure below summarizes the analysis.

**Linear Regression Analysis using Excel**

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung

Source: [Mojtaba Nabipour], (2023).

We inspect the *p* values to determine the most significant predictors. If the *p* value of a predictor is greater than 0.15, we exclude the variable from the regression because it does not significantly impact the prediction of the outcome. In this case, the *p* value of product A is greater than 0.15, so we exclude it and run the analysis again for regression-based prediction (see the figure below). The linear regression searches for a linear relationship between the independent variables and the dependent variable. The formula is

(Eq.7)

The regression analysis shows that, to predict the outcome *Y* using the linear formula, the constant value should be replaced with the intercept value, *a*1 with the coefficient for product A, *a*2 with the coefficient for product B, and *a*3 with the coefficient for product C. However, as the *p*-value column suggests, we can eliminate the variable *x*1 from the equation.

**Regression Analysis after refining**

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung

Self-check questions

1. In the previous example, assuming the following values for the products, what is the predicted outcome?

Product A = 1200

Product B = 800

Product C = 1000

2. What is the difference between confounding and correlation in the context of supervised learning?

***Answer****: To summarize, correlation and confounding in supervised learning are two distinct concepts. Correlation refers to a statistical relationship between variables, whereas confounding refers to a situation where the effect of one variable is mixed with the effect of another variable, and this mixture can affect the accuracy of predictive models. Different strategies are available to overcome confounding variables, such as controlling for confounding variables, stratifying the data, or using techniques such as propensity score matching or instrumental variable analysis.*

3. Can a strong correlation between predictor variables cause overfitting in a supervised learning model?

***Answer****: A high degree of correlation between predictor variables can cause overfitting in a supervised learning model by providing redundant information that can lead to unstable estimates, inflated standard errors, and reduced statistical power.*

4. Which steps can be taken to address issues related to correlation and confounding in supervised learning?

***Answer***: *To mitigate this problem, techniques such as feature selection, regularization, or dimensionality reduction can be used to reduce the complexity of the model and improve its generalizability.*

5. How can you determine the significance of the relationship between the predictor variable and the response variable in a regression model?

***Answer****: Various methods are available to assess the importance of the connection between the predictor variable and the response variable in a regression model. These include calculating the correlation coefficient, conducting hypothesis testing, evaluating the coefficient of determination, and examining the confidence interval of the regression coefficient. These techniques help determine both the strength and statistical significance of the relationship between the predictor and response variables.*

3.4 Big-Data Newsvendor Model

The big-data newsvendor model is a statistical approach used in supply chain management to enhance inventory management and reduce expenses. It is founded on the well-known newsvendor problem, an optimization challenge within operations research. In the traditional newsvendor problem, a newsstand proprietor must determine the appropriate quantity of daily newspapers to order to maximize profits. Although the actual demand for newspapers is uncertain, the owner can use historical data to estimate demand. The objective of the newsvendor problem is to identify the order quantity that yields the highest expected profits given the uncertain demand.

The big-data newsvendor model extends this problem to incorporate additional data sources such as weather data, social media data, and sales data from other retailers. By incorporating these additional data sources, the model can more accurately predict demand and optimize inventory levels.

For example, a retailer might use weather data to predict how much ice cream will sell on a hot day or use social media data to predict which products will likely be popular. By incorporating these additional data sources into the newsvendor model, the retailer can optimize inventory levels and minimize costs. Overall, the big-data newsvendor model is a powerful tool for supply chain management that leverages statistical models and machine learning algorithms to optimize inventory levels and minimize costs. This section reviews two recent studies on this matter.

Ban and Rudin (2019) used contemporary machine learning methods, including algorithmic stability theory and quantile regression, to examine the newsvendor dilemma in situations where the decision-maker possesses both historical demand data and a substantial set of demand-related features. The authors used algorithmic stability theory and quantile regression, which are techniques from machine learning, to analyze the newsvendor problem. They suggest two methods to determine the optimal order quantity using a dataset that includes both demand and related feature observations. One method involves applying empirical risk minimization with or without regularization, a machine-learning principle. The second method, kernel-weights optimization, draws inspiration from the Nadaraya-Watson kernel-regression technique. These approaches are discussed in more detail in Section 2 of (Ban & Rudin, 2019). The authors apply these approaches to develop a sample average approximation algorithm that handles big data and provides practical insights for decision-makers. The past works against which the authors compare their algorithms are based on incorporating exogenous information in inventory decision-making. Specifically, they compare their work with (Liyanage & Shanthikumar, 2005), who use a Bayesian approach to incorporate demand forecasts and other information into the newsvendor problem. This is discussed in more detail in Section 2.4.1 of (Ban & Rudin, 2019).

Huber et al. (2019) used machine learning to establish the ideal order quantity by applying a data-oriented technique to solve the newsvendor problem. They classify inventory-management data-driven approaches into three levels and use a normal distribution in the second level, which distinguishes it from the data-driven approach. To optimize the process, they select a specific quantile of the corresponding demand distribution. In the third level, they use a parametric quick response (QR) model, such as artificial neural networks, and estimate its parameters by resolving problem (8) instead of minimizing the MSE. The outcome of this procedure is the order quantity.

The authors compare their technique with established methods across three levels of data-driven approaches in inventory management. They use their real-world dataset to demonstrate that data-driven methods surpass model-based methods in most instances. They also empirically examine how choices made at each level affect performance. The current literature lacks clarity on how these choices influence performance.

Clausen and Li (2022) used machine learning techniques to identify the most suitable order quantity in a dynamic inventory model that uses the principle of empirical risk minimization. They developed a machine learning algorithm to address the problem and used genuine big data from businesses to empirically investigate the efficiency of their inventory model and solution algorithm. The numerical results of the study reveal that their comprehensive big-data-driven model yields cost reductions of up to 60% compared with the most successful single-variable benchmark model and up to 6.37% cost savings compared with the most successful big-data-driven benchmark model.

This literature demonstrates the potential of machine learning and data-driven approaches for improving inventory management decisions. By leveraging large quantities of data and advanced algorithms, these approaches can provide more accurate demand forecasts, optimize inventory levels, and reduce costs. However, challenges remain to be addressed, such as the need for high-quality data and the difficulty of interpreting complex machine learning models. In conclusion, this literature highlights the importance of incorporating machine learning and data-driven approaches into inventory management research and practice. As technology continues to advance, we can expect to see more innovative solutions that leverage these approaches to improve supply chain efficiency and effectiveness.

Self-check Questions

1. How can machine learning algorithms be used to improve newsvendor decision making?

***Answer****: Machine learning algorithms can enhance newsvendor decision making by improving demand forecasts, inventory optimization, dynamic pricing, and customer segmentation. By leveraging these techniques, newsvendors can improve their competitiveness, profitability, and customer satisfaction.*

2. What are some limitations of using big data in newsvendor analysis?

***Answer****: Although big data offers new avenues for improving newsvendor analysis, its use is not without limitations. These include factors such as data quality, volume, privacy, interpretability, and cost. Addressing these limitations requires careful planning, expertise, and adherence to legal and ethical standards.*

3. What are the main challenges faced by newsvendors working with big data?

***Answer****: Newsvendors face several challenges when working with big data, including data integration, quality, privacy, analysis, infrastructure, and cost. Addressing these challenges requires careful planning, expertise, and investment in technology and resources.*

**Summary**

Different approaches to demand forecasting, such as recurrent neural networks, supervised learning, correlation and confounding effect analysis, and the big-data newsvendor model, each have their own advantages and disadvantages. Recurrent neural networks work well for capturing intricate patterns in time-series data but necessitate a substantial amount of data and can be computationally intensive. Supervised learning is a straightforward and efficient method for demand forecasting but assumes that future demand resembles past demand. Analyzing correlation and confounding effects helps identify the factors influencing demand but requires a deep understanding of the business and market. The big-data newsvendor model handles vast amounts of data and generates accurate demand forecasts but necessitates a significant investment in data infrastructure and analytical capabilities. In general, data-driven methods for demand forecasting offer valuable insights and enhance decision-making, but their strengths and limitations must be carefully considered. Choosing the appropriate approach depends on the specific needs of the business and the available data.

Abdel-Malek, L. L., & Areeratchakul, N. (2007). A quadratic programming approach to the multi-product newsvendor problem with side constraints. *European Journal of Operational Research*, *176*(3), 1607-1619.

Burgin, T., & Wild, A. (1967). Stock control—Experience and usable theory. *Journal of the Operational Research Society*, *18*(1), 35-52.

Choi, S., Ruszczyński, A., & Zhao, Y. (2011). A multiproduct risk-averse newsvendor with law-invariant coherent measures of risk. *Operations Research*, *59*(2), 346-364.

Chopra, S., Meindl, P., & Kalra, D. V. (2007). *Supply Chain Management by Pearson*. Pearson Education India.

Garvey, M. D., & Carnovale, S. (2020). The rippled newsvendor: A new inventory framework for modeling supply chain risk severity in the presence of risk propagation. *International Journal of Production Economics*, *228*, 107752.

Hadley, G., & Whitin, T. M. (1963). *Analysis of inventory systems*.

Kobbacy, K. A., & Liang, Y. (1999). Towards the development of an intelligent inventory management system. *Integrated Manufacturing Systems*, *10*(6), 354-366.

Libiseller, C., & Grimvall, A. (2002). Performance of partial Mann–Kendall tests for trend detection in the presence of covariates. *Environmetrics: The official journal of the International Environmetrics Society*, *13*(1), 71-84.

Matsuyama, K. (2006). The multi-period newsboy problem. *European Journal of Operational Research*, *171*(1), 170-188.

Ostertagova, E., Ostertag, O., & Kováč, J. (2014). Methodology and application of the Kruskal-Wallis test. Applied mechanics and materials,

Reid, R. D., & Sanders, N. R. (2023). *Operations management: an integrated approach*. John Wiley & Sons.

Ross, S. M. (2017). *Introductory statistics*. Academic Press.

Snyder, R. D. (1984). Inventory control with the gamma probability distribution. *European Journal of Operational Research*, *17*(3), 373-381.

Vandeput, N. (2020). De Gruyter. <https://doi.org/doi:10.1515/9783110673944>

Walpole, R. E., Myers, R. H., Myers, S. L., & Ye, K. (2011). Probability and Statistics for Engineers and Scientists, 9th. In: Pearson.

Yao, L., Chen, Y. F., & Yan, H. (2006). The newsvendor problem with pricing: Extensions. *International Journal of Management Science and Engineering Management*, *1*(1), 3-16.