Impact of Machine Learning on Demand Forecasting In Supply Chains: A Case Study on Global Manufacturing Company

Roshan Pandey[1]

[1] Harrisburg University of Science and Technology

Author Note

I would like to thank my advisor Arnie Miles for helping me with this research. This paper consists of Introduction to the topic of Supply Chain Analytics and how various econometric / statistical/ Machine Learning algorithms can be applied for demand forecasting. The different sections of this research are Introduction, Related Literature, Hypothesis, Method, Result and Conclusion. This document is created for Harrisburg University of Science and Technology GRAD695 and ANLY699 courses.

Correspondence concerning this article should be addressed to Roshan Pandey, 326 Market St, Harrisburg, PA 17101. E-mail: rpandey@my.harrisburgu.edu

Abstract

As global supply chains have become sophisticated over the years, demand planning has become sophisticated as well. Companies need to be plan well in advance to consider the lead-time between when a supply order is placed, and the product gets delivered in warehouses for companies to sell. As companies grow and sell items across various channels, an automated way to forecast demand is essential. Forecasting has become essential part of demand planning activities so that companies can proactively plan the resources as well as materials. The goal of good demand planning is to have right products at the right time for companies to fulfill customer demands with and at the same time not have overages, so it does not negatively impact financial stability. Traditionally, Excel has been a goto tool to do forecasting for most companies, however, it has become increasingly difficult because the manual effort and lack of capability to handle big volumes of data. This paper will explore how companies can use econometric / statistical/ Machine Learning to forecast future demand using programming language like R. The paper compares various timeseries econometric / statistical/ Machine Learning algorithms on a global retailer data to predict future demand by 97% accuracy.

Keywords: Supply Chain, Analytics, Manufacturing, Machine Learning , R, Research, Research paper

Word count: 183

Impact of Machine Learning on Demand Forecasting In Supply Chains: A Case Study on Global Manufacturing Company

Businesses today reach far wider customer base than the periphery of their headquarters and the towns/zip-codes they operate on. Brick and mortar locations today are being supplemented by their digital stores reaching very wide geographies. Multi-channel distribution has become the go-to strategy of nearly every business in modern retail (Simchi-Levi and Wu (2018)).

With the proliferation of internet services to majority of households, widely accessible web-development services for businesses, payment solutions that easily integrate to e-commerce systems, widely adopted and secure technological infrastructure, companies have been operating on a technologically advanced system . Along with structural and operational complexities, companies also operate in a very competitive environment in today's markets. Customer expectations also are ever changing with short selling seasons and many products competing in the same markets (Minelli, Chambers, and Dhiraj (2013)). These companies are already lean so there is little room to cut cost (Ho, Kaufmann, and Liang (2017)). With this operating model, supply chain departments are crunched between meeting sales, avoiding stock outages and minimizing overages in their inventory. Hence, designing a robust and optimized Supply Chain system is a top priority of every modern company .

Literature Review

Traditionally Supply chain is defined as a network between company and its suppliers to produce and distribute a specific product to the final buyer (Beamon (1999)). However, in the modern technological business world, supply chain means a lot more. Modern supply chain involves different activities in the process, people, entities, information and resources (Strandhagen, Ola, and Reed(2017)). Supply chain deals with lots of parameters like demand of products, promotion schedules, supply schedule from vendors, logistics, warehouse optimization, cost optimization, Supply & Operations (SNOP) optimization and making sure product lands in the warehouse at the time of fulfillment. Supply chain management then means optimizing the whole process in a way to lower cost and increase operating efficiency while improving customer service and revenue in tandem . Making sure all these process above happens in the seamless fashion is hard to track in spreadsheets anymore.

According to Souza (2014), as part of digital transformation, companies are widely adopting Enterprise Resource Planning systems(ERP) that manage the workflow of business electronically. As these ERP systems are being implemented, companies are generating lots of data. According to Nagar et al. (2021), with the adaptation of advanced technology, manufacturing as well as retail have gone through a digital transformation in the events of procurement, manufacturing/assembly and distribution of goods. This digital transformation known as Industry 4.0 which generally means all technology systems are connected and it is easy to mimic actual flow of goods with information flow. This data has helped business to track their processes in a seamless fashion with IT systems running reporting in an automated fashion. It has helped operations greatly to get a good track of where things are mainly looking at data in the rear-view mirror, however, tools looking into the future for predictive and prescriptive analysis seems to be lacking. Academia and practitioners agree that this flood of data creates new opportunities, therefore many organization have been trying to develop and enhance their analytics capability for descriptive, predictive and prescriptive functions to uncover and gain a better understanding from their big data hidden values.(Boone, Ganeshan, Jain, and Sanders (2019)).

Analytics is the way to uncover hidden insights from raw data.Because of the way data is stored in ERP systems, data cannot be patched togther and drawn insights from in the way it is found. According to Russom and others (2011), analytics includes data extraction, cleansing, transformation, data mining, statistical analysis, predictive analysis to uncover hidden patterns, trends and correlations. Analytics covers integration of data in such a way that that it can be easily studied and drawn insights from. According to Mohamed-Iliasse, Loubna, and Abdelaziz (2020), digitization of business have become a revolution and machine learning is at the heart of it. It has radically transformed companies for the better in terms of sustaining competition, reducing cost of keeping inventory as well as improving cash flows. It provides new possibilities by increasing their growth and generating sustainable competitive advantages and security operations. Information flow has become just as important as real flow of goods to get actionable insights and achieve business goals (Tiwari, Wee, and Daryanto (2018)). Analytics then enables machine learning where researchers and practitioners can map all the functions digitally and optimize the pathways to achieve optimal performance by ways of various algorithms.

Some of the components of supply chain that can be generalized across majority of companies are 1) demand forecasting and planning 2) production planning 3) Planning and managing inventories 4) Planning and managing transportation. According to Malviya, Chittora, Chakrabarti, Vyas, and Poddar (2021), there are many tools for demand forecasting in supply chain or within the ERP itself but having less accuracy of predicting customer demands. Excel which is the tool of the analysts can also help but in a very small scale. Excel becomes out of scale when volume, variety and veracity of data comes into consideration. Eventually with the advent of AI system, ML lets us predict the demand and accuracy by comparing different algorithms. Unpredictable customer behavior creates problems in supply chain and creates back orders. “To overcome shortages and back-orders, industries must have to identify parts with premier probability of deficiency to operate optimally and achieve company goals” (Malviya, Chittora, Chakrabarti, Vyas, and Poddar (2021)).The ability to forecast future based on past data is a key tool to support decision making in organizations. This is achieved by a type of forecasting called time series forecasting (TSF) by looking into past data to predict future behavior (Kochak and Sharma (2015)).

In addition to traditional machine learning algorithms, Wong and Guo (2010) in their paper discuss how an hybrid intelligent model can be used to create medium term sales forecasting. The model based on neural network can be used to predict initial sales forecast that can be done using heuristic fine tuning process. Model derived this way is more accurate than traditional ARIMA models. A key concern for global manufacturers today is to reduce inventory and inventory driven costs across their supply and distribution networks. Efficient and effective management of inventory throughout the supply chain significantly improves the ultimate service provided to the customer Minimizing the total supply chain cost refers to the reduction of holding and shortage cost in the entire supply chain. Efficient inventory management is a complex process which entails the management of the inventory in the whole supply chain. The dynamic nature of the excess stock level and shortage level over all the periods is a serious issue when implementation is considered. In addition, consideration of multiple factories, multiple products leads to very complex inventory management process The complexity of the problem increases when more distribution centers and agents are involved. (Perumalsamy and Natarajan (2010))

According to Carbonneau, Laframboise, and Vahidov (2008), full collaboration is something that supply chain needs to achieve, however, because of reluctance to share information between departments and complexity of supply chain management, it is difficult to measure performance and forecast accurately in supply chain management. In some cases, department has been seen to share information and it creates a “bullwhip effect” (Lee, Padmanabhan, and Whang (1997a)) as a result of information asymetry (Thonemann (2002)). The objective of the collaborative demand planning is to include demand signals at the end of supply chain, however, it creates demand signal distortion (Lee, Padmanabhan, and Whang (1997a)) because a number of factors hinder the long term stable collaboration effort (Premkumar (2000)). Some of the factors are -

- Alignment of Business Interests
- Long term relationship management
- reluctance to share information
- complexity of large-scale supply chain management
- competence of personnel supporting supply chain management
- performance measurement and incentive systems to support supply chain management

Achieving the collaborative environment also might be restrictive because of power structure in the organizations. Finally with the advent of E- business and the technology business systems, supply chain might be totally independent on coming up with a baseline demand and use collaborative demand planning as a reference to test the baseline. According to Thonemann (2002), rather than using collaborative demand analysis, sharing advance information with various departments can improve supply chain performance. One of the ways to do this rather than manually coming up with the forecast, feeding the historical data into machine learning systems which can come up with product specific forecast. Machine learning teaches computers to learn based the historical data with explicit programming it Gately (2017). According to SAS.com these analytic models allow the supply chain analyst, managers, researchers and data scientist to produce reliable , repeatable results and uncover the hidden patterns of insights in the data.

There are three types of forecasting techniques - qualitative method, time series method and causual method. Qualitative method are then based on opinion of subject matter expert and are therefore subjective. Time series model are models to predict the future based on historical data and causual methods are based on assumptions that demand is based on certain factors and explore the corelations of those factors(Kochak and Sharma (2015)).

In the research conducted by Carbonneau, Laframboise, and Vahidov (2008), the researchers compare the results obtained by implementing machine learning with other traditional methods of supply chain planning. The researchers compare traditional methods with naive forecasting, trend, moving averages and linear regression. According to their research neural networks and support vector machines show the best performance their forecasting was not statistically significantly better than regression model. Some of the methods used in this research are the following-

- Naive Forecast - Naive forecast is one of the simplest forecast methods and often used as as baseline methods against which the performance of other method is used. It simply uses the latest value of the variable of interest as a best guess for the future value.
- Moving average- Moving average is a method that uses the average of defined number of previous periods as a forecast of future demand.
- Trend based forecasting- Trend based forecasting is based on simple linear regression that takes time as an independent variable and tries to forecast demand as a function of time.
- Multiple linear regression- Multiple linear regression is an auto regressive model that tries to predict the change in demand using past changes in demand observations as independent variables.

According to Carbonneau, Laframboise, and Vahidov (2008), more advance methods are as follows:-

- Neural networks- Neural networks possess the input later of data and it creates a layer of of neurons with associated data to give an output. In the feed forward type of neural nets the algorithm learns from the past data and generalizes the prediction.

- Recurrent neural networks- Recurrent neural networks allow output signals of some of their neurons to flow back and serve as inputs for the neurons of the same layer or those of the previous layers. RNN serve as a powerful tool for many complex problems, in particular when time series data is involved. The training method called "back-propagation through time" could be applied to train a RNN on a given training set (Werbos, 1990).

- Support vector machines- Support vector machines (SVMs) are a newer type of universal function approximators that are based on the structural risk minimization principle from statistical learning theory (Vapnik, 1995) as opposed to the empirical risk minimization principle on which neural networks and linear regression, to name a few, are based.Forecasts have traditionally become a basis of planning and executing supply chain activities and they have become critically important due to increasing customer expectations, shortening delivery times and the need to manage scarce resources.

According to Mohamed-Iliasse, Loubna, and Abdelaziz (2020), Machine learning is most suitable for supply chain function. As this area is being flooded with data, the adoption of machine learning can help make supply chain automated with very minimal human intervention. As per Dejonckheere, Disney, Lambrecht, and Towill (2003), although various algorithms can predict the future, selecting the right algorithm for right kind of data may be necessary as various algorithms have their strength and weakness.

Hypothesis. Applying machine learning methods in supply chain can improve forecast accuracy to keep optimal inventory levels.While there were many theoretical analysis on how Analytics can help supply chain management of small size companies there was gap on using data to test the hypothesis, I also want to test which algorithm can predict the future with best accuracy through my research.

Methods

The data for this study was derived from Contoso corporation which is a multinational business headquartered in Paris. The company is in the business of manufacturing, sales and support with more than 100,000 products. The database of this fictitious company was publicly published by Microsoft on January 21,2010 to demonstrate their data related offerings. This dataset includes C-level, sales/marketing, IT and common finance scenarios of the retail industries. I downloaded this database from https://www.microsoft.com/en-us/download/details.aspx?id=18279 and restored into a local instance of my SQL Server relational database. The database mimics the Business Intelligence star schema platform which is common in most retail companies which is the primary data source for most Supply Chain and Demand Planning Analytics. The research that I plan to conduct with this data is quantitative to predict quantity sales by month. In this case study, I plan to carry out statistical and forecasting modeling using mean forecast model, moving averages model, seasonal naive model, linear regression, exponential smoothing model, Holt-winters model, ARIMA (Auto Regressive Integrated Moving Average , neural networks and Facebook Prophet in R programming language.

Participants

For the purpose of this research, I have narrowed the focus to Contoso Ltd. although the database consisted of eleven brands The unit of study for this research is Supply chain using machine learning to predict forecast accuracy. The participant of this research is a fictitious company Contoso Ltd. Agrregated 710 products will be part of the analysis which are produced by Contoso. This is a case study which studies each member(SKU) contained in the data.All other products related to other Brands from the database will be excluded. The data is purely a time series data and doesnot contain any external regressors.

Procedure

After I restored the database to the local instance of SQL Server, I queried the table using SQL (Structured query Language) to derive the relevant data related to Contoso for my study. The SQL query is outline below-

“SELECT cast(DATEADD(month, DATEDIFF(month, 0, [DateKey]), 0) as date) AS YearMonth,s.[StoreKey],s.[ProductKey],sum([SalesQuantity]-[ReturnQuantity]) QtySold,sum([SalesAmount]-[ReturnAmount]) RevenueSold FROM [ContosoRetailDW].[dbo].[FactOnlineSales] s join [dbo].[DimStore] st on s.StoreKey=st.StoreKey join [dbo].[DimProduct] p on p.ProductKey=s.ProductKey where st.Status=‘on’ and BrandName=‘Contoso’ group by DATEADD(month, DATEDIFF(month, 0, [DateKey]), 0) ,s.[StoreKey] ,s.[ProductKey] order by 1,2,3”

Once the data was load, data type of the variables were changed. Productkey and storekey wer changed to character while YearMonth was changed to date datatype. Qtysold and Revenue were changed to numerical .

The data in the database spans from 1st of January 2007 till December 31 2010. Before approaching bottom up forecasting strategies, I summarized the data by month so high level understanding of the data can be achieved. The transformed data was changed to time series with ts function with frequency value of 12. For the purpose of this study, data was grouped by month and the corresponding quantity sold in that month was calculated. Once data was studied at the high level and was determined that it was not the case of white noise, further process of analysis were undertaken. Multiple types of algorithms were run through the data to study which model fitted the data best. Forecast accuracy was studied.

In this case study I plan to split the data into training data and test data. Data from January of 2007 until July of 2009 was part of the testing data while data past July of 2009 until the end of 2009 was put in testing dataset. In this case study, I have omitted any closed stores from the data so I do not have to plan predict forecast for that store. The response variables on this data is quantity of UnitSales while time is the independent variables. This makes it a time series forecasting case study.

Measures

This study was conducted in R with the use of data extraction , data transformation, visualization and forecasting libraries. In this research I used DBI and RODBC packages to connect and query SQL server databases. Dplyr, dbpylr, data. table and tidyr were the libraries used for data manipulation. Lubridate package was used for date transformation and forecast, zoo, fpp2, prophet and neuralnet libraries were used for econometric statistical analysis, forecasting and prediction.

In terms of data it consisted of yearmonth of transaction, a date field while other variables used were storekey, productkey, QuantitySold and Revenue. Storekey, ProductKey, Quantitysold and Revenue were numerical fields.

Lots of approaches mentioned by Robert Hyndaman (Hyndman and Khandakar (2008)) are adopted in my case study as part of this research. In this case study YearMonth is the independent variable while Qty sold is a dependent variable or the measure to forecast into the future.

Analysis

To analyze the data, I used the zoo, fpp2 and forecast package in R. I started my analysis with a basic understanding of aggregated data. A basic timeplot was graphed with time in the horizontal axis and quantity in y- axis. The data showed clearly a positive trend in the data. Another plot was created to see the seasonality of the data. The timeseries plot clearly showed seasonality in the graph in regular intervals within the year. Whitenoise was tested on the data if the full data has some trend and time series could be used for forecasting. Ggacf function was used for this purpose. Some data points were displayed outside of the 95 % confidence interval which proved that this data was suitable for time series analysis. Upon creating subseries plot so data would be spread-out over months, it showed that data behaved differently across various months. There was an indication that Quantity sales was lower in first three months of the year while and the sales peaked in summer and December. There also seems seasonality on the data which will be tested by seasonal plot. From the seasonal plot in fig 4, we can see that business for Contoso peaks in April, May, July and December. Business starts slow in the first three months of the year and is tapers down in June and October across all years. Data was de-trended which allowed me to study seasonality and variance more clearly. Upon studying the trend strength was 0.95 and seasonal strength was 0.68 on the data.

Next step was to use some benchmark methods and get some residuals to compare with various machine learning algorithm.

Results

As stated in the objective of this paper, I fitted various time series models with my data and results are the following-

Mean Forecast Model- The most basic method of forecasting was fitted on the training data. The model does not seem to fit it well. The model missed the trend, seasonality and variation in the data. Fitted Model Mean Average Error was 22264.53 while the Mean Percentage Error was-4.82%.Upon running this fitted model to the test data, MAE was recoreded at 59928.52 and MPE at 33.14%. The training residual was higher than desired and a flat forecast was outputted.

Central Moving Average Model- The central moving average model was fitted to the data. Due to the seasonality and trend present in the data , it was not a optimum model for this data. The model missed the variations on the data as shown in the plots. To confirm what I found visually, a Cox-Start test was performed on the central moving average model. It showed significant trend with p value of 3.814697e-06. Seasonal Boxplot and seasonal distribution supported the seasonality.

Seasonal Naive Method- Seasonal Naive method was applied on the data as seasonality was seen to be present in the training data. It fitted the seasonal data better than the previous two models with MAE of 29504.37 and MPE of 17.58%. However when tested for accuracy on the testing data, the testing data showed MAE of 64096 and MPE of 35.66%. Upon running the model through ACF test for auto-correlation, it seemed to miss lots of signals on the data. The residuals clearly showed signals missed by the fitted model. This model is not a good model to predict the data.

Exponential smoothing model- The training data was also run through an exponential smoothing model after differencing the data for stationary. The algorithm picked ETS(M,N,N) model as the best fitted model for the data. The sigma(residual) of fit was 0.1137 with AIC of 696 and BIC at 701. This model fits lot better than seasonal naive method. The model presented MAE of 10448 and MAPE of 2.389% which further aligned us towards a better model. Upon running the results through the Ljung- Box test, p value of 0.4666 which was higher than desired. When running this model through forecast function for the next 5 months, it yielded a constant forecast which was less ideal than desired. This was a good model fitted but not versatile enough to calculate the seasonal forecast. The testing dataset resulted into MAE of 20770 and MPE of -10.90%.

Holt Winters Method- As a result of non seasonal methods not working so far and their limitation, Holt winters algorithm was applied on this data. The model fitted the data well. The training residual(sigma) calculated was 0.1219 which is very similar to the ETS model above with AIC of 709 and BIC of 734. Ljung - Box test was run and it presented p value of 0.0009679 which is significant and no significant auto-corelations were present on the residuals. I concluded this being the best model so far. Upon extrapolating the forecast on the model on test data this model provided considerably good prediction with MAE of 8169 and MPE of -2.85%.

Seasonal ARIMA method- Seasonal ARIMA was run on the data. The model fitted reasonably well with MAE of 4600 and training MPE of 1.90%. This is the best fit on the training model. Upon testing the model on the testing data, it yielded a MAE of 13772 and MPE of -7.78%. The residual showed some data patterns left by the model as per ACF chart.

Facebook Prophet- Prophet algorithm was run on the data. The model fitted reasonably well with MAE of 12059.73 and training MPE of -6.58%.Upon the initial testing on the testing dataset, it yielded a MAE of 12059.73 and MPE of -6.582227% which is better than ARIMA but not as good as Holt Winters. The cross validation function was used which performed better with MAE of 9399.411 and MPE of 2.216871%. This was concluded the best model so far .

Neural Network- Neural Network was applied on the data. Testing MAE of 6818.629 and testing MPE of -3.748293% was obtained as a result. This is the third best model after Facebook prophet and Holt winters.

Linear regression model was not run because of the seasonality in the data. A machine learning method called AutoCatBoostCARMA method was tried on the data but it was beyond the hardware limitation of my laptop. The model would run for 5 hours but it was beyond the capability of my laptop and it would crash my laptop and eventually freeze and shut down. An external GPU was required to run deep learning models which I didn't carry at the time of this research.

The result was concluded by being able to predict the demand by month of products with 98% accuracy on a aggregate level with the help of Prophet, Holt Winters and Neural Network Methods.

Discussion

From the result section we can infer that an algorithm that can handle a seasonal data is required to properly fit a seasonal data. Upon running the mean, naive and exponential smoothing data to the model, it was not able to properly capture the trend, variability and monthly fluctuation in the data. In the other hand, the methods like Prophet, Holt Winters and Neural networks were able to properly capture the seasonality in the data and fit the data and predict with better accuracy.

Based on the results and their statistics I would rank prophet as the best model that could accurately predict the test data with ~ 98 % accuracy. That was possible because of the cross-validation method built into the model. The model would go back and calculate cross validate error term from the forecast value and adjust accordingly. This feature improved the model by 4% to come to a top rank. It was not the easiest algorithm to implement but it was the most powerful. Holt winters on the other hand was surprisingly better model to rank as the second-best model with accuracy close to 97%. It was easy to implement, and output was simple enough to comprehend the results. One of the drawbacks of this model was it did not allow the external regressors to be factored in the model. In my case I didn't have any external regressors, so it was an optimal model and hence the good performance. Neural Networks was the third best model. It is a good model and easy to use. It was robust to outliers. It allowed continuous regressors which is a positive aspect of the algorithm. It is a black box, and it provides low insights but results. It is very simple to use but has low flexibility; some also refer this to as black box because of the nature of the algorithm. Sarimax which is the seasonal counterpart of ARIMA was simple to implement. The automated optimization built into the model made it easy to implement which would calculate the seasonal, trend and integrated parameters. One of the drawback was it does not allow continuous regressors.

While implementing and testing the performances of each model it was found that each model has some advantages and disadvantages, and it is to supply chain experts to know when to implement the correct model. This is driven by the data and behavior such as trend, seasonality, and correlations on the data. It was found that some models such as ARIMA work with the differencing data where trend is removed while others like prophet and neural networks have internal mechanism to difference the data around zero if needed. In some cases, the pattern in the data could be a random walk and forecasting cannot be achieved via econometric or machine learning methods above. If the data exhibits random walk or not should be initially studied in the initial stages of exploratory data analysis of time series.

The work conducted here can solve the deficiency in existing ERP system as stated by Souza (2014) in their paper. While the ERP system automate the descriptive reporting on actual data about the business, machine learning and forecasting can utilize the existing data to extrapolate future demand based on that data in similar environments. Forecasters can also overlay some external regressors like holidays and promotions to get better forecasting results. As stated by Mohamed-Iliasse, Loubna, and Abdelaziz (2020) in their research, this capability has only been possible because the digitization of companies which have unlocked the possibilities of machine learning and digital forecasting. As stated by Tiwari, Wee, and Daryanto (2018), as companies plan better for the future with good accuracy, resources are not wasted, and customer satisfaction is increased which results in companies improving market share.

As stated by Lee, Padmanabhan, and Whang (1997b), in the previous studies conducted, bullwhip effect is a result of information bais between the sales and supply chain function in the company. My research confirms it that data machine learning algorithms can be used to forecast the demand of products. Past historical data can act as a goldmine for companies. Rather than wholly relying on the sales departments in companies who tend to be overly optimistic than realistic in their estimates , a machine learning forecast can act as a perfect baseline where further alignment can be achieved across the organization. From my previous knowledge of most of the Enterprise Resource Planning(ERP) systems, these complex algorithms might be part of ERPs. It is to the supply chain departments to employ Analytics professionals to give them insights of the analytic capabilities. Programming, analytic and mathematical skill-sets will be required to employ such algorithms and maximize the results as it involves proving the appropriate parameters to the algorithms.

Conclusion

This research confirms my hypothesis that machine learning can be used to predict future demand. My research proves the theory pushed by Carbonneau, Laframboise, and Vahidov (2008) that advance machine learning algorithms can be used to predict demand with higher accuracy than simple methods.Facebook's prophet algorithm predicted the forecast on test data by 97% accuracy. One of the caveats of my current research was the shipment data was used as the true demand to test the accuracy of models. In majority of companies, the shipment data accurately does not capture the true demand, there are also shipments waiting for products to be available that shifts the true demand into next months. There might also be cases of lost sales where shipment of customer demand was never possible within the service level agreement required by the customer. Also to further make the model more robust, past promotion data can ba fed into the model where possible.

Future work can be done to implement modern machine learning algorithms on supply chain data. Machine learning algorithms such as XGBoost, RNN, AutoCatBoostCARMA and deep learning methods that are sophisticated and but needing higher hardware configuration can be tried upon to see if the performance can further improved. Also, the addition of other external regressors and market indicators such as promotions, weather, economic recessions and holidays can be further incorporated into the research.

References

Beamon, B. M. (1999). Designing the green supply chain. *Logistics Information Management*.

Boone, T., Ganeshan, R., Jain, A., & Sanders, N. R. (2019). Forecasting sales in the supply chain: Consumer analytics in the big data era. *International Journal of Forecasting*, *35*(1), 170–180. https://doi.org/https://doi.org/10.1016/j.ijforecast.2018.09.003

Carbonneau, R., Laframboise, K., & Vahidov, R. (2008). Application of machine learning techniques for supply chain demand forecasting. *European Journal of Operational Research*, *184*(3), 1140–1154. https://doi.org/https://doi.org/10.1016/j.ejor.2006.12.004

Dejonckheere, J., Disney, S. M., Lambrecht, M., & Towill, D. R. (2003). The dynamics of aggregate planning. *Production Planning & Control*, *14*(6), 497–516. https://doi.org/10.1080/09537280310001621967

Gately, C. (2017). Vekia: Pioneering machine learning in retail supply chain. *Small Enterprise Research*, *24*(3), 326–332. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=bth&AN=126591391&site=eds-live&scope=site&custid=s4786267

Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, *26*(3), 1–22. Retrieved from https://www.jstatsoft.org/article/view/v027i03

Kochak, A., & Sharma, S. (2015). Demand forecasting using neural network for supply chain management. *International Journal of Mechanical Engineering and Robotics Research*, *4*(1), 96–104.

Lee, H. L., Padmanabhan, V., & Whang, S. (1997a). Information distortion in a supply chain: The bullwhip effect. *Management Science*, *43*(4), 546–558.

Lee, H. L., Padmanabhan, V., & Whang, S. (1997b). The bullwhip effect in supply chains. *Sloan Management Review*, *38*, 93–102.

Malviya, L., Chittora, P., Chakrabarti, P., Vyas, R. S., & Poddar, S. (2021). Backorder prediction in the supply chain using machine learning. *Materials Today: Proceedings.* https://doi.org/https://doi.org/10.1016/j.matpr.2020.11.558

Minelli, M., Chambers, M., & Dhiraj, A. (2013). *Big data, big analytics: Emerging business intelligence and analytic trends for today's businesses* (Vol. 578). John Wiley & Sons.

Mohamed-Iliasse, M., Loubna, B., & Abdelaziz, B. (2020). Is machine learning revolutionizing supply chain? In *2020 5th international conference on logistics operations management (GOL)* (pp. 1–10). https://doi.org/10.1109/GOL49479.2020.9314713

Nagar, D., Raghav, S., Bhardwaj, A., Kumar, R., Lata Singh, P., & Sindhwani, R. (2021). Machine learning: Best way to sustain the supply chain in the era of industry 4.0. *Materials Today: Proceedings.* https://doi.org/https://doi.org/10.1016/j.matpr.2021.01.267

Perumalsamy, R., & Natarajan, J. (2010). Predictive analytics using genetic algorithm for efficient supply chain inventory optimization. In *2010 second international conference on computing, communication and networking technologies* (pp. 1–8). https://doi.org/10.1109/ICCCNT.2010.5591607

Premkumar, G. P. (2000). Interorganization systems and supply chain management. *Information Systems Management*, *17*(3), 1–14.

Russom, P., & others. (2011). Big data analytics. *TDWI Best Practices Report,* *Fourth Quarter*, *19*(4), 1–34.

Simchi-Levi, D., & Wu, M. X. (2018). Powering retailers' digitization through analytics and automation. *International Journal of Production Research*, *56*(1/2), 809–816. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=edb&AN=128748384&site=eds-live&scope=site&custid=s4786267

Souza, G. C. (2014). Supply chain analytics. *Business Horizons*, *57*(5), 595–605.

Thonemann, U. W. (2002). Improving supply-chain performance by sharing advance demand information. *European Journal of Operational Research*, *142*(1), 81–107. https://doi.org/https://doi.org/10.1016/S0377-2217(01)00281-8

Tiwari, S., Wee, H. M., & Daryanto, Y. (2018). Big data analytics in supply chain management between 2010 and 2016: Insights to industries. *Computers & Industrial Engineering*, *115*, 319–330. https://doi.org/https://doi.org/10.1016/j.cie.2017.11.017

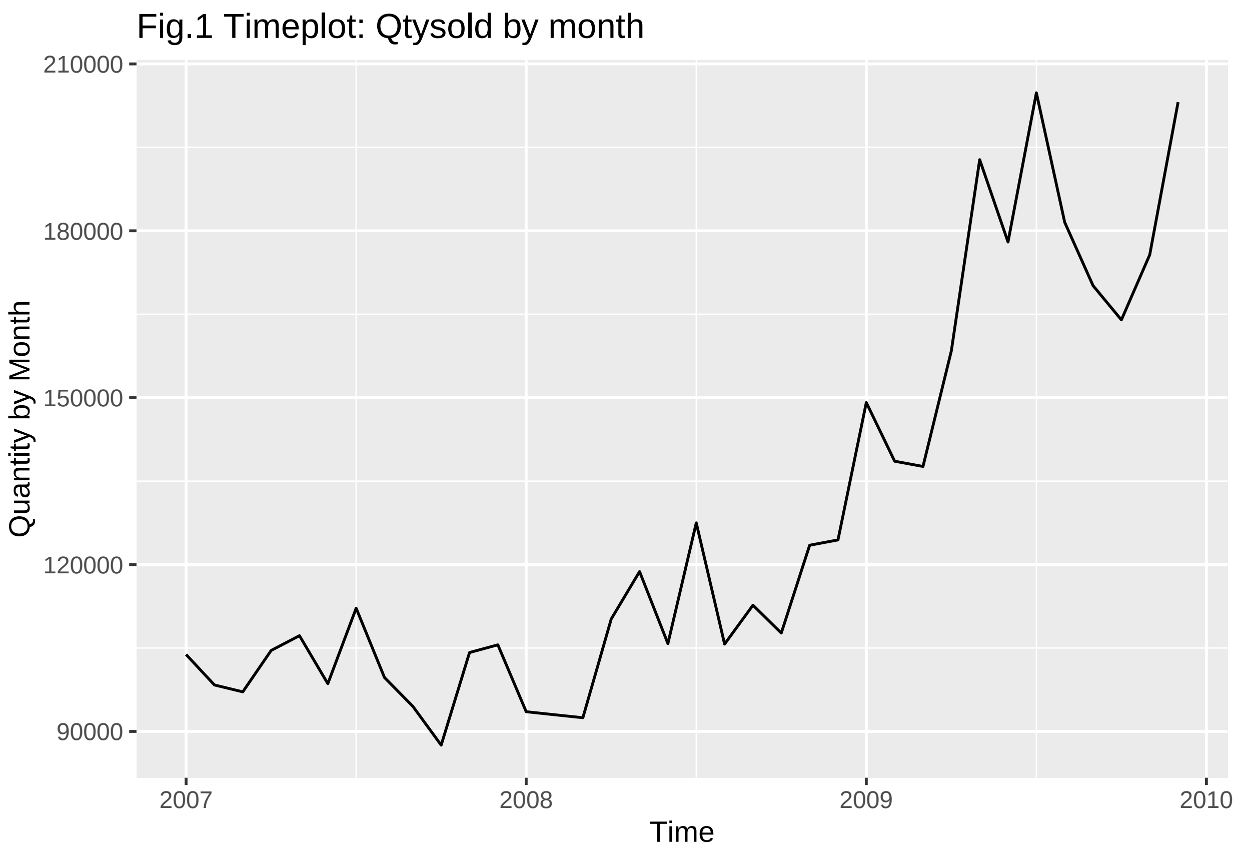
Wong, W. K., & Guo, Z. X. (2010). A hybrid intelligent model for medium-term sales forecasting in fashion retail supply chains using extreme learning machine and harmony search algorithm. *International Journal of Production Economics*, *128*(2), 614–624. https://doi.org/https://doi.org/10.1016/j.ijpe.2010.07.008

Appendix

```
#pull data from SQL Server
connStr <- "Driver=SQL Server;Server=localhost;Database=ContosoRetailDW;trusted_connecti
conn <- odbcDriverConnect(connStr)

dtmain <- sqlQuery(conn, " SELECT cast(DATEADD(month, DATEDIFF(month, 0, [DateKey]), 0)
                   RevenueSold FROM [ContosoRetailDW].[dbo].[FactOnlineSales] s join [db
                   on p.ProductKey=s.ProductKey where st.Status='on' and BrandName='Cont

#head(dtmain) # Gettign a glimpse of the data

# changing datatype


dtmain$ProductKey<-as.character(dtmain$ProductKey)
dtmain$StoreKey<-as.character(dtmain$StoreKey)
dtmain$YearMonth<-as.Date(dtmain$YearMonth)
dtmain$QtySold<-as.numeric(dtmain$QtySold)

# str(dtmain) # Saw that the year month is date formatted while Storekey and Productke
#head(dtmain)
#Before starting to forecast on each product level, I want to aggregate this data at t

Qtysoldbydate_df<- dtmain %>% group_by(YearMonth)%>% summarise(TotalQty=sum(QtySold))
#head(Qtysoldbydate_df) # this provided the total quantity sold by month
#summary(Qtysoldbydate_df)
```

```
# inserting this data in timeseries


Qtysoldbydate<- ts(Qtysoldbydate_df$TotalQty, start = c(2007,01), frequency = 12)
```

```
autoplot(Qtysoldbydate)+ ggtitle("Fig.1 Timeplot: Qtysold by month")+ ylab("Quantity by
```



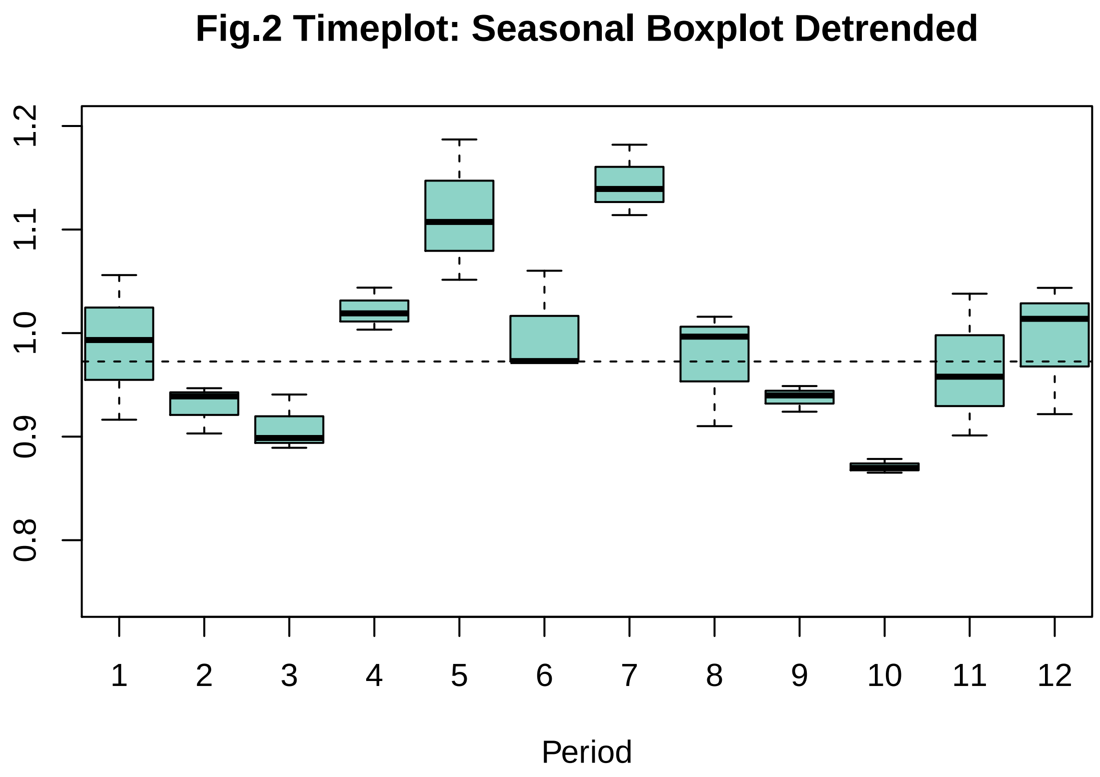
```
# from the chart in Fig 1, there seems to be some trend in the data as well as seasona
```

```
#Alternatively lets look at the data with Central moving average
# THIS FUNCTION CAN TAKE IN VECTORS OR TS OBJECTS
cma <- cmav(Qtysoldbydate, outplot=1)
print(cma)


# Test COX STUART on the cma above to test if there is a significant trend
```

```
coxstuart(cma)

# Output shows significatnt trend>0.05
```

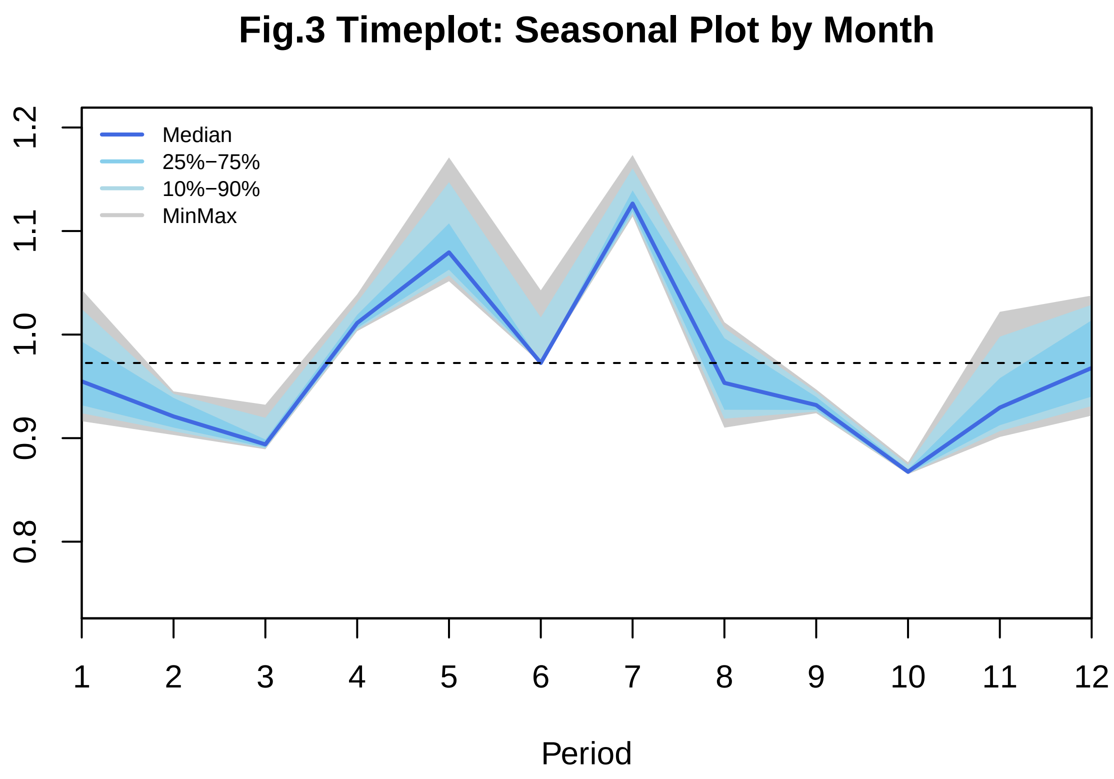
```
#Lets test for seasonality

seasplot(Qtysoldbydate,  main="Fig.2 Timeplot: Seasonal Plot by Month")


# Evidence of seasonlity= true
```

```
#Lets plot seasonlity with seasonal box plot

seasplot(Qtysoldbydate, outplot = 2,main="Fig.2 Timeplot: Seasonal Boxplot Detrended" )
```



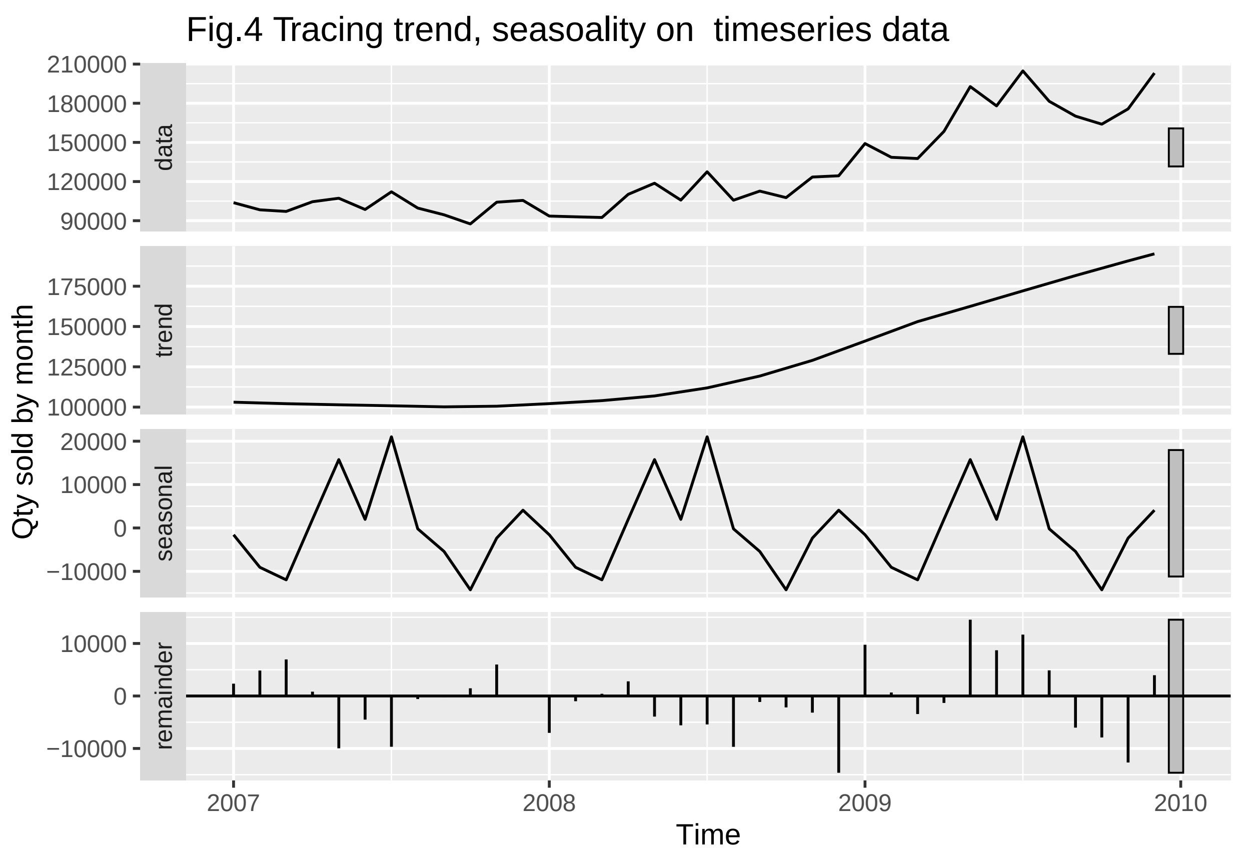
```
## Results of statistical testing

## Evidence of trend: TRUE  (pval: 0)

## Evidence of seasonality: TRUE  (pval: 0.005)
```

```
seasplot(Qtysoldbydate, outplot = 4,main="Fig.3 Timeplot: Seasonal Plot by Month") # sea
```



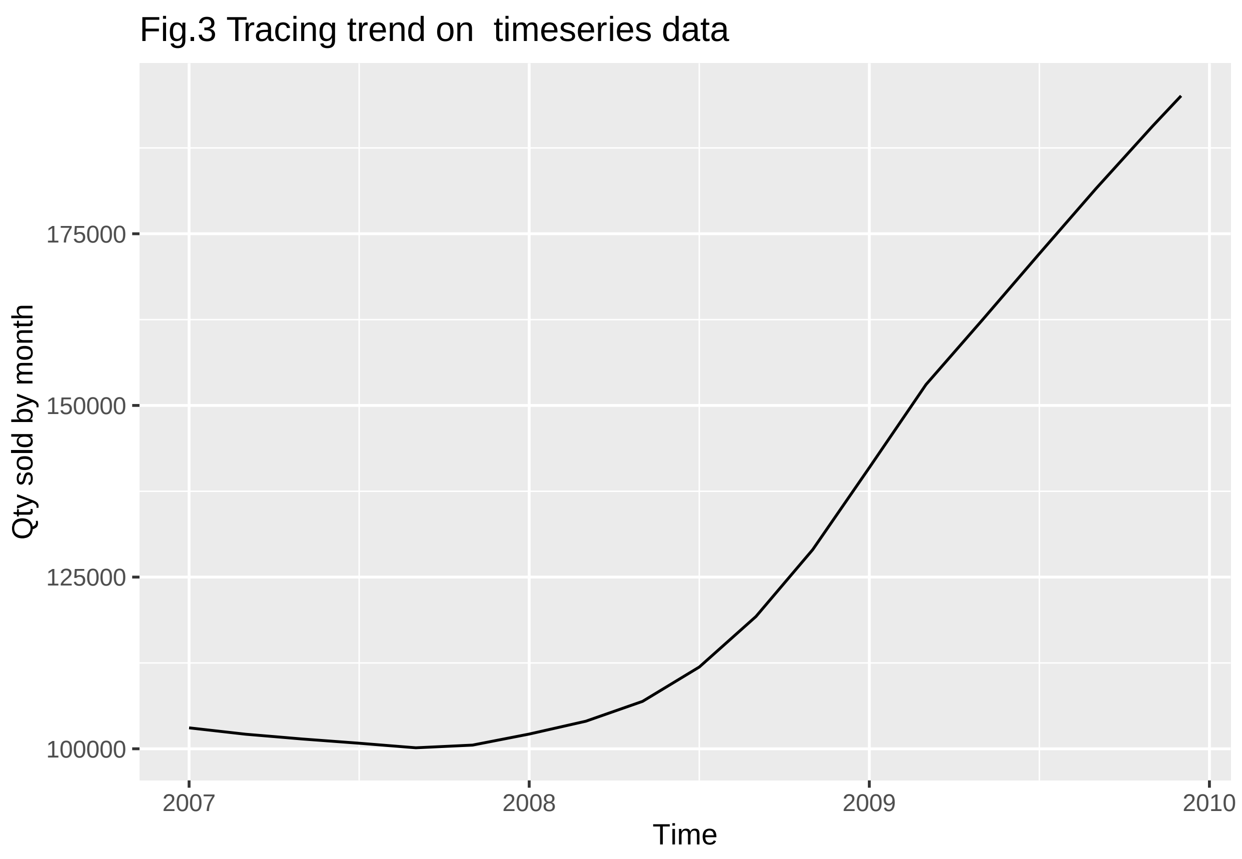
```
## Results of statistical testing

## Evidence of trend: TRUE  (pval: 0)

## Evidence of seasonality: TRUE  (pval: 0.005)
```

```
## get decompostion of data

# The idea of decompostion is we decompose the data into constituents parts so we fore

decomp<-Qtysoldbydate%>% stl(s.window = "periodic") # the values of trend, seasonality

autoplot(decomp)+  ggtitle("Fig.4 Tracing trend, seasoality on  timeseries data ")+ylab(
```

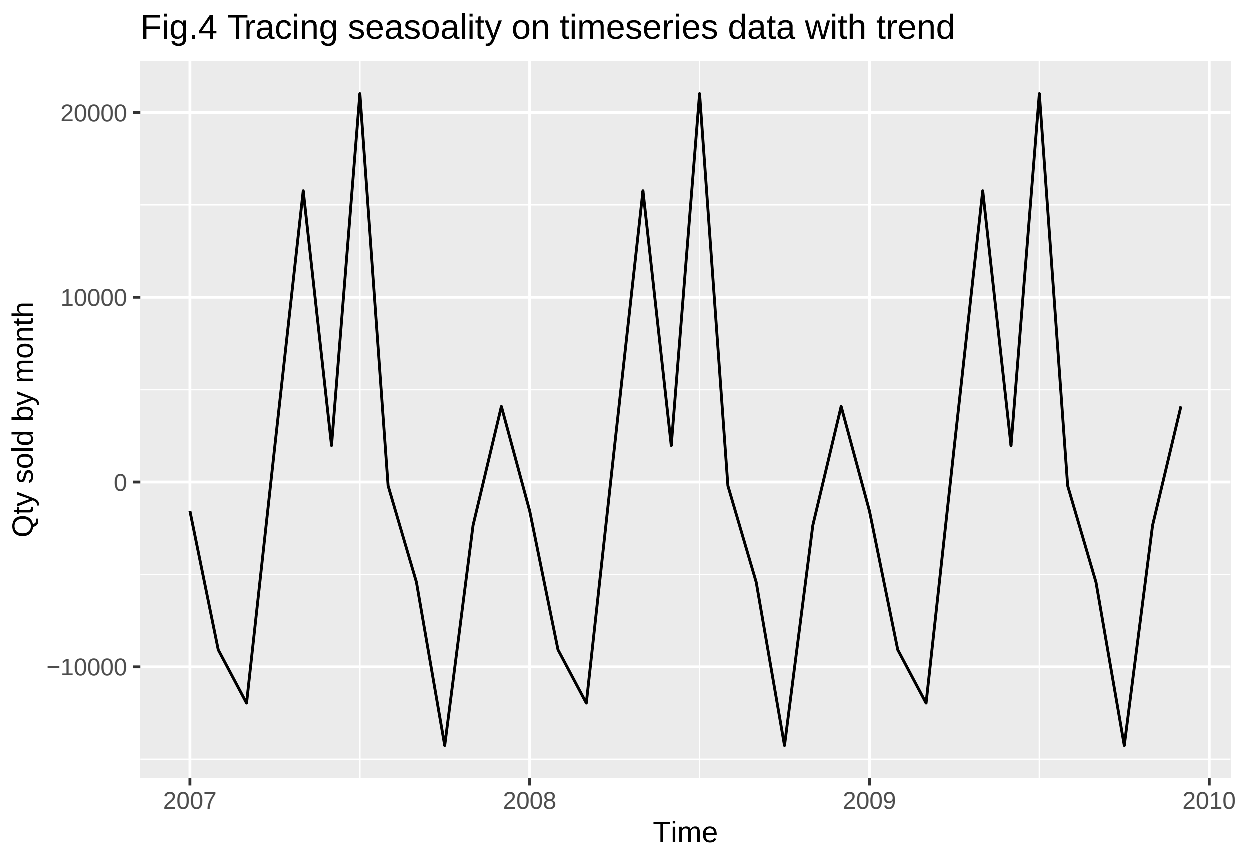


```
# From the chart above there seems a possbibility of seasonality in the data and shows
```

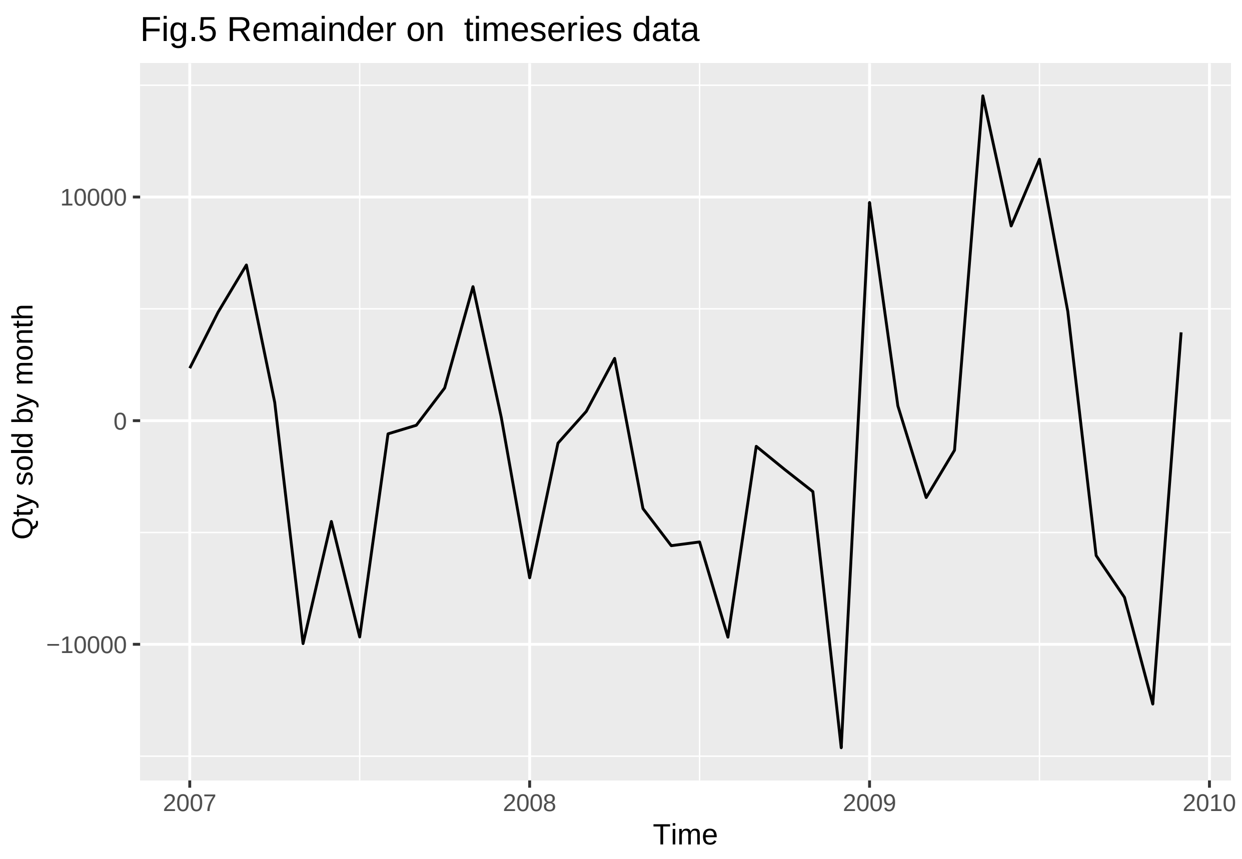
```
Trendseries<-trendcycle(decomp) # ger trend cycle independently


autoplot(Trendseries) + ggtitle("Fig.3 Tracing trend on  timeseries data ")+ylab("Qty so
```



```
Seasonal<-seasonal(decomp)


autoplot(Seasonal)+  ggtitle("Fig.4 Tracing seasoality on timeseries data with trend ")+
```

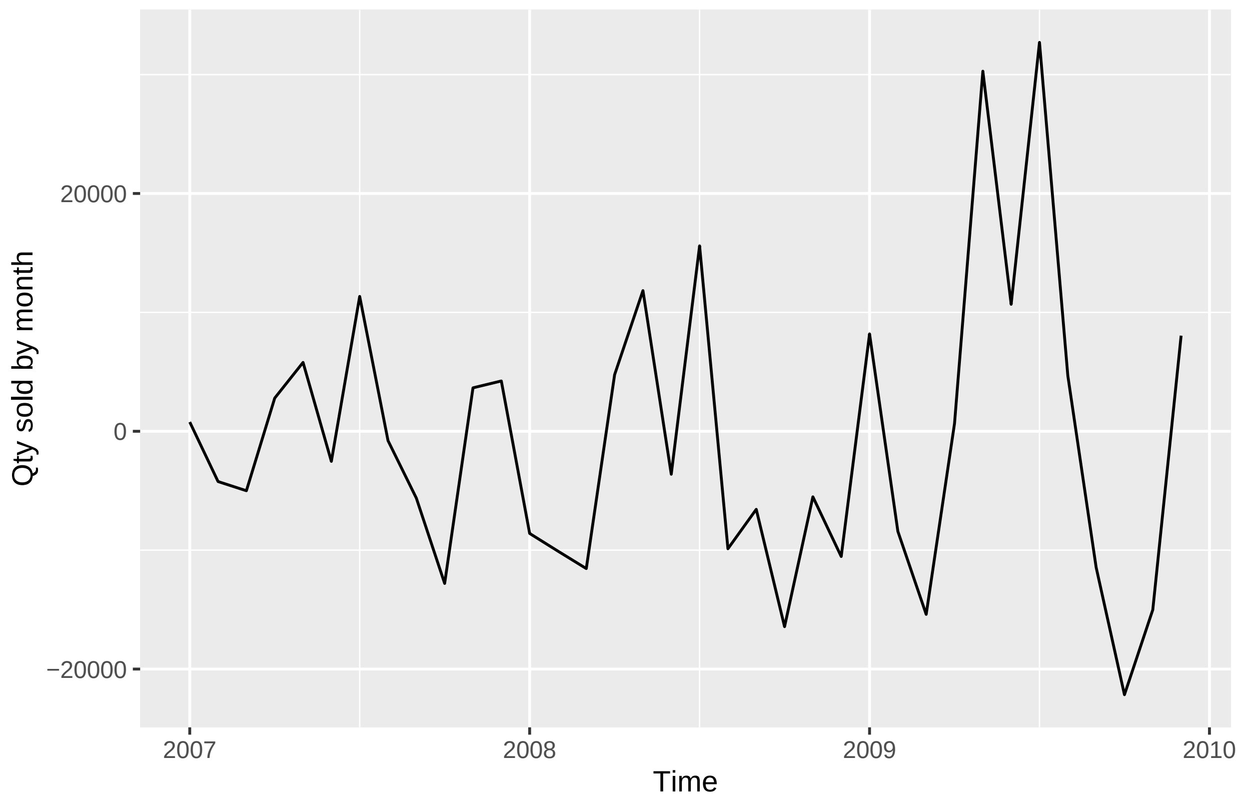


```
# Decomposing the data shows seasonal component as well but we will confirm it once we


## remainder
## this is what you get when you take out trend and seasonality on this price or promo
rem<-remainder(decomp)
autoplot(rem) +  ggtitle("Fig.5 Remainder on  timeseries data ")+ylab("Qty sold by month
```

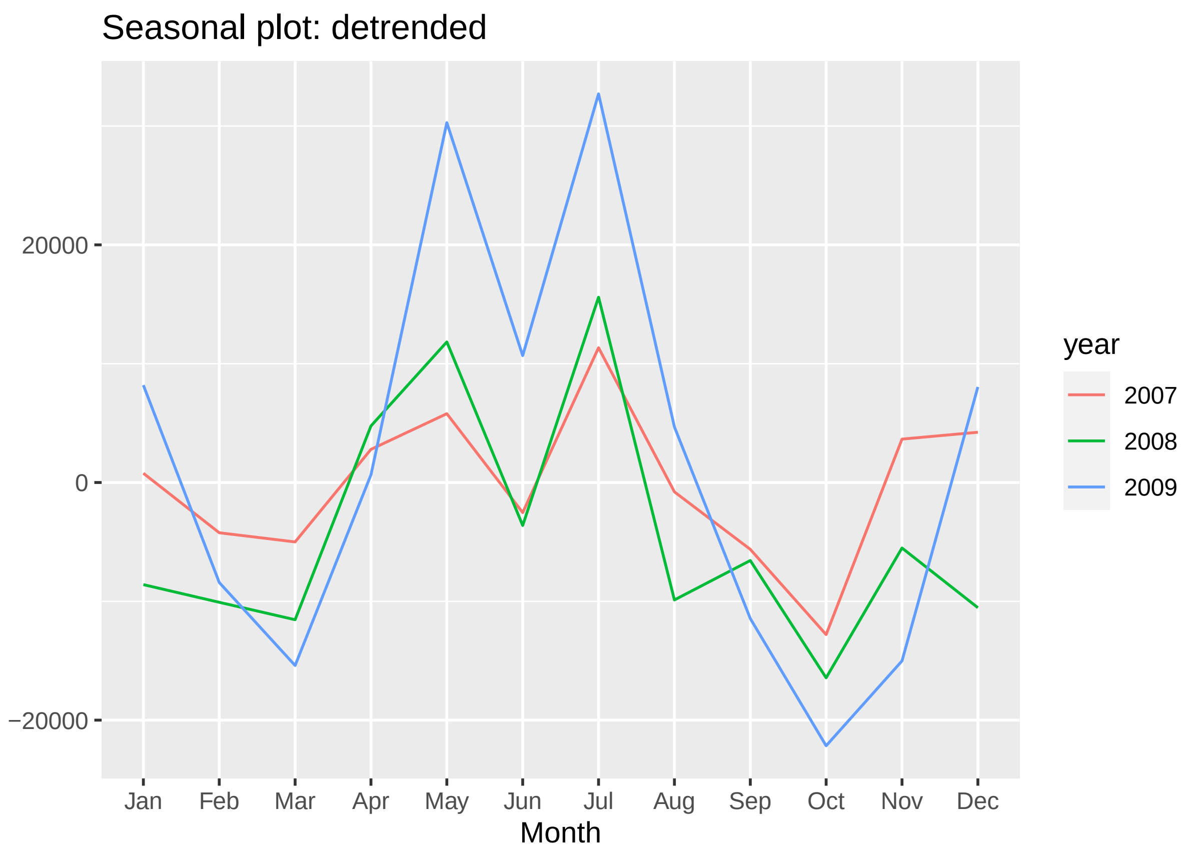


```
# Another way to look at this is to see the variation in months with their mean
#Lets look at it further with subseris plot

#### Alternate way to test the trend strenth and other strengths as above
# Alternatively another way to detrend the data is to add seasonality and remainder fr

detrended<-Seasonal+rem
autoplot(detrended)+  ggtitle("Fig.7 Remainder and sesoanlity with trend removed on  tim
```

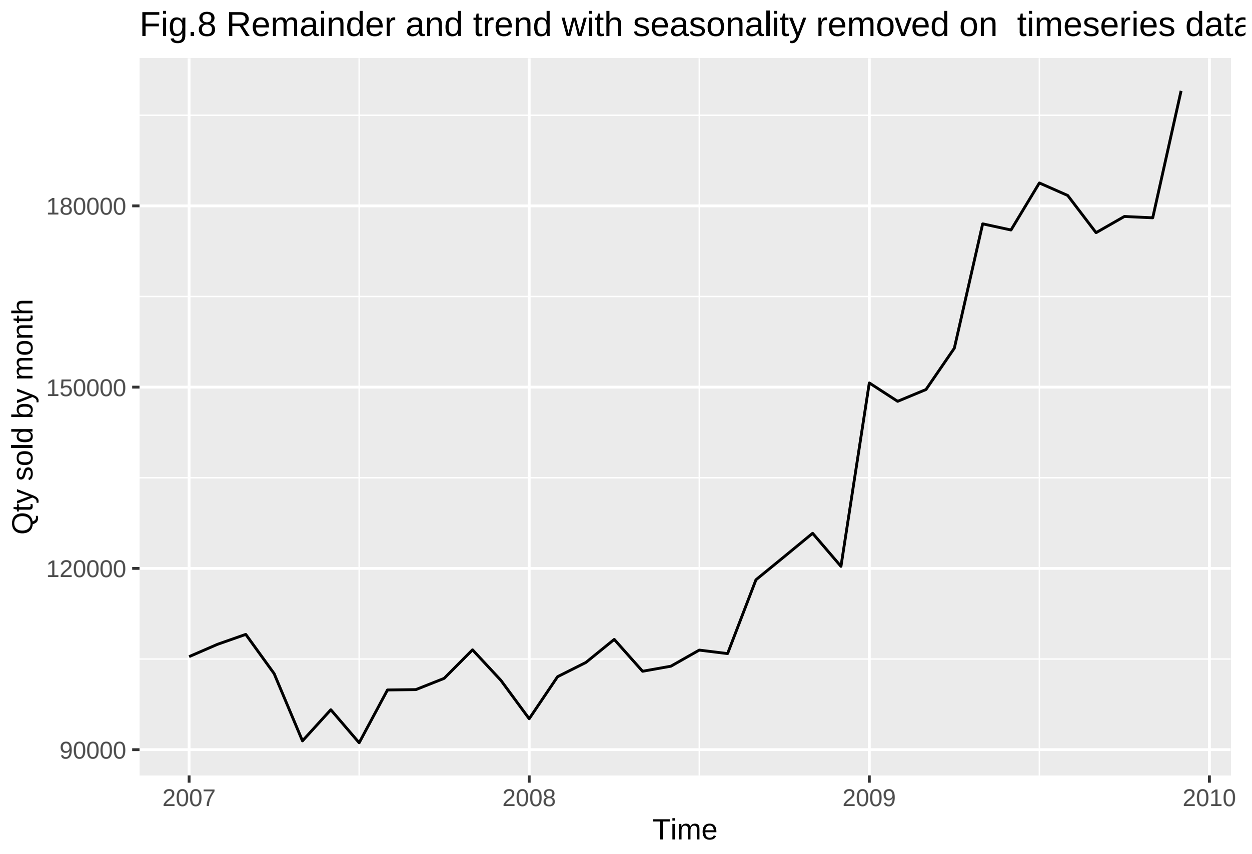
Fig.7 Remainder and sesoanlity with trend removed on timeseries data



```
ggseasonplot(detrended)
```



```
## remove the seasonality

# seasonal adjustment



deseasonalize<-rem+Trendseries

autoplot(deseasonalize)+  ggtitle("Fig.8 Remainder and trend with seasonality removed on
```



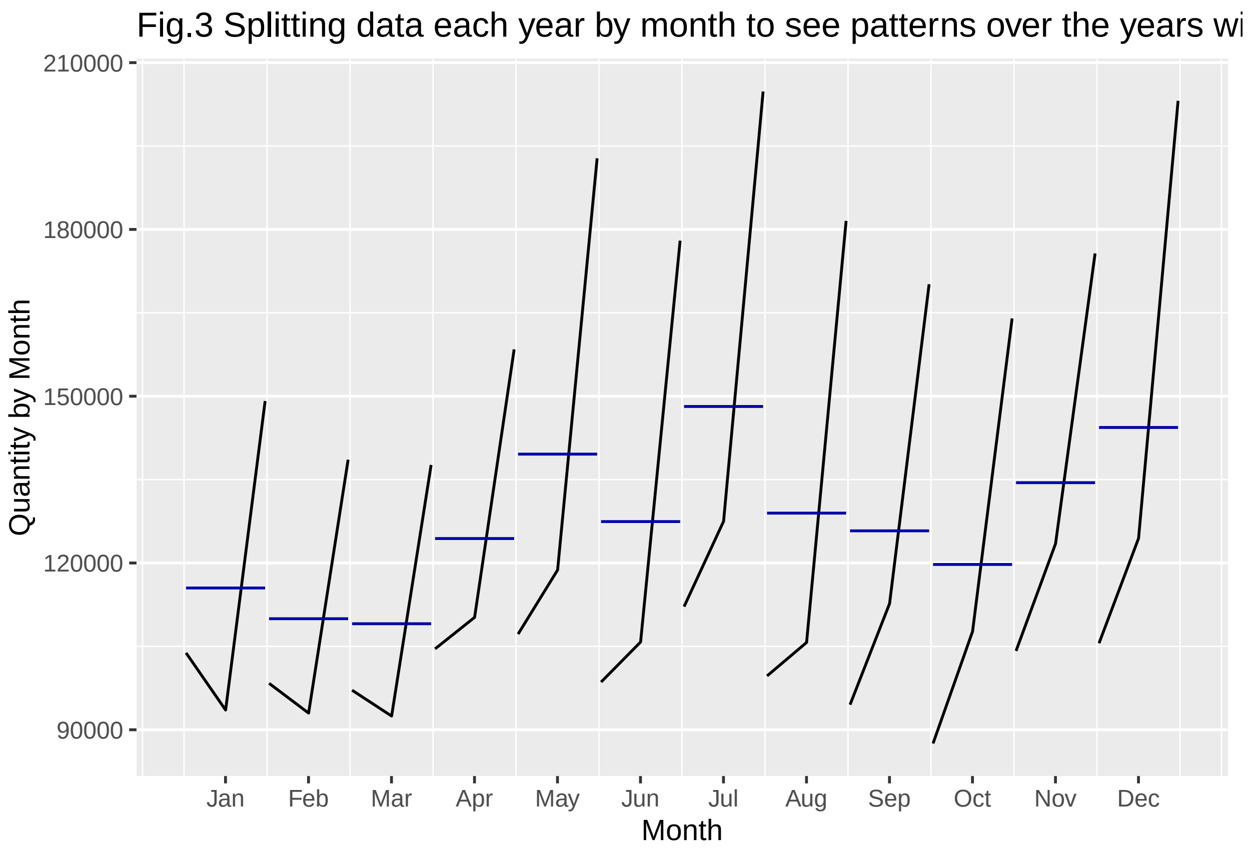
```
## Measure the strength of seasonality and trend


trendstrength<-max(0,1-(var(rem)/var(rem+Trendseries)))
trendstrength # 0.95 means there is a good trend
```

```
## [1] 0.9594412
```

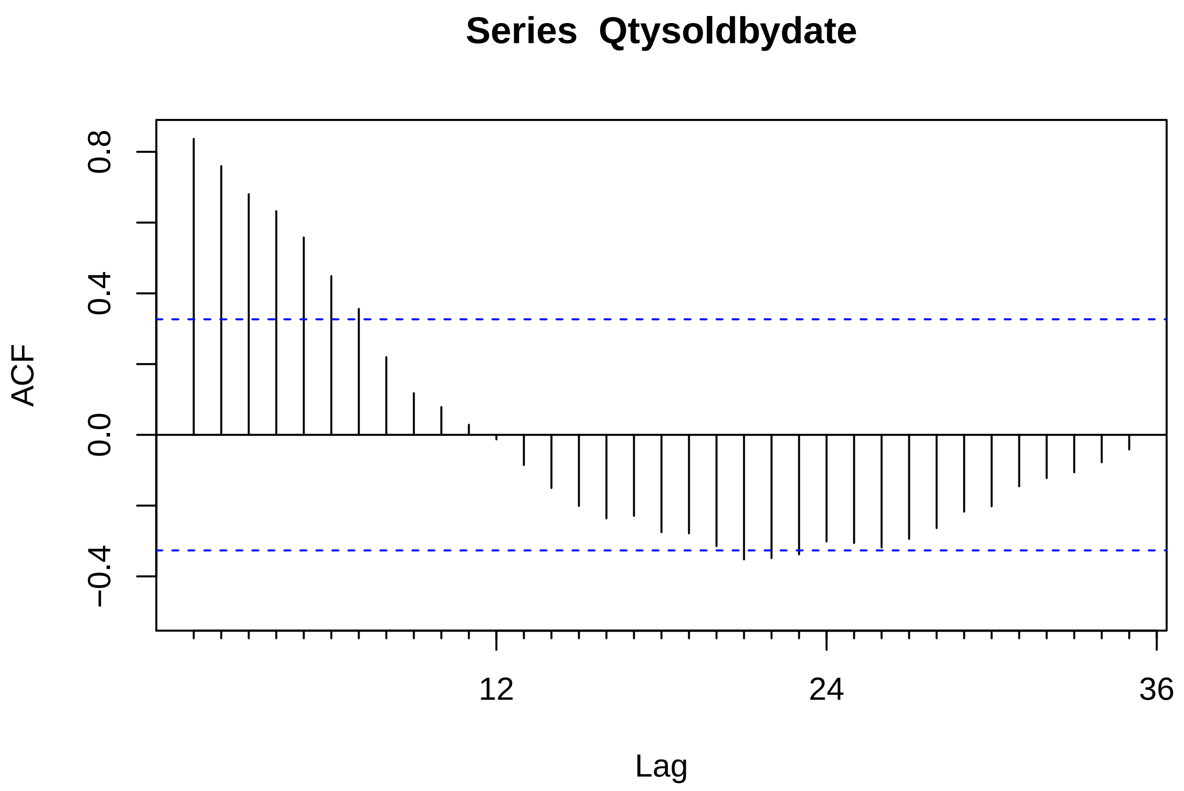
```
seasonalstrength<-max(0,1-(var(rem)/var(rem+Seasonal)))
seasonalstrength # 0.68 means there is a good seasonality
```

```
## [1] 0.6881794
```

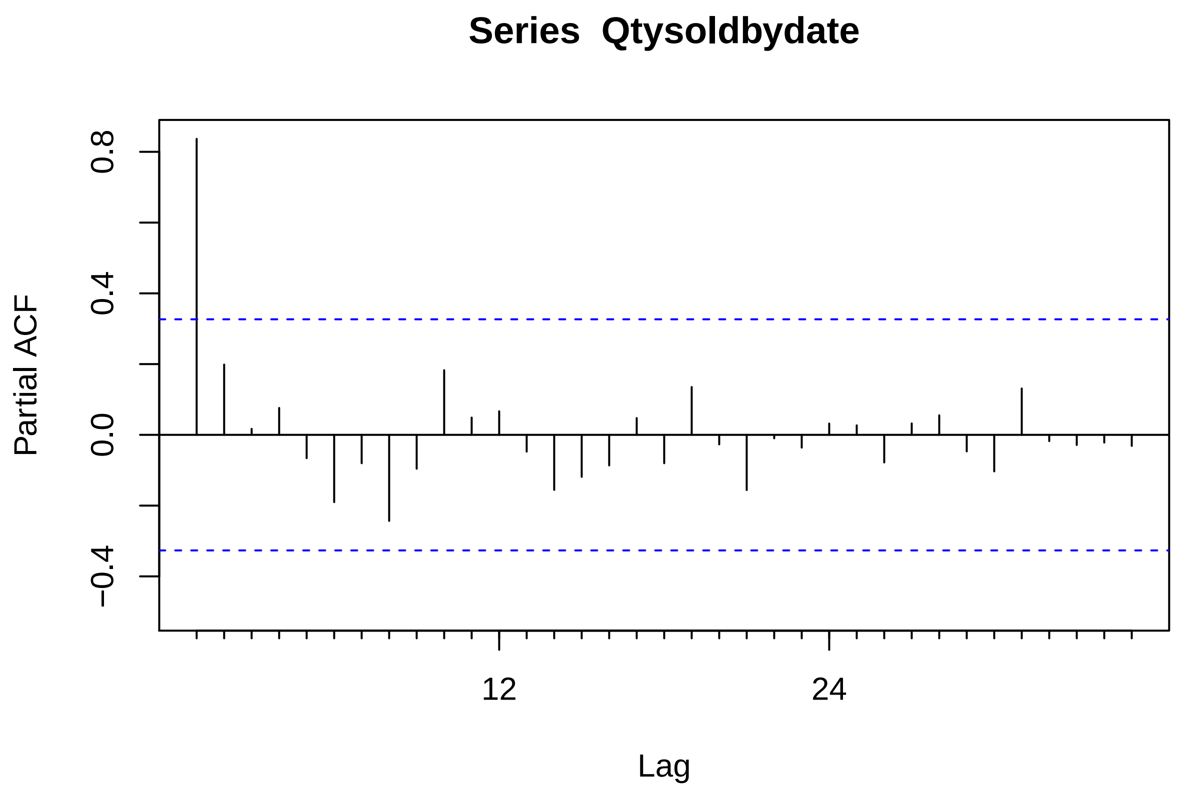
```
ggsubseriesplot(Qtysoldbydate)+ ggtitle("Fig.3 Splitting data each year by month to see
```



```
#Above graph shows the means with pattern on each month

# Looking at the subseries plot in Fig 3, there is an indication that the sales os low


# Autocorelation and partial corelation

# The idea of auto corealtion is self corelation, auto corealation is the moving corel


# Acf

Acf(Qtysoldbydate, lag.max = 36)
```



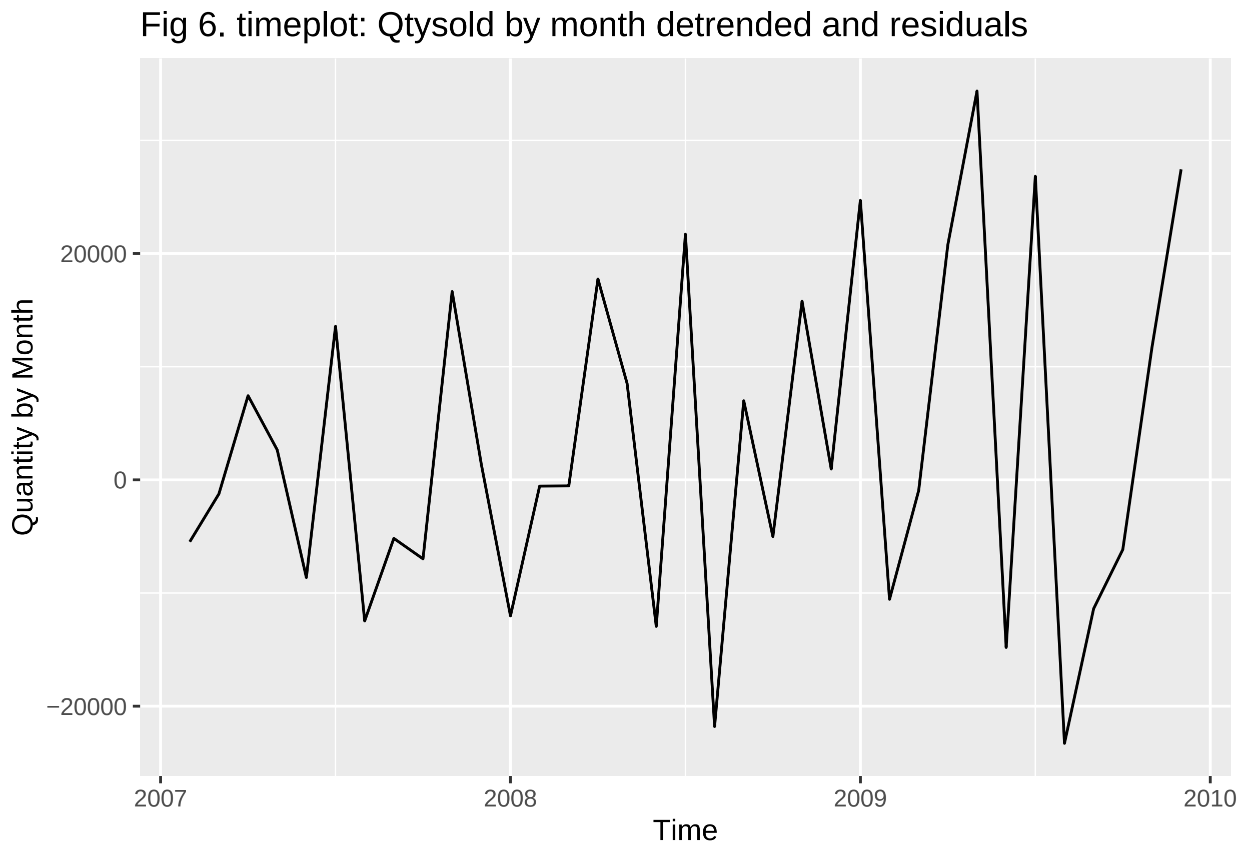
```
#It shows corelations at large in the nearest lags and starts to decrease autocorelati
Pacf(Qtysoldbydate, lag.max = 36)
```



```
#Pacf show the autocorelations and it is significant (over blue dotted lines). This al



# ARIMA woeks on differenced or stationary data so we would like to differce it. It re
# Performing dickey fuller tes
library(tseries)
adf.test(Qtysoldbydate)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Qtysoldbydate
## Dickey-Fuller = -1.7943, Lag order = 3, p-value = 0.6532
## alternative hypothesis: stationary
```

```
# from above the p-value is 0.6532 , that means differencing is needed


ts_data_diff<-diff(Qtysoldbydate)
# Getting detrended plot
autoplot(ts_data_diff)+ggtitle("Fig 6. timeplot: Qtysold by month detrended and residual
```

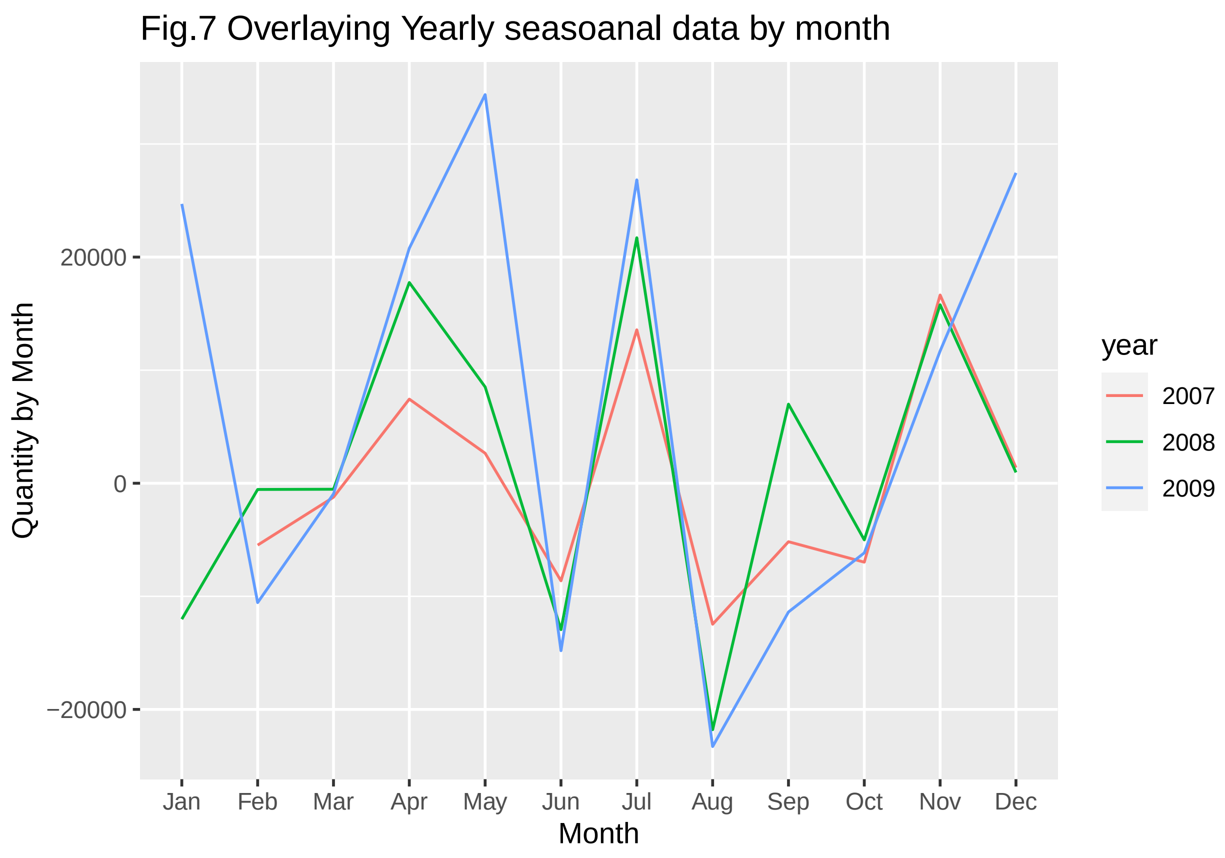


```
# Testing for stationary with adf test to see if the data was stationary and removed t
adf.test(ts_data_diff)
```

```
##
##  Augmented Dickey-Fuller Test
##
```

```
## data:  ts_data_diff
## Dickey-Fuller = -3.8984, Lag order = 3, p-value = 0.02491
## alternative hypothesis: stationary
```

```
# P value of 0.02 and lag order of 3 shows that data was turned staitonary and we are

ggseasonplot(ts_data_diff)+ ggtitle("Fig.7 Overlaying Yearly seasoanal data by month")+
```

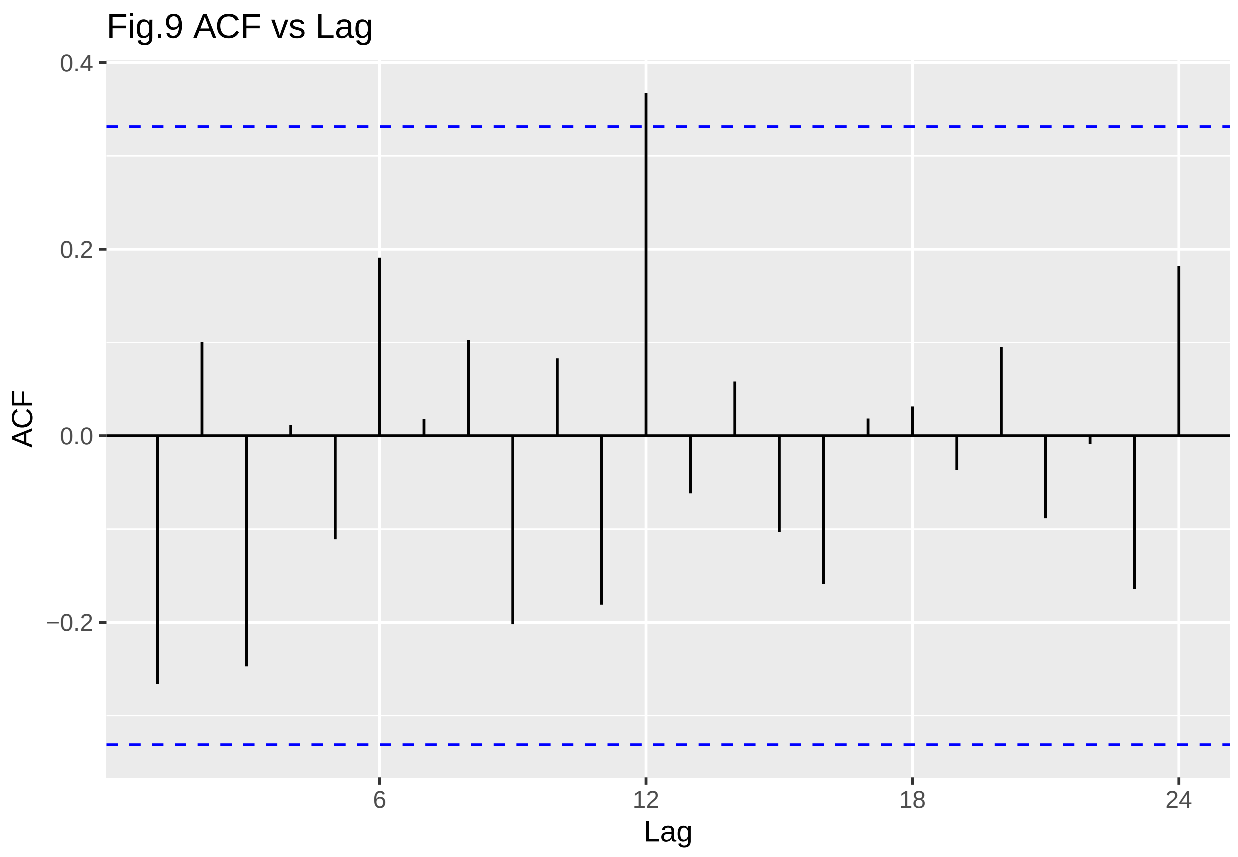


```
# The seasonal plot shows the seasonlity which lies every year. This gives us a indica

ggAcf(ts_data_diff, plot = FALSE)
```

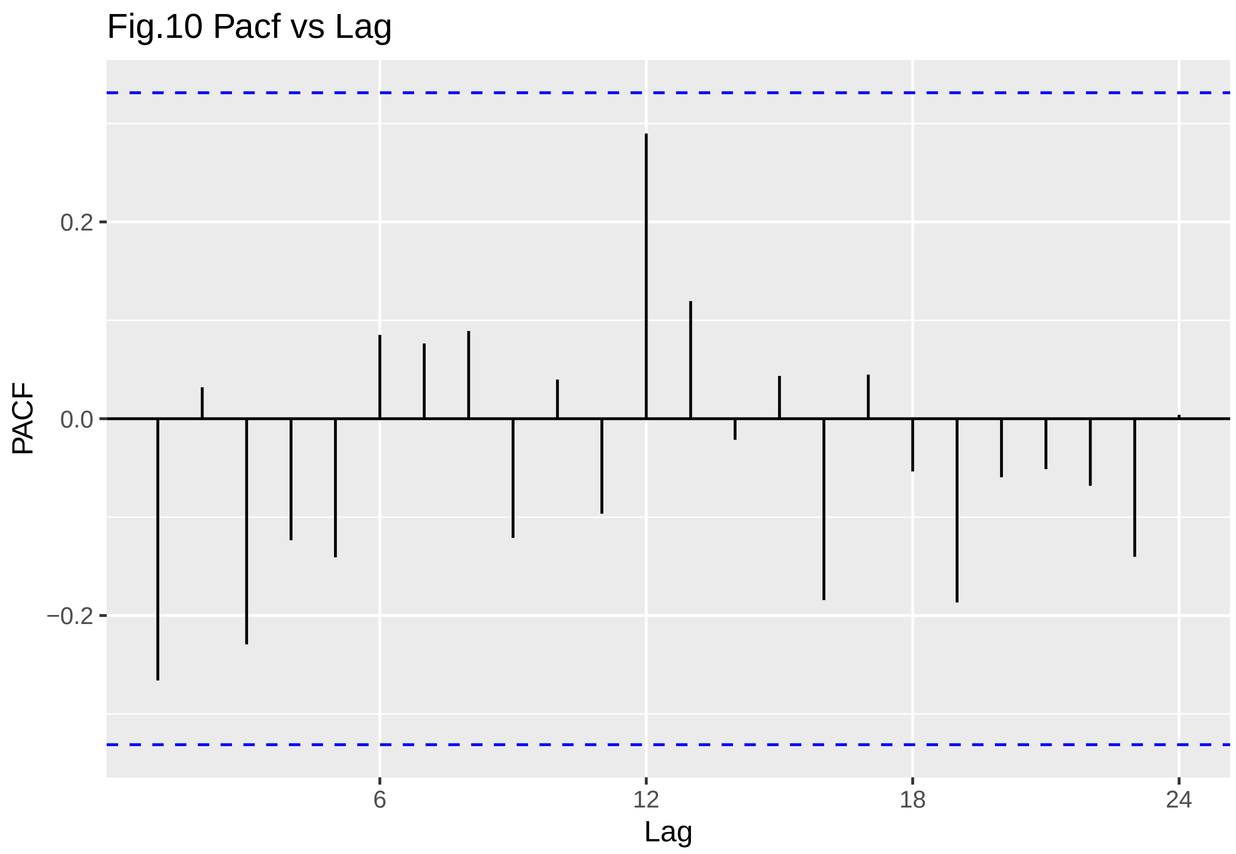
```
##
```

```
## Autocorrelations of series 'ts_data_diff', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000 -0.266  0.100 -0.247  0.012 -0.111  0.191  0.018  0.103 -0.202  0.083
##     11     12     13     14     15     16     17     18     19     20     21
## -0.181  0.368 -0.062  0.058 -0.103 -0.159  0.018  0.031 -0.037  0.095 -0.088
##     22     23     24
## -0.009 -0.164  0.182
```

```
# There seems to be some autocorrlations on the detrended data
# First testing on the white noise assumption
ggAcf(ts_data_diff, plot = TRUE) + ggtitle("Fig.9 ACF vs Lag ")
```



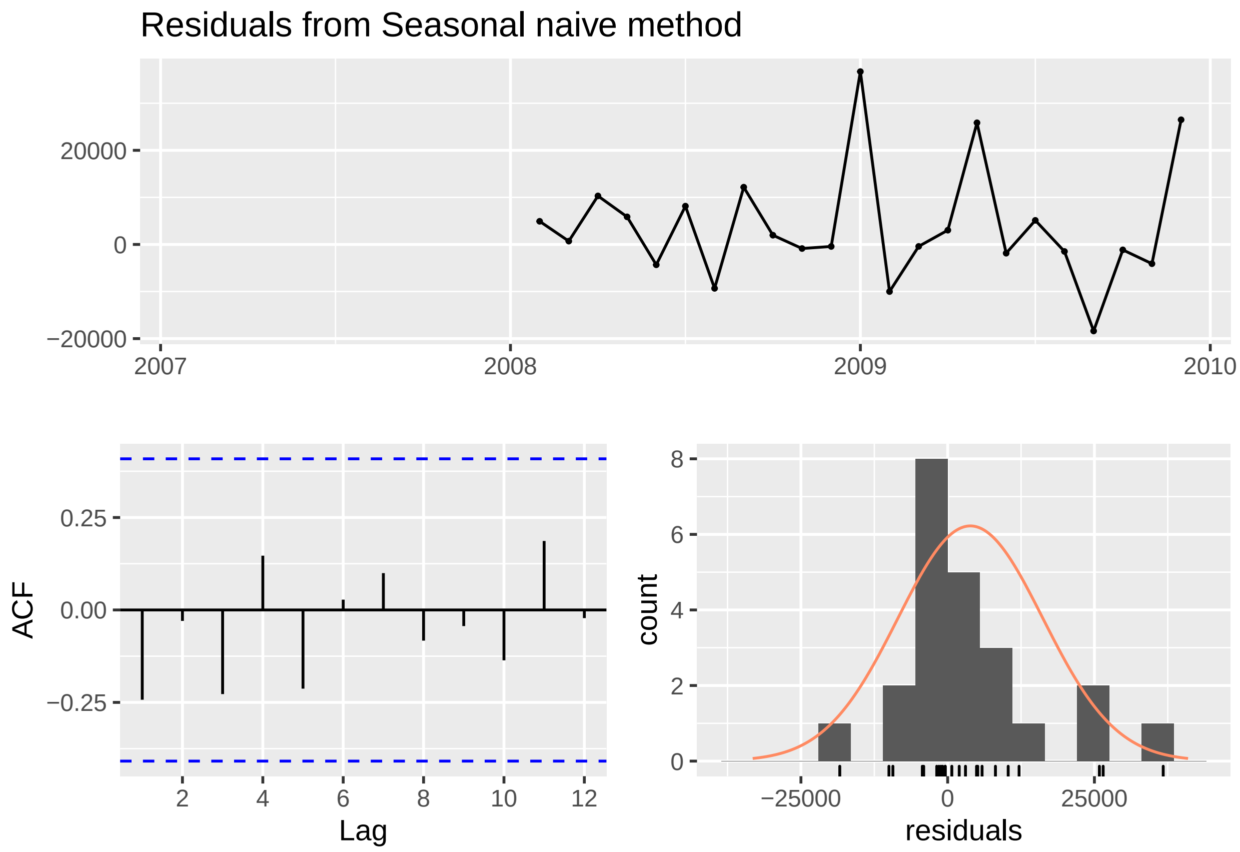
```
ggPacf(ts_data_diff) + ggtitle("Fig.10 Pacf vs Lag ")
```



```
# There seems to be some randomness in the data which can be forecasted via time serie


## Our series has both trend and seasonality so we need to choose such methods

########################################################## Forecast ##################

## we already took a first difference so we can use it on some benchmark method to pre

#############################Lets use the bench mark methods##########################

## As the data is seasonal we will use snaive method. Seaonal naive method value is th


######################################################################################

fitsnaive<-snaive(ts_data_diff)

print(summary(fitsnaive))# residual sd 12693.6602 on all data
```

```
##
## Forecast method: Seasonal naive method
##
## Model Information:
## Call: snaive(y = ts_data_diff)
##
## Residual sd: 12693.6602
##
## Error measures:
##                    ME     RMSE      MAE       MPE     MAPE MASE       ACF1
## Training set 3868.609 12693.66 8417.652 -1.935097 98.10028    1 -0.2427104
##
## Forecasts:
##          Point Forecast      Lo 80      Hi 80       Lo 95     Hi 95
## Jan 2010          24703   8435.420 40970.5801   -176.1168 49582.117
## Feb 2010         -10555 -26822.580  5712.5801 -35434.1168 14324.117
## Mar 2010           -939 -17206.580 15328.5801 -25818.1168 23940.117
## Apr 2010          20783   4515.420 37050.5801  -4096.1168 45662.117
## May 2010          34368  18100.420 50635.5801   9488.8832 59247.117
## Jun 2010         -14809 -31076.580  1458.5801 -39688.1168 10070.117
## Jul 2010          26831  10563.420 43098.5801   1951.8832 51710.117
## Aug 2010         -23286 -39553.580 -7018.4199 -48165.1168  1593.117
## Sep 2010         -11383 -27650.580  4884.5801 -36262.1168 13496.117
## Oct 2010          -6157 -22424.580 10110.5801 -31036.1168 18722.117
## Nov 2010          11698  -4569.580 27965.5801 -13181.1168 36577.117
## Dec 2010          27448  11180.420 43715.5801   2568.8832 52327.117
## Jan 2011          24703   1697.168 47708.8324 -10481.3845 59887.384
## Feb 2011         -10555 -33560.832 12450.8324 -45739.3845 24629.384
## Mar 2011           -939 -23944.832 22066.8324 -36123.3845 34245.384
## Apr 2011          20783  -2222.832 43788.8324 -14401.3845 55967.384
## May 2011          34368  11362.168 57373.8324   -816.3845 69552.384
## Jun 2011         -14809 -37814.832  8196.8324 -49993.3845 20375.384
## Jul 2011          26831   3825.168 49836.8324  -8353.3845 62015.384
## Aug 2011         -23286 -46291.832  -280.1676 -58470.3845 11898.384
## Sep 2011         -11383 -34388.832 11622.8324 -46567.3845 23801.384
## Oct 2011          -6157 -29162.832 16848.8324 -41341.3845 29027.384
## Nov 2011          11698 -11307.832 34703.8324 -23486.3845 46882.384
## Dec 2011          27448   4442.168 50453.8324  -7736.3845 62632.384
##          Point Forecast      Lo 80      Hi 80       Lo 95     Hi 95
## Jan 2010          24703   8435.420 40970.5801   -176.1168 49582.117
## Feb 2010         -10555 -26822.580  5712.5801 -35434.1168 14324.117
## Mar 2010           -939 -17206.580 15328.5801 -25818.1168 23940.117
## Apr 2010          20783   4515.420 37050.5801  -4096.1168 45662.117
## May 2010          34368  18100.420 50635.5801   9488.8832 59247.117
## Jun 2010         -14809 -31076.580  1458.5801 -39688.1168 10070.117
## Jul 2010          26831  10563.420 43098.5801   1951.8832 51710.117
## Aug 2010         -23286 -39553.580 -7018.4199 -48165.1168  1593.117
## Sep 2010         -11383 -27650.580  4884.5801 -36262.1168 13496.117
## Oct 2010          -6157 -22424.580 10110.5801 -31036.1168 18722.117
## Nov 2010          11698  -4569.580 27965.5801 -13181.1168 36577.117
## Dec 2010          27448  11180.420 43715.5801   2568.8832 52327.117
## Jan 2011          24703   1697.168 47708.8324 -10481.3845 59887.384
## Feb 2011         -10555 -33560.832 12450.8324 -45739.3845 24629.384
## Mar 2011           -939 -23944.832 22066.8324 -36123.3845 34245.384
## Apr 2011          20783  -2222.832 43788.8324 -14401.3845 55967.384
## May 2011          34368  11362.168 57373.8324   -816.3845 69552.384
## Jun 2011         -14809 -37814.832  8196.8324 -49993.3845 20375.384
## Jul 2011          26831   3825.168 49836.8324  -8353.3845 62015.384
## Aug 2011         -23286 -46291.832  -280.1676 -58470.3845 11898.384
## Sep 2011         -11383 -34388.832 11622.8324 -46567.3845 23801.384
## Oct 2011          -6157 -29162.832 16848.8324 -41341.3845 29027.384
## Nov 2011          11698 -11307.832 34703.8324 -23486.3845 46882.384
## Dec 2011          27448   4442.168 50453.8324  -7736.3845 62632.384
```

```
# the lessor the residuls the better
checkresiduals(fitsnaive) # the acf on the fitted data is within the two dotted line sc
```



```
##
```

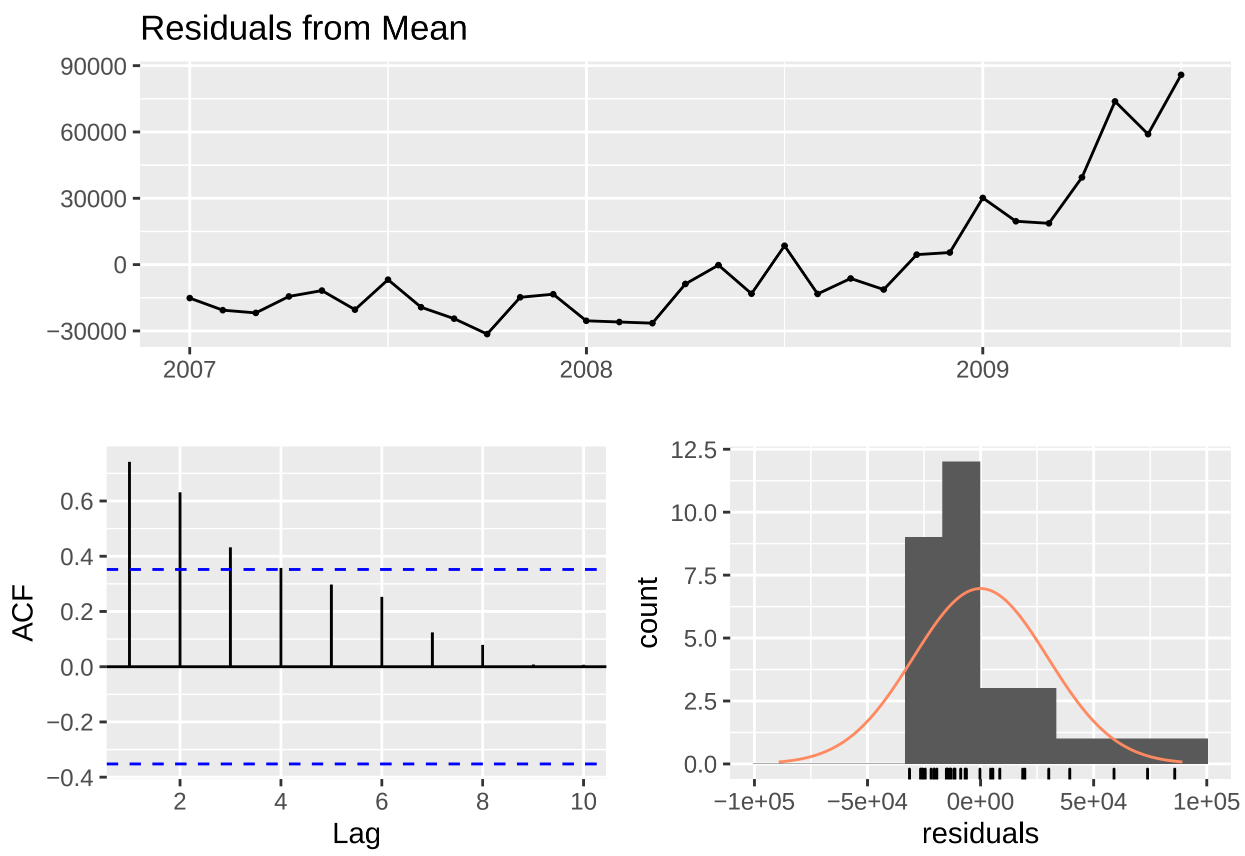
```
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 5.5322, df = 7, p-value = 0.5953
##
## Model df: 0.   Total lags used: 7
```

```
#Lest split the data into test with data until July and keep the rest of the data in 2

training <- window(Qtysoldbydate,start=2007,end=c(2009,7), frequency=12)

mean_model_fit <- meanf(training,h=5)# mean forecast method
naive_model_fit <- rwf(training,h=5)#naive forecast method
snaive_model_fit <- snaive(training,h=5)#seasonal naive method



# Now lets fitting and residuals for mean_model_fit
print(summary(mean_model_fit)) # model training MAE=22264.53 and MPE=-4.82065
```

```
##
## Forecast method: Mean
##
## Model Information:
## $mu
## [1] 118963.7
##
## $mu.se
## [1] 5341.414
```

```
##
## $sd
## [1] 29739.73
##
## $bootstrap
## [1] FALSE
##
## $call
## meanf(y = training, h = 5)
##
## attr(,"class")
## [1] "meanf"
##
## Error measures:
##                        ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 5.160603e-12 29256.13 22264.53 -4.82065 17.78558 0.7546181
##                  ACF1
## Training set 0.741704
##
## Forecasts:
##          Point Forecast   Lo 80    Hi 80    Lo 95    Hi 95
## Aug 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Sep 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Oct 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Nov 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Dec 2009       118963.7 79368.7 158558.7 57255.19 180672.2
##          Point Forecast   Lo 80    Hi 80    Lo 95    Hi 95
## Aug 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Sep 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Oct 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Nov 2009       118963.7 79368.7 158558.7 57255.19 180672.2
## Dec 2009       118963.7 79368.7 158558.7 57255.19 180672.2
```

```
checkresiduals(mean_model_fit) # Ljung - Box test states that residual plot has data le
```



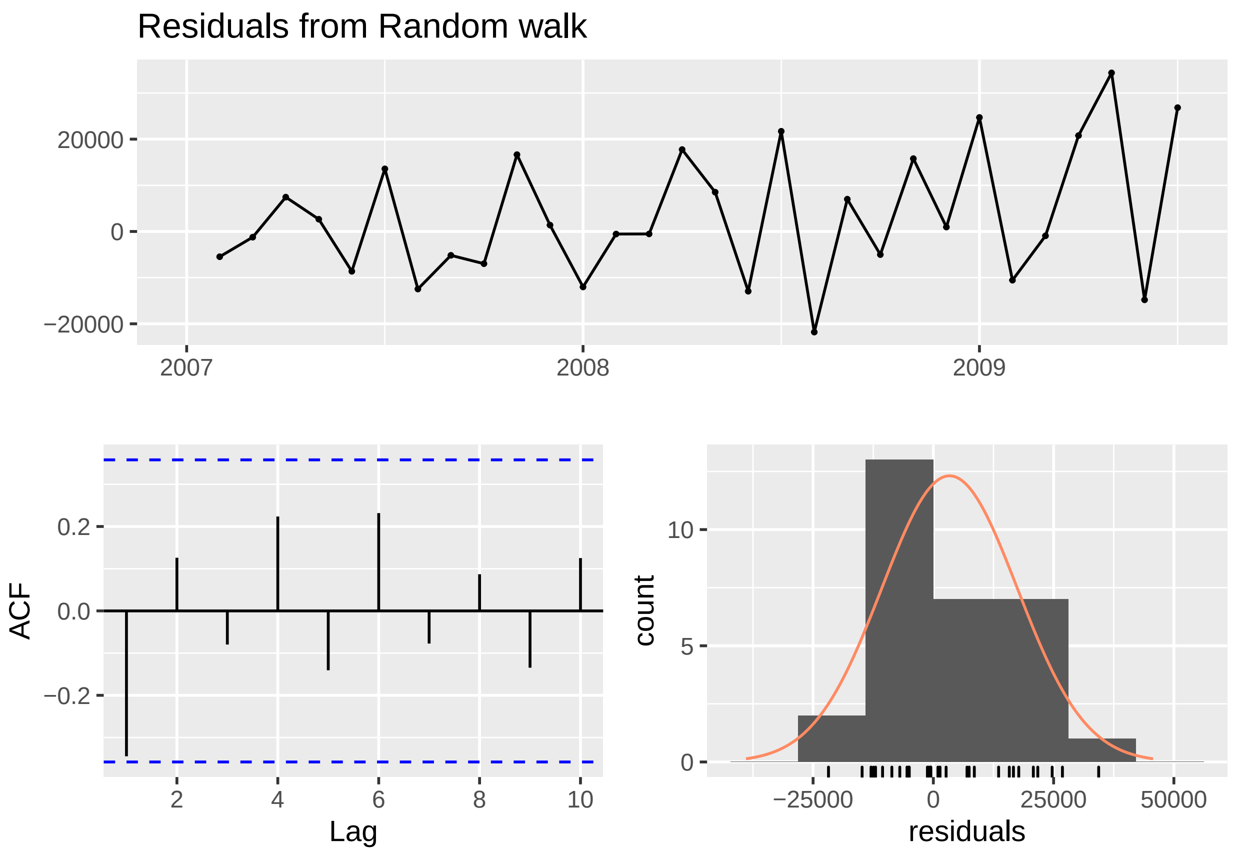
```
##
##  Ljung-Box test
##
## data:  Residuals from Mean
## Q* = 50.593, df = 5, p-value = 1.048e-09
##
```

```
## Model df: 1.   Total lags used: 6
```

```r
# Now lets fitting and residuals for naive_model_fit
print(summary(naive_model_fit)) # model training MAE=11307.03 and MAPE=17.54458%%
```

```
##
## Forecast method: Random walk
##
## Model Information:
## Call: rwf(y = training, h = 5)
##
## Residual sd: 14269.1379
##
## Error measures:
##                     ME     RMSE      MAE      MPE     MAPE      MASE       ACF1
## Training set 3366.167 14269.14 11307.03 1.665768 8.978212 0.3832325 -0.3444244
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Aug 2009         204810 186523.4 223096.6 176843.0 232777.0
## Sep 2009         204810 178948.8 230671.2 165258.7 244361.3
## Oct 2009         204810 173136.6 236483.4 156369.7 253250.3
## Nov 2009         204810 168236.7 241383.3 148876.0 260744.0
## Dec 2009         204810 163919.8 245700.2 142273.9 267346.1
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Aug 2009         204810 186523.4 223096.6 176843.0 232777.0
## Sep 2009         204810 178948.8 230671.2 165258.7 244361.3
## Oct 2009         204810 173136.6 236483.4 156369.7 253250.3
## Nov 2009         204810 168236.7 241383.3 148876.0 260744.0
## Dec 2009         204810 163919.8 245700.2 142273.9 267346.1
```

```
checkresiduals(naive_model_fit) # Ljung - Box test and p value=0.1498 which is  not sig
```

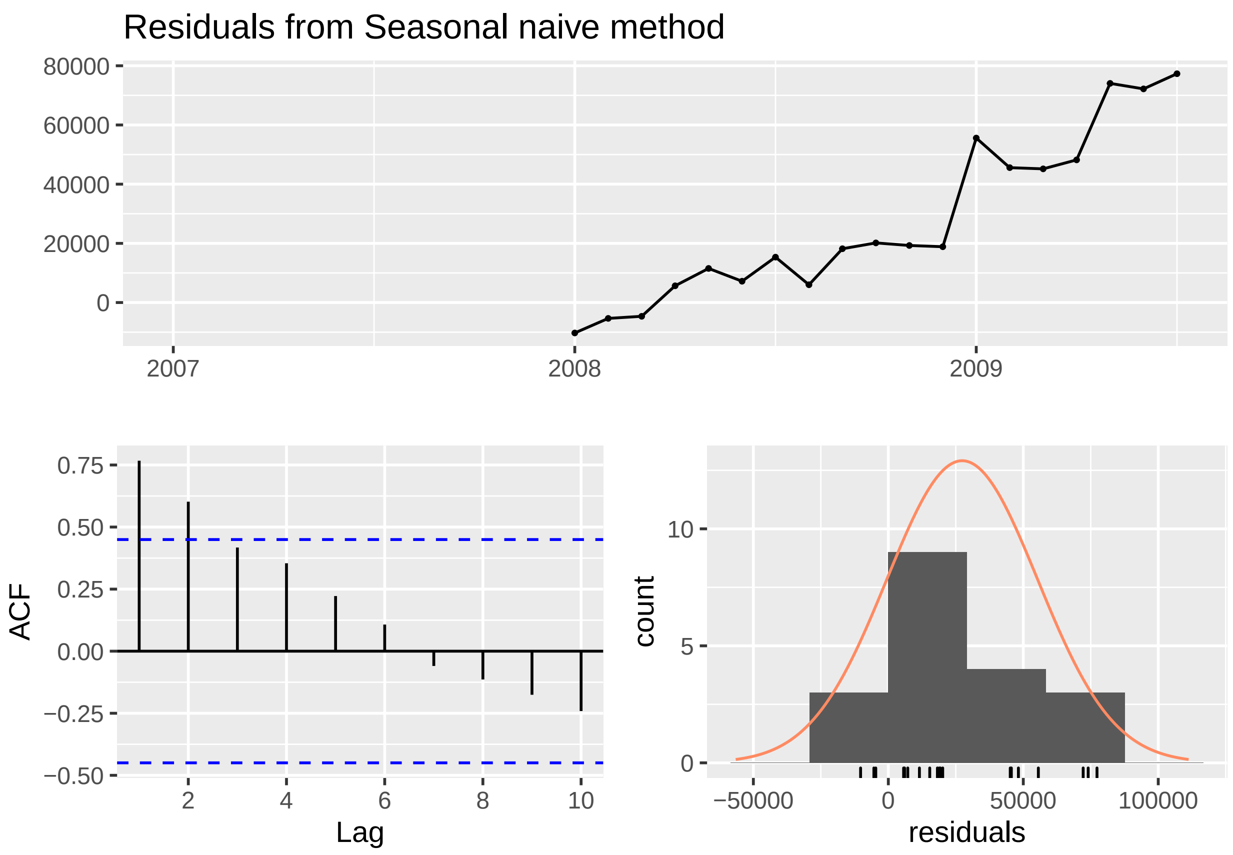


```
##
##  Ljung-Box test
##
## data:  Residuals from Random walk
## Q* = 9.4497, df = 6, p-value = 0.1498
##
## Model df: 0.   Total lags used: 6
```

```
# Now lets fitting and residuals for snaive_model_fit
print(summary(snaive_model_fit)) # model training MAE=29504.37 and MAPE=17.54458%%
```

```
##
## Forecast method: Seasonal naive method
##
## Model Information:
## Call: snaive(y = training, h = 5)
##
## Residual sd: 38601.1977
##
## Error measures:
##                    ME    RMSE      MAE      MPE     MAPE MASE      ACF1
## Training set 27369.63 38601.2 29504.37 17.54458 19.83621    1 0.7670977
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80    Lo 95  Hi 95
## Aug 2009         105699 56229.57 155168.4 30042.04 181356
## Sep 2009         112696 63226.57 162165.4 37039.04 188353
## Oct 2009         107686 58216.57 157155.4 32029.04 183343
## Nov 2009         123471 74001.57 172940.4 47814.04 199128
## Dec 2009         124428 74958.57 173897.4 48771.04 200085
##          Point Forecast    Lo 80    Hi 80    Lo 95  Hi 95
## Aug 2009         105699 56229.57 155168.4 30042.04 181356
## Sep 2009         112696 63226.57 162165.4 37039.04 188353
## Oct 2009         107686 58216.57 157155.4 32029.04 183343
## Nov 2009         123471 74001.57 172940.4 47814.04 199128
## Dec 2009          124428 74958.57 173897.4 48771.04 200085
```

```
checkresiduals(snaive_model_fit) # Ljung - Box test and p value=2.542e-05 which is  sig
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 30.994, df = 6, p-value = 2.542e-05
##
## Model df: 0.   Total lags used: 6
```

```
#Testing for accuracty with test data
testing <- window(Qtysoldbydate, start=c(2009,8), frequency=12)



mean_model_predict<- predict(mean_model_fit, h=5)
naive_model_predict<- predict(naive_model_fit, h=5)
snaive_model_predict<- predict(snaive_model_fit, h=5)

# getting accuracy of prediction
accuracy(mean_model_predict$mean, testing) # Testing set fit - MAE= 59928.52 and MPE=33
```

```
##                ME     RMSE      MAE      MPE     MAPE       ACF1 Theil's U
## Test set 59928.52 61417.72 59928.52 33.14335 33.14335 0.08575525  3.788616
```

```
accuracy(naive_model_predict$mean, testing) # Testing set fit - MAE= 25917.8 and MPE=-1
```

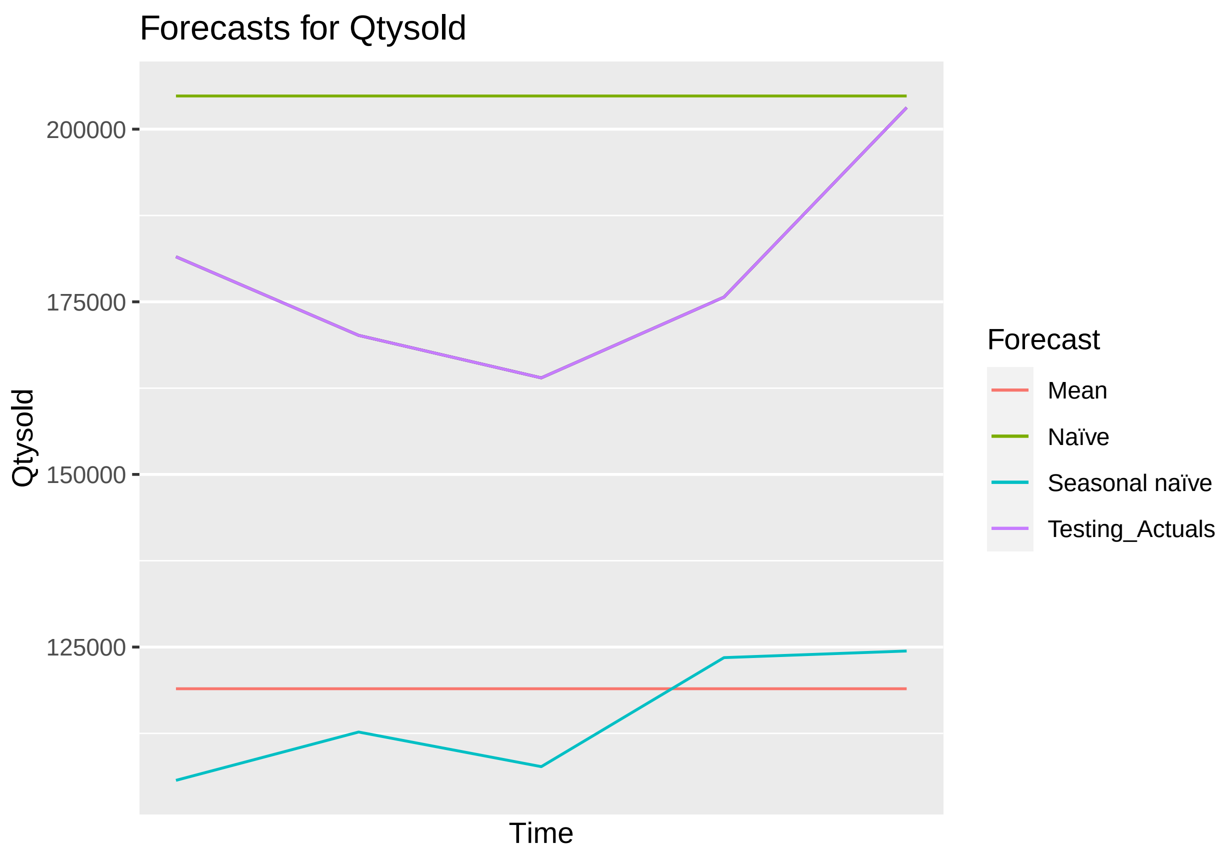
```
##                ME     RMSE     MAE       MPE     MAPE       ACF1 Theil's U
## Test set -25917.8 29196.58 25917.8 -15.10161 15.10161 0.08575525  1.902147
```

```
accuracy(snaive_model_predict$mean, testing) # Testing set fit - MAE= 64096.2  and MPE=
```

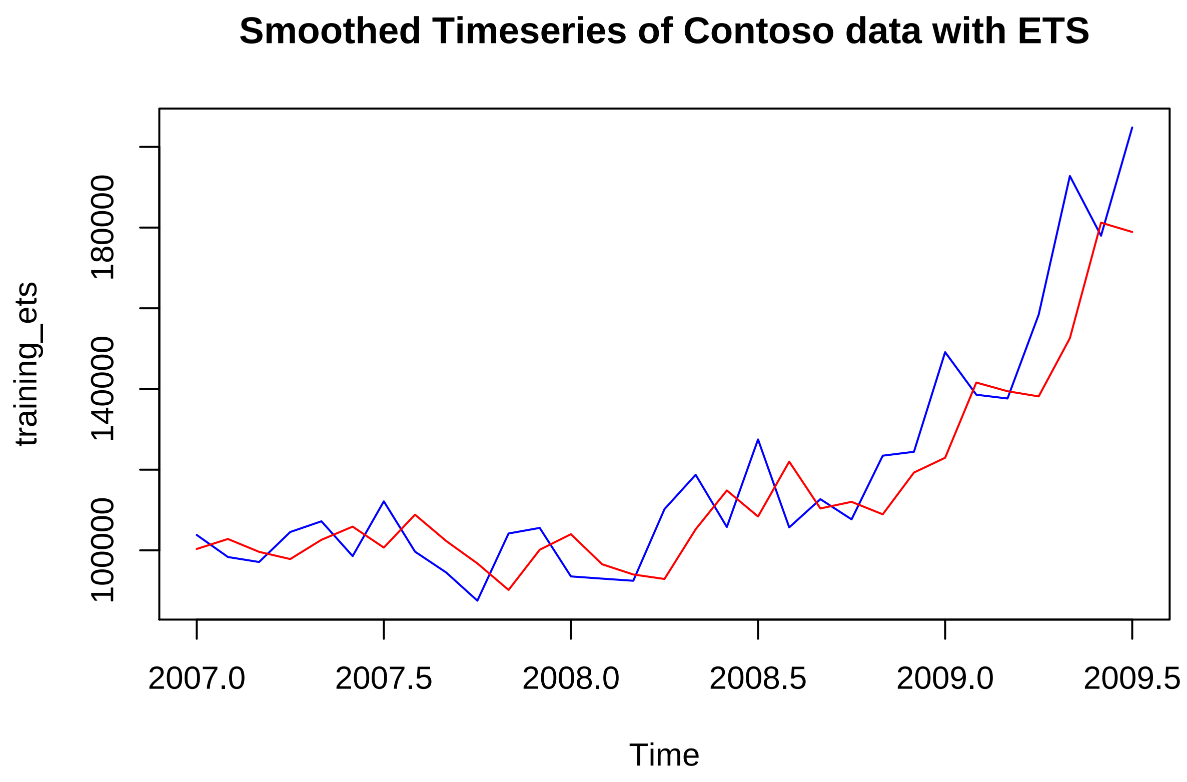
```
##               ME     RMSE     MAE      MPE     MAPE       ACF1 Theil's U
## Test set 64096.2 65021.25 64096.2 35.66592 35.66592 -0.1792573  3.837833
```

```
autoplot(testing) +
  autolayer(testing, series="Testing_Actuals", PI=FALSE)+
  autolayer(mean_model_predict$mean, series="Mean", PI=FALSE) +
  autolayer(naive_model_predict$mean, series="Naïve", PI=FALSE) +
  autolayer(snaive_model_predict$mean, series="Seasonal naïve", PI=FALSE) +
 ylab("Qtysold") + # xlab("Year") +
  ggtitle("Forecasts for Qtysold") +
  guides(colour=guide_legend(title="Forecast"))
```

```
## Warning: Ignoring unknown parameters: PI

## Warning: Ignoring unknown parameters: PI

## Warning: Ignoring unknown parameters: PI

## Warning: Ignoring unknown parameters: PI
```



```
## My analysis so far was the seasonal method fitted the data better. The drawback was


#Differenced data is not needed for ETS
training_ets <- window(Qtysoldbydate,start=2007,end=c(2009,7))
testing_data <- window(Qtysoldbydate, start=c(2009,8))
ETS<-ets( training_ets)
plot.ts(training_ets,main = "Smoothed Timeseries of Contoso data with ETS", col = "blue"
  lines(fitted(ETS),col = "red")
```

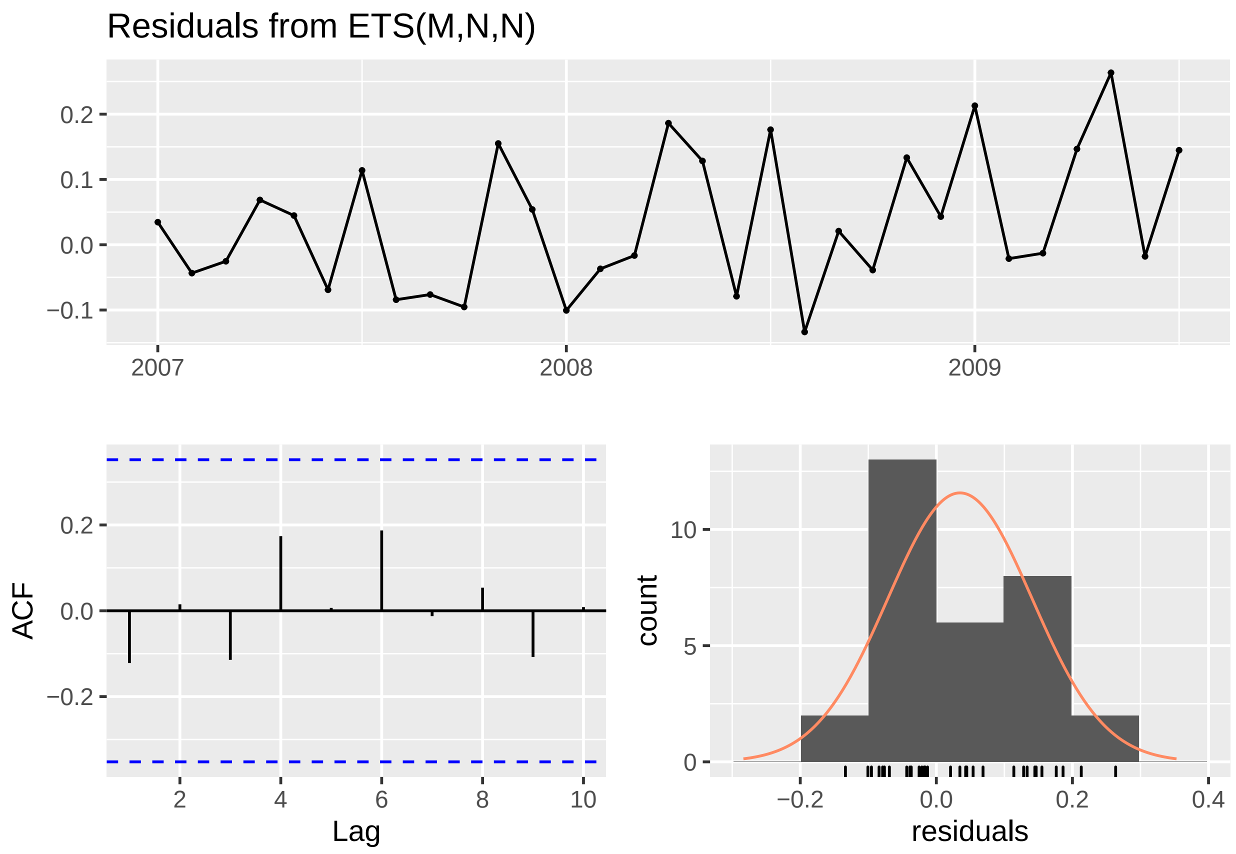


```
## integer(0)
```

```
# The model it picked was (M, N, N)- M error type, N trend type, and N season type.The


# Now lets fitting and residuals for ETS
print(summary(ETS)) # model training MAE=10448.99 and MAPE=2.389901%%
```

```
## ETS(M,N,N)
##
## Call:
##  ets(y = training_ets)
##
##   Smoothing parameters:
##     alpha = 0.712
##
##   Initial states:
##     l = 100354.3346
##
##   sigma:  0.1137
##
##      AIC     AICc      BIC
## 696.7724 697.6613 701.0744
##
## Training set error measures:
##                    ME     RMSE      MAE      MPE     MAPE      MASE        ACF1
## Training set 4394.416 13597.03 10448.99 2.389901 8.362708 0.3541505 -0.08489171
##                    ME     RMSE      MAE      MPE     MAPE      MASE        ACF1
## Training set 4394.416 13597.03 10448.99 2.389901 8.362708 0.3541505 -0.08489171
```

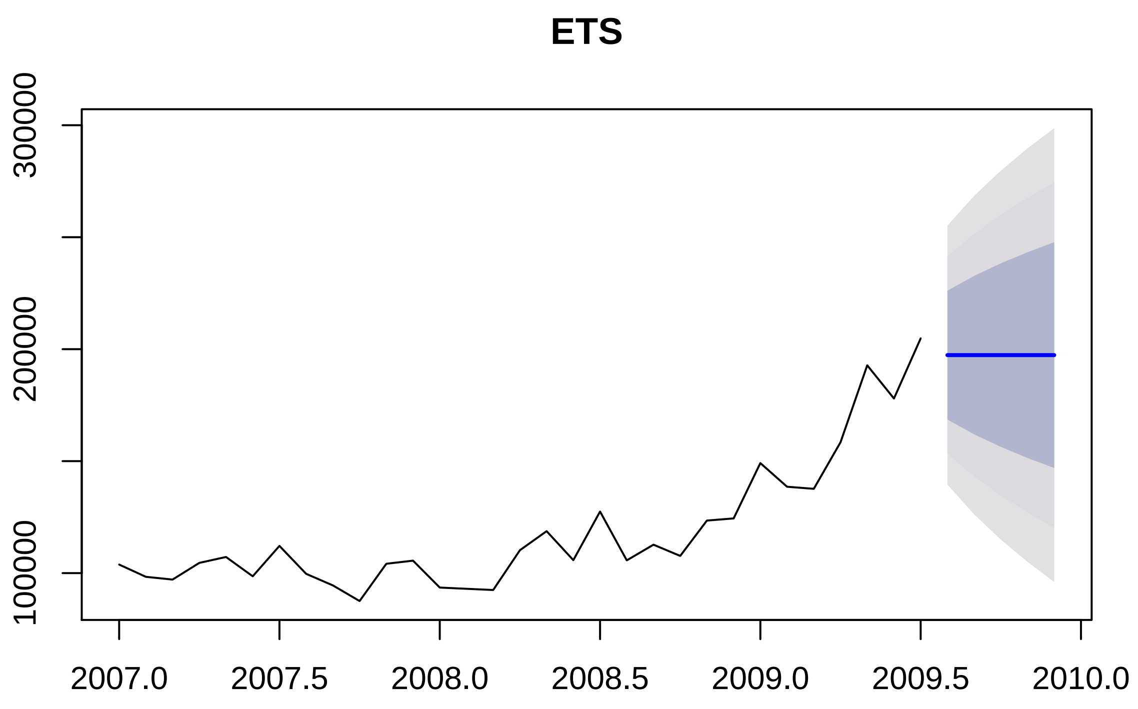
```
checkresiduals(ETS) # Ljung - Box test and p value=0.4666 which is  significant # There
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,N,N)
## Q* = 3.575, df = 4, p-value = 0.4666
##
## Model df: 2.   Total lags used: 6
```

```
predictions_ets<-forecast(ETS,
                          h=5, level = c(80, 95, 99))
## Plotting
```

```
plot(predictions_ets, main="ETS")
```



```
# Accuracy
accuracy(predictions_ets$mean, testing_data)
```

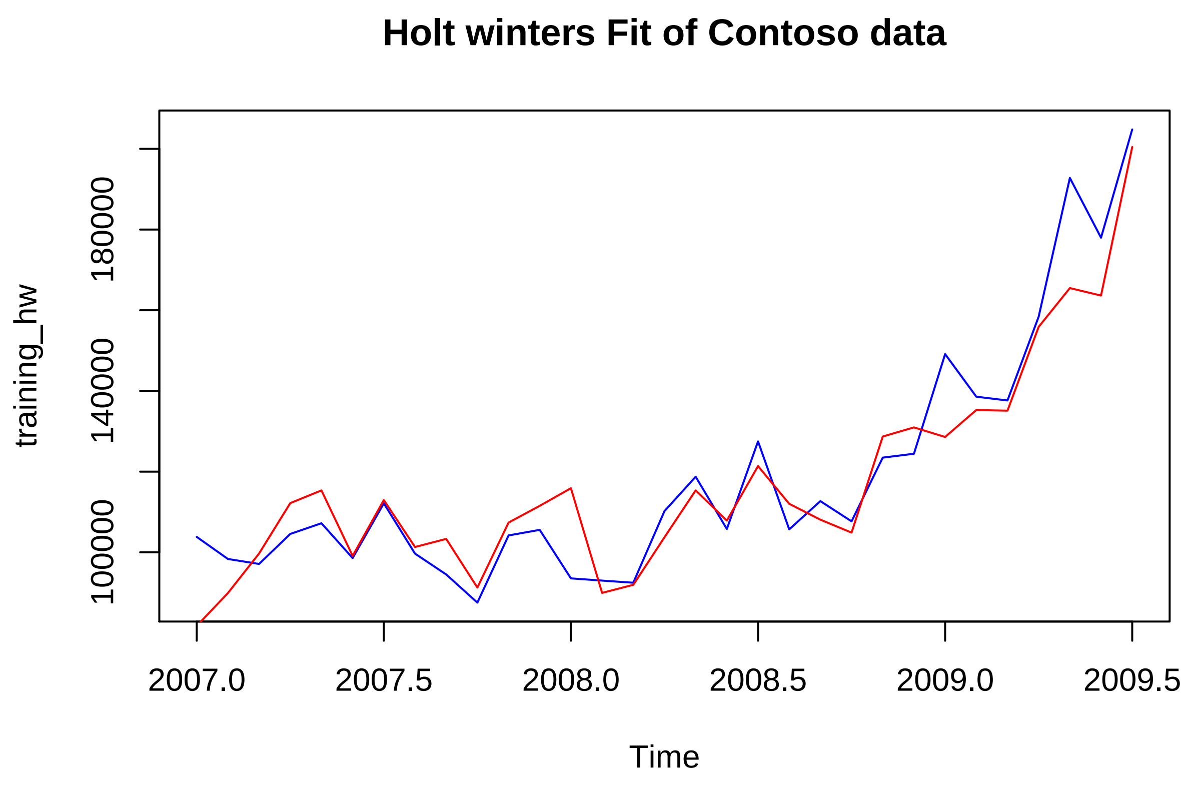
```
##                   ME     RMSE      MAE      MPE     MAPE       ACF1 Theil's U
## Test set -18459.14 22835.24 20770.61 -10.9099 12.04782 0.08575525  1.512503
```

```
# Model predicted a constant value of 197351.3 for a next five months. While the model

####################################################################
# Now try Holt winters seasonal trend method
##################################################################
```

```r
#Differenced data is not needed for Holt winters
training_hw <-training_ets
holtwinters <- hw(training_hw, seasonal = "multiplicative")
fitted(holtwinters)
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2007  81770.13  89912.49  99712.24 112195.69 115352.65  99114.03 112955.94
## 2008 115893.65  89948.47  91949.17 103730.92 115355.56 107865.05 121374.97
## 2009 128591.12 135272.19 135113.89 155884.98 165494.36 163636.85 200455.76
##            Aug       Sep       Oct       Nov       Dec
## 2007 101319.56 103327.50  91266.89 107375.52 111520.53
## 2008 112029.71 108095.60 104889.67 128705.80 130966.24
## 2009
```

```r
plot.ts(training_hw,main = "Holt winters Fit of Contoso data", col = "blue")+
  lines(fitted(holtwinters),col = "red")
```



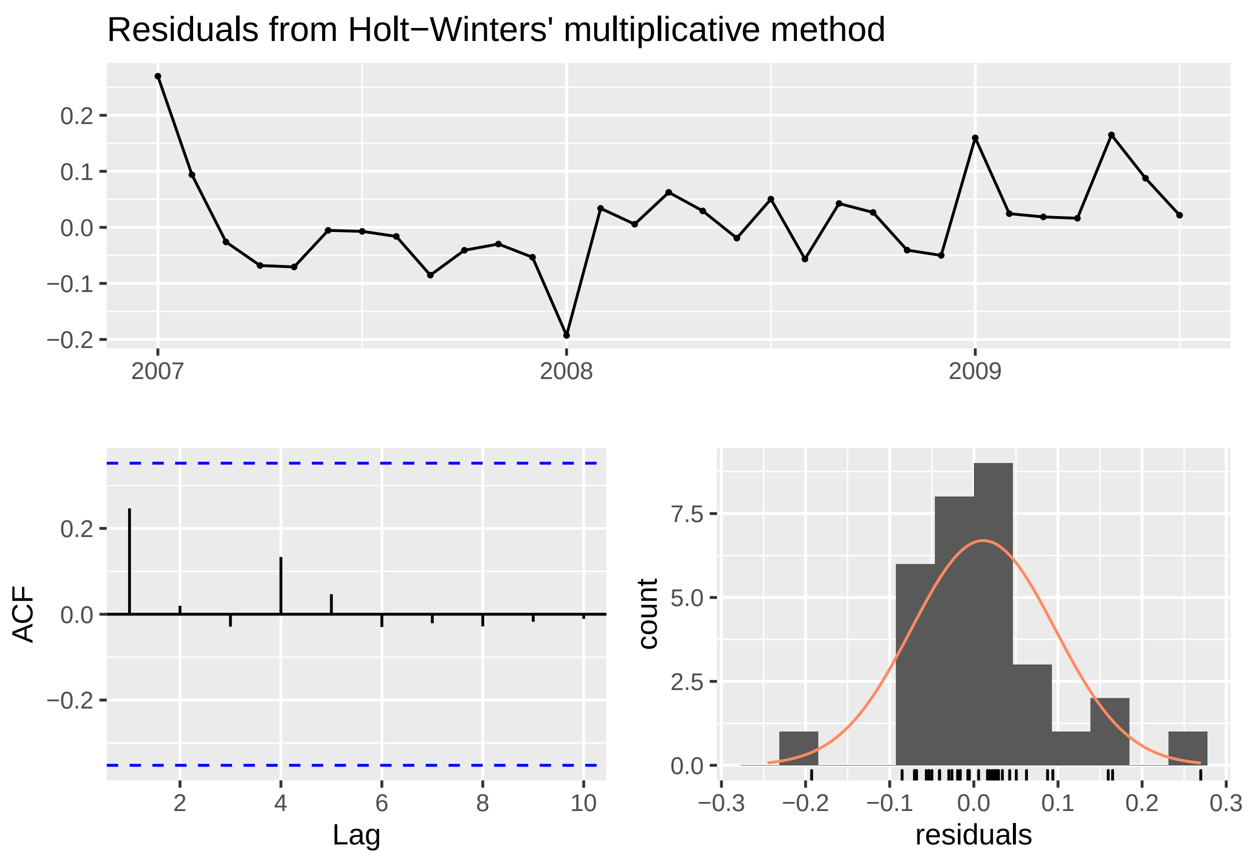
```
## integer(0)
```

```
# The Holt witners model seems to be a good fit from the plot. It seems to fit the pat



# Now lets fitting and residuals for holtwinters
holtwinters$model #This holt winters model has the residual of 0.1219 which is negligi
```

```
## Holt-Winters' multiplicative method
##
## Call:
##  hw(y = training_hw, seasonal = "multiplicative")
##
```

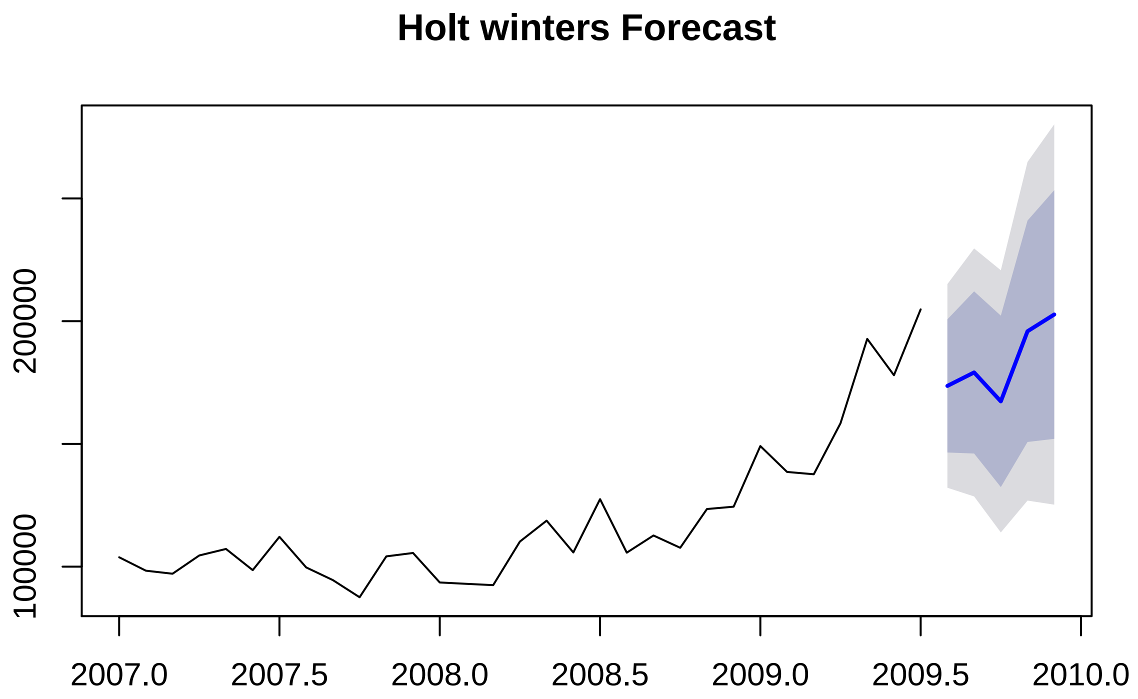
```
##   Smoothing parameters:
##     alpha = 0.6268
##     beta  = 0.0049
##     gamma = 0.3732
##
##   Initial states:
##     l = 78032.7739
##     b = 2484.6894
##     s = 1.0646 1.0301 0.8738 0.9593 0.9536 1.0843
##            0.972 1.109 1.0589 0.9492 0.9296 1.0156
##
##   sigma:  0.1219
##
##      AIC     AICc      BIC
## 709.9671 757.0441 734.3449
```

```
checkresiduals(holtwinters) # Ljung - Box test and p value=0.0009679 which is  signific
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 16.335, df = 3, p-value = 0.0009679
##
## Model df: 16.   Total lags used: 19
```

```
predictions_hw<-forecast(holtwinters,  h=5)
## Plotting
plot(predictions_hw, main="Holt winters Forecast")
```



```
# Accuracy
accuracy(predictions_hw$mean, testing_data)
```

```
##                    ME     RMSE      MAE       MPE     MAPE       ACF1 Theil's U
## Test set -4842.937 10606.57 8169.102 -2.852502 4.675093 -0.3648649 0.7201366
```

```
# Model didn't predict a contant value but value changed over the months which is a be


# Now lets apply Arima
training_ARIMA <-training_ets
fit_arima<-auto.arima(training_ARIMA,d=1,D=1, stepwise = F, approximation = F , trace=T)
```

```
##
```

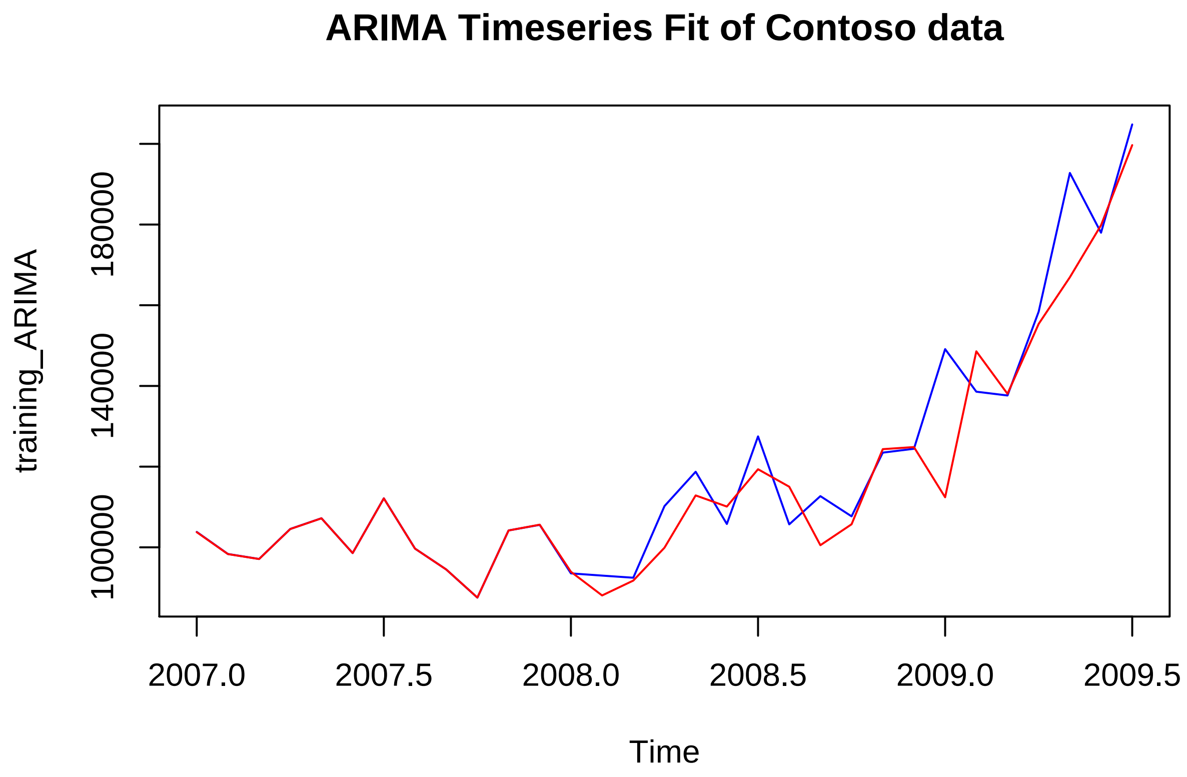
```
##  ARIMA(0,1,0)(0,1,0)[12]                    : 391.8397
##  ARIMA(0,1,1)(0,1,0)[12]                    : 393.8015
##  ARIMA(0,1,2)(0,1,0)[12]                    : 396.5803
##  ARIMA(0,1,3)(0,1,0)[12]                    : 398.6692
##  ARIMA(0,1,4)(0,1,0)[12]                    : 398.5008
##  ARIMA(0,1,5)(0,1,0)[12]                    : Inf
##  ARIMA(1,1,0)(0,1,0)[12]                    : 393.7175
##  ARIMA(1,1,1)(0,1,0)[12]                    : 395.4737
##  ARIMA(1,1,2)(0,1,0)[12]                    : 395.5273
##  ARIMA(1,1,3)(0,1,0)[12]                    : 399.1734
##  ARIMA(1,1,4)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,0)(0,1,0)[12]                    : 396.6008
##  ARIMA(2,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,2)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,3)(0,1,0)[12]                    : Inf
##  ARIMA(3,1,0)(0,1,0)[12]                    : 399.9635
##  ARIMA(3,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(3,1,2)(0,1,0)[12]                    : 403.3915
##  ARIMA(4,1,0)(0,1,0)[12]                    : 396.4012
##  ARIMA(4,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(5,1,0)(0,1,0)[12]                    : 400.6542
##
##
##
##  Best model: ARIMA(0,1,0)(0,1,0)[12]
```

```
#d=1 calculates the first difference(same as above DY), D=seasonal difference
#stepwise by default tries to things as fast as possible
#approximate approximates AIC than exactly calulating AIC
#Trace shows all tryouts
print(summary(fit_arima)) # the model fitted very well with sigma= 12125.05 while AIC c
```

```
## Series: training_ARIMA
## ARIMA(0,1,0)(0,1,0)[12]
##
## sigma^2 estimated as 1.47e+08:  log likelihood=-194.79
## AIC=391.59   AICc=391.84   BIC=392.48
##
## Training set error measures:
##                    ME     RMSE      MAE      MPE     MAPE      MASE       ACF1
## Training set 2816.041 9239.295 4600.965 1.902115 3.385625 0.1559418 -0.3151891
```

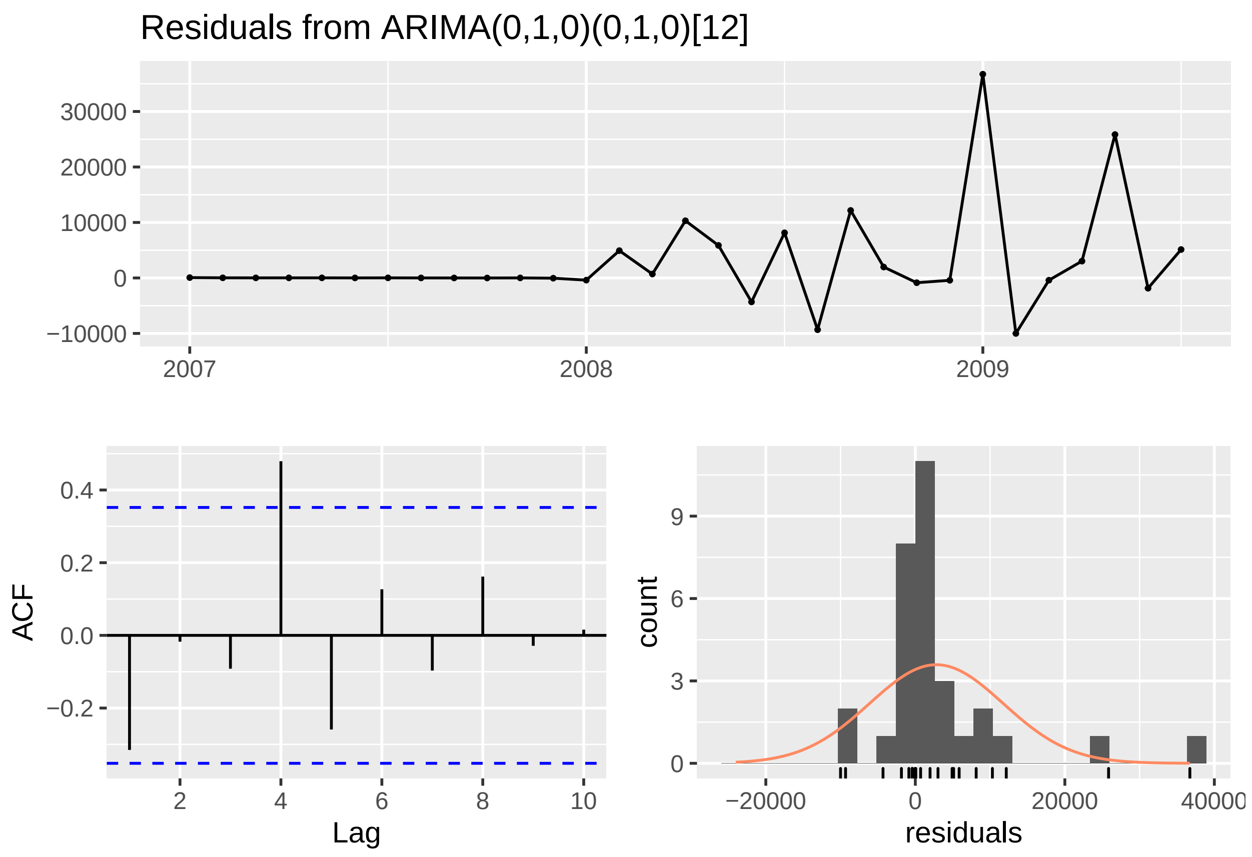
```
##                    ME     RMSE      MAE      MPE     MAPE      MASE       ACF1
## Training set 2816.041 9239.295 4600.965 1.902115 3.385625 0.1559418 -0.3151891
```

```
plot.ts(training_ARIMA,main = "ARIMA Timeseries Fit of Contoso data", col = "blue")+
  lines(fitted(fit_arima),col = "red")
```



```
## integer(0)
```

```
checkresiduals(fit_arima) # the residual was significant with a value of 0.01542 which
```

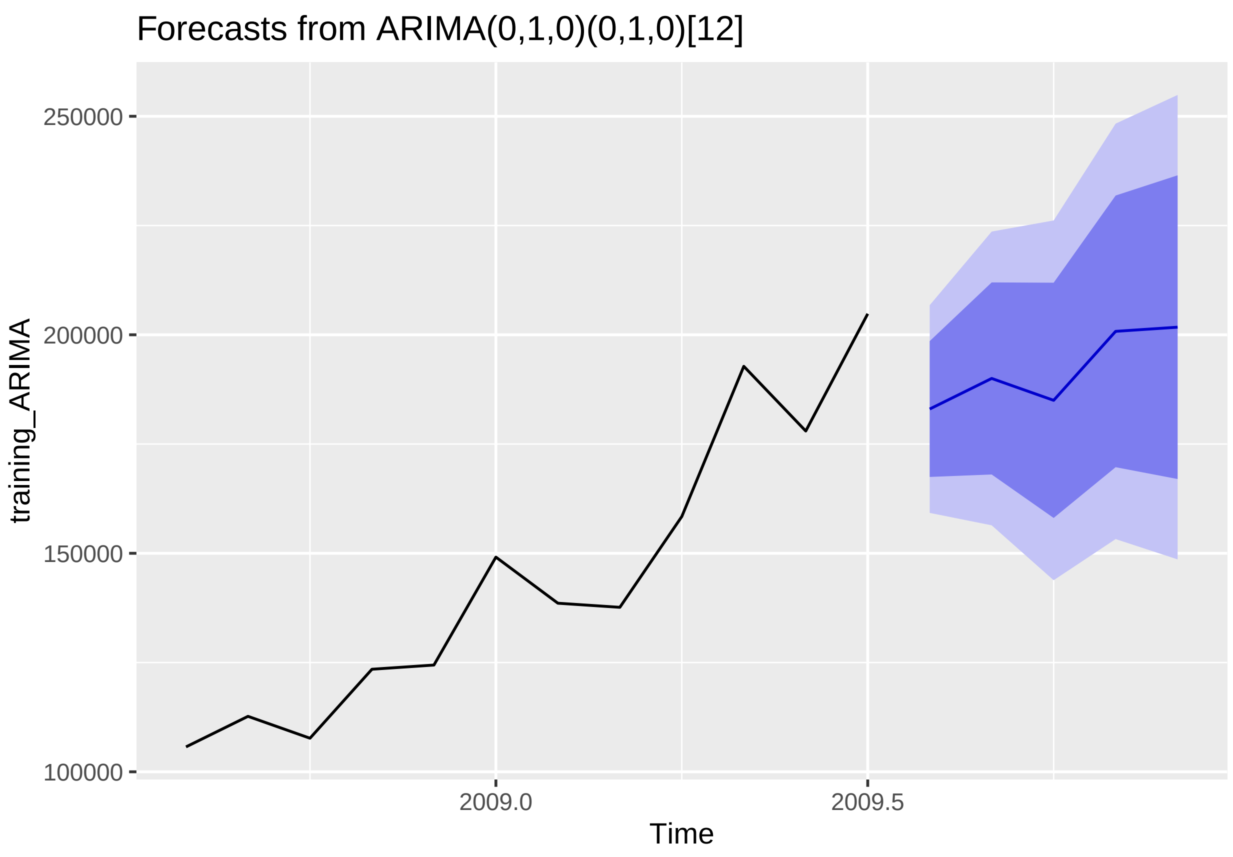


```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,0)(0,1,0)[12]
## Q* = 15.706, df = 6, p-value = 0.01542
##
## Model df: 0.   Total lags used: 6
```

```
# It returns the sigma squared, if we do a square deviation , it is 560267.8
sqrt(fit_arima$sigma)
```

```
## [1] 12125.05
```

```
# Now finally forcast

fsct<- forecast(fit_arima, h=5)
autoplot(fsct, include=12) # includes just 180 months of data
```

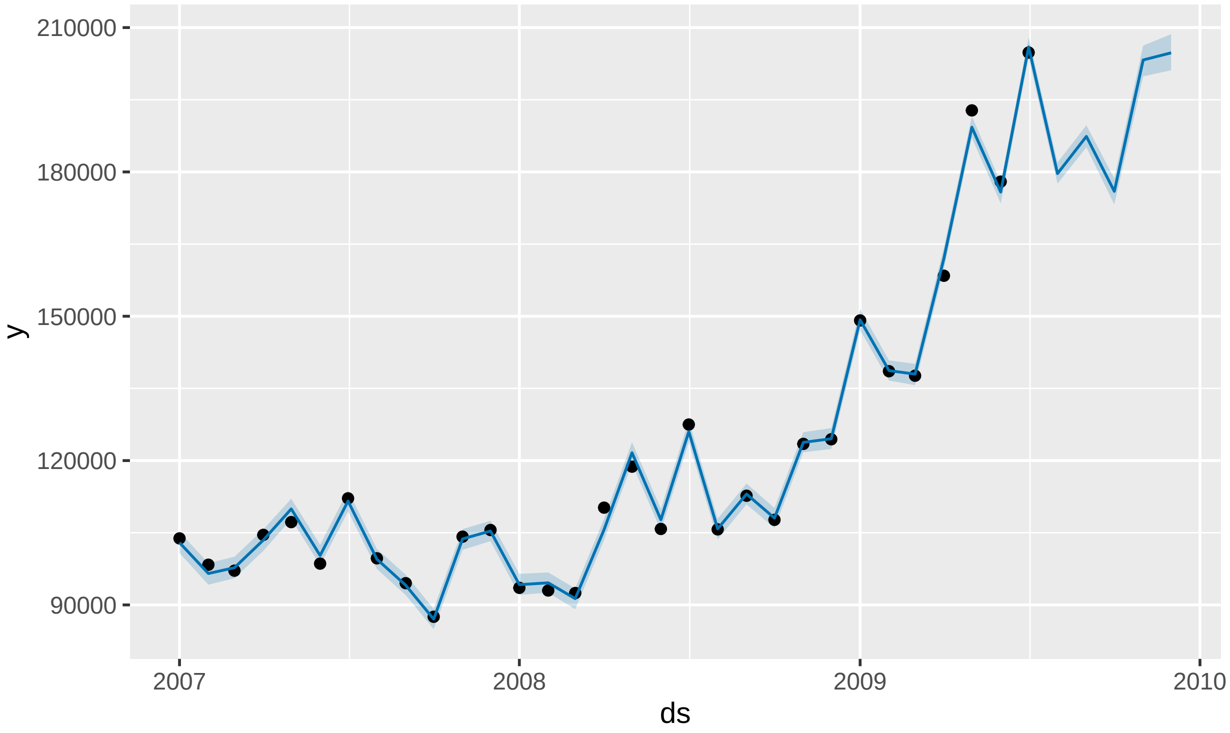


```
#export<-summary(fsct)

#summary(fsct)

# Accuracy
accuracy(fsct$mean, testing_data) # Model had the least error so far with MAE of 13772.
```

```
##                   ME     RMSE     MAE       MPE     MAPE       ACF1 Theil's U
## Test set -13216.8 17150.05 13772.4 -7.783378 8.056898 -0.1792573  1.208411
```

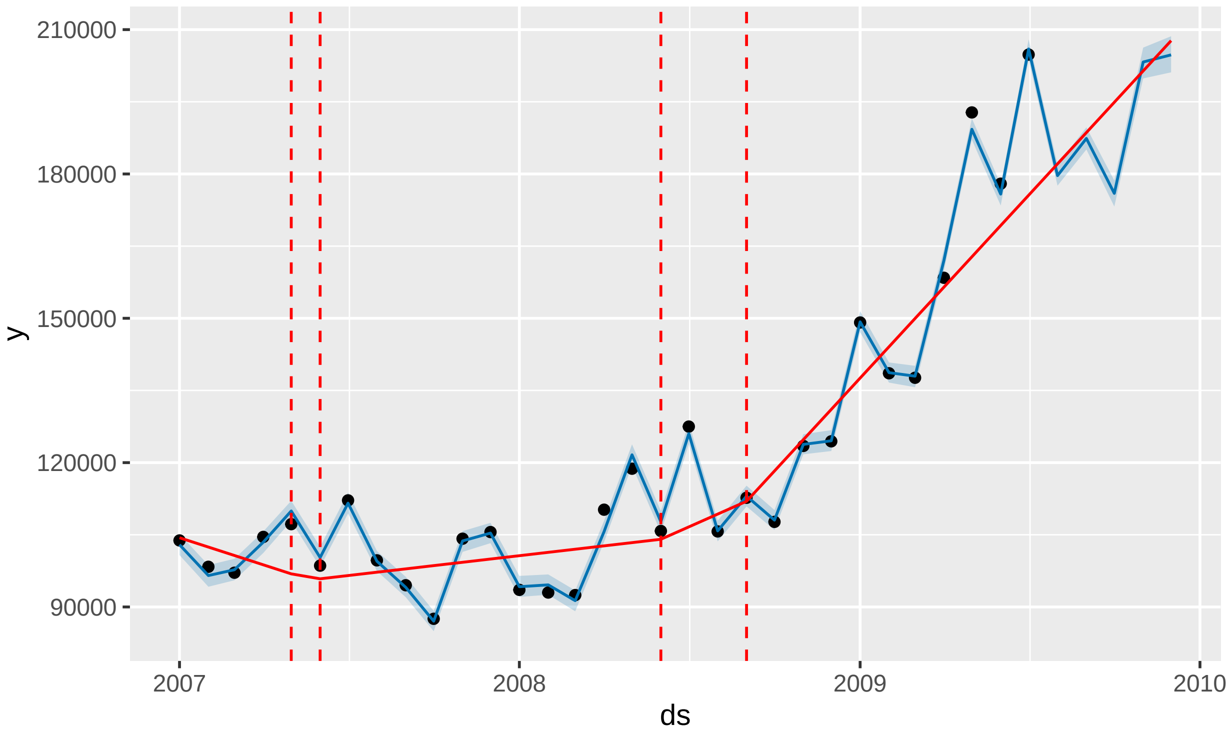
```r
# Now lets use Facebook prophet
#stan background- probabilistic programming language for statistical interference
# Dynamic regressors is a feature added from other models
#Built in cross validation
test_set<-subset(Qtysoldbydate_df, Qtysoldbydate_df$YearMonth>'2009-07-01')
training_set<-subset(Qtysoldbydate_df, Qtysoldbydate_df$YearMonth<='2009-07-01')
df<- training_set # df is to be named for prophet
colnames(df)[1]<-"ds" # we need to do ds for prophet
colnames(df)[2]<-"y"
#tail(df)

## prophet
#install.packages("prophet")
library(prophet)
m<- prophet(growth = "linear", yearly.seasonality = TRUE, seasonality.mode = "multiplica

m_p<-fit.prophet(m, df)

#summary(m_p)

## create prediction dataframe

test_period<-make_future_dataframe(m_p, periods = nrow(test_set), freq = "month")

# Prediction

prophet_forecast<-predict(m_p,test_period)
```

```
#tail(prophet_forecast[c("ds","yhat","yhat_lower", "yhat_upper")])

forecast_ts<-prophet_forecast%>%select("ds", "yhat")
forecast_prophet_partial<-subset(forecast_ts, forecast_ts$ds>'2009-07-01' & forecast_ts$
forecast_prophet_subset_ts<- ts(forecast_prophet_partial[,2], start = c(2009,8),end=c(20

#plotting

plot(m_p, prophet_forecast)
```

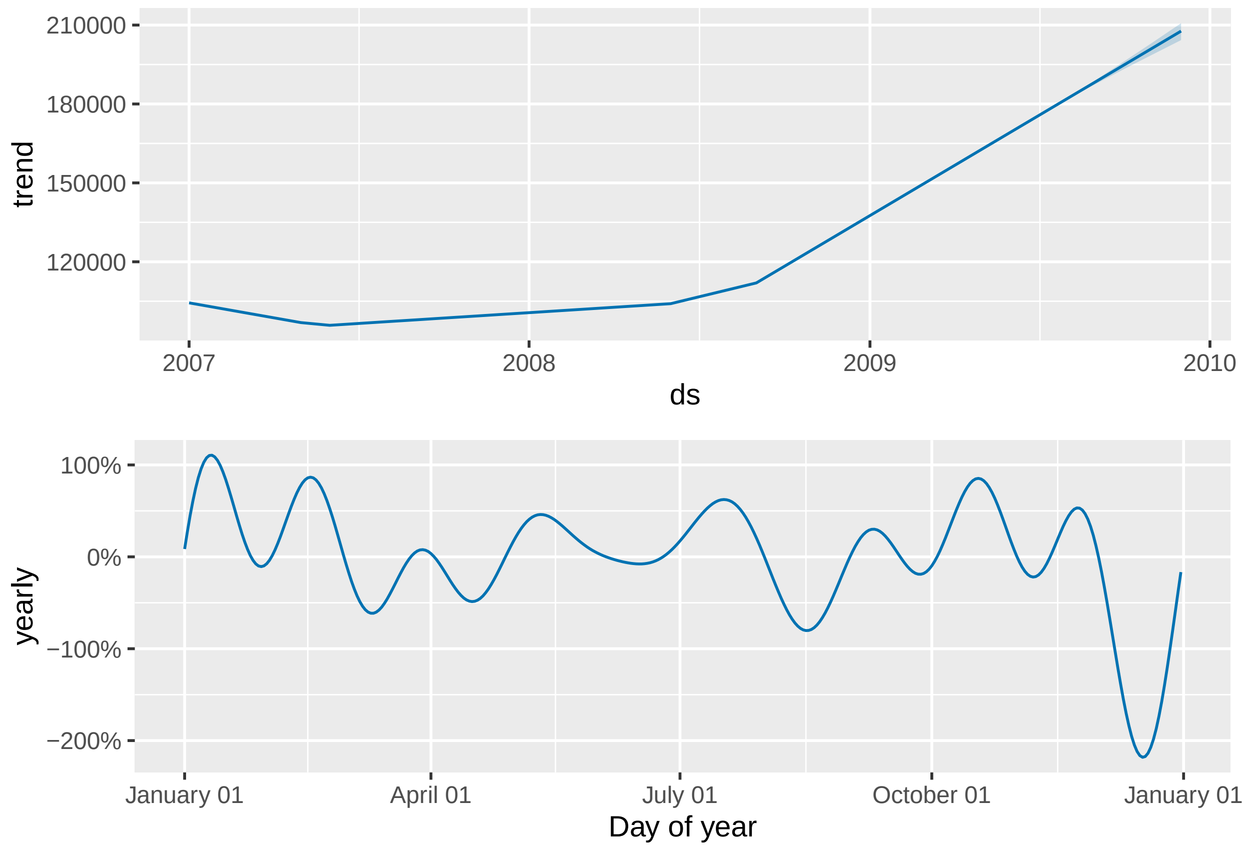


```
#dynamic plot
#dyplot.prophet(m_p, prophet_forecast)
```

```
plot(m_p, prophet_forecast)+add_changepoints_to_plot(m_p)
```



```
prophet_plot_components(m_p,prophet_forecast)
```



```
#accuracy
predictions_prophet<-tail(prophet_forecast$yhat, nrow(test_set))
accuracy(predictions_prophet, test_set$TotalQty)
```

```
##                  ME     RMSE      MAE       MPE     MAPE
## Test set -11312.34 15543.09 12059.73 -6.582227 6.993954
```

```
# This model is better than ARIMA where mae is 12059.73 and MPE is-6.582227%


#But we can use the cross validation to make the model better


### cross validation
#Training data (initial): The amount of data set aside for training. The parameter is
```

```
#Horizon: The data set aside for validation. If you don't define a period the model wi
#Cutoff (period): a forecast is made for every observed point between cutoff and cutof

#cutoffs_input <- seq.Date(from=as.Date('2009-07-01'), to=as.Date('2009-12-01'), by="m
cv<-cross_validation(m_p, horizon =5, units = 'days', initial = 800) #, period = '31 day

#results of cv after cross validation

accuracy(cv$yhat, cv$y)
```

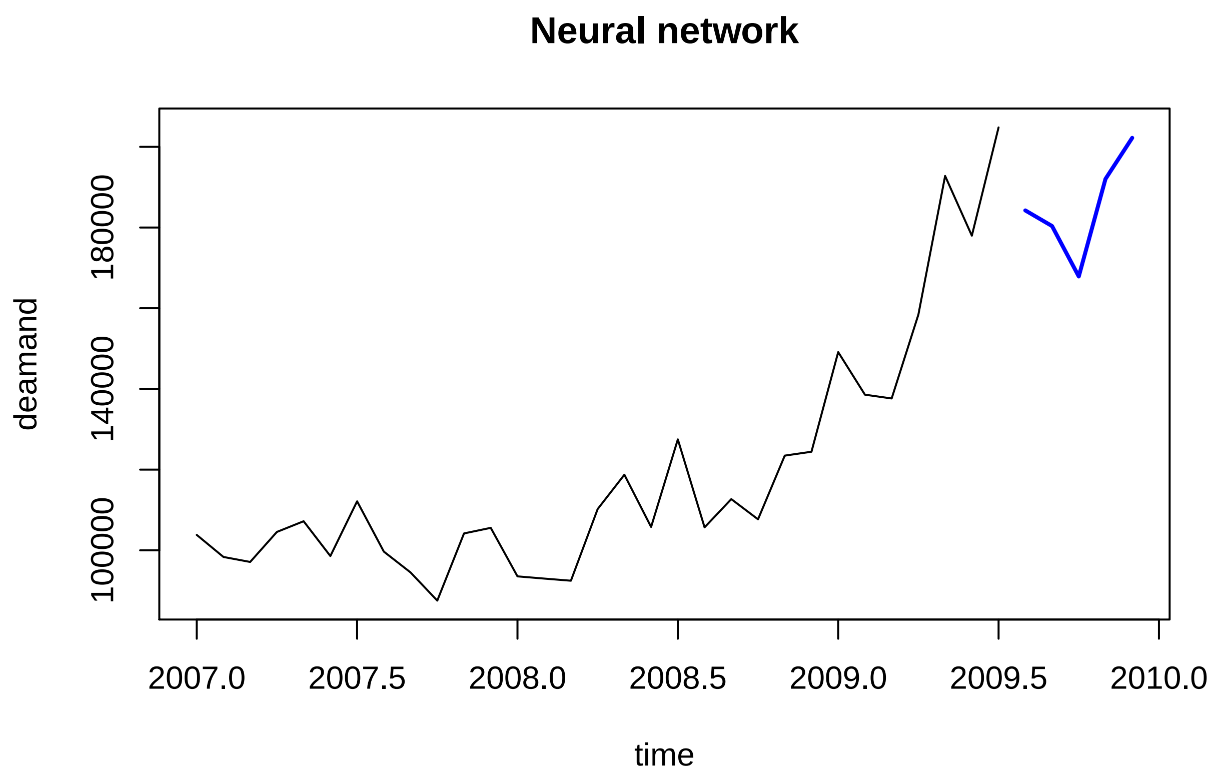
```
##                 ME     RMSE      MAE      MPE     MAPE
## Test set 4120.74 11233.62 9399.411 2.216871 4.963418
```

```
# After crossw validation model got a better accuracy , MAE was 9399.411 and MPE=2.216

##Neural network Autoregressions
#timeseries is needed for neural network and external regressors can be used
training_nnt <- window(Qtysoldbydate,start=2007,end=c(2009,7), frequency=12)
testing_nnt <- window(Qtysoldbydate, start=c(2009,8),end=c(2009,12), frequency=12)
#from forecast package use nnetar
nnetar_model<-nnetar(training_nnt, decay=0.05, P=2 )#  P is for seasonal lags
sqrt(31973675)
```

```
## [1] 5654.527
```

```
## prediction
prediction_nn<-forecast(nnetar_model, h=5)
```

```
#plotting NN

plot(prediction_nn, ylab = "deamand", xlab = "time", main="Neural network")
```



```
#accuracy


accuracy(prediction_nn$mean, testing_nnt)
```

```
##                   ME     RMSE      MAE      MPE    MAPE       ACF1 Theil's U
## Test set -6455.35 8892.19 6818.323 -3.74798 3.92667 -0.6529511 0.6272528
```

```
# The accuracy of the model are MAE= 14130.27 and MPE=-8.213749%, This model is no as


#Neural network is notgood for predicting trends generally.
```

```
#Putting it all together
```

```
autoplot(testing) +
  autolayer(testing, series="Test_actual", PI=FALSE) +
  autolayer(mean_model_predict$mean, series="Mean", PI=FALSE) +
  autolayer(naive_model_predict$mean, series="Naïve", PI=FALSE) +
  autolayer(snaive_model_predict$mean, series="Seasonal naïve", PI=FALSE) +
  autolayer(predictions_ets$mean, series="Exponential Smoothing", PI=FALSE) +
  autolayer(fsct$mean, series="Sarimax", PI=FALSE) +
  xlab("Month") + ylab("Qtysold") +
  ggtitle("Forecasts for Qtysold with simple, seasonal Averages and ETS") +
  guides(colour=guide_legend(title="Forecast"))
```

```
## Warning: Ignoring unknown parameters: PI
```

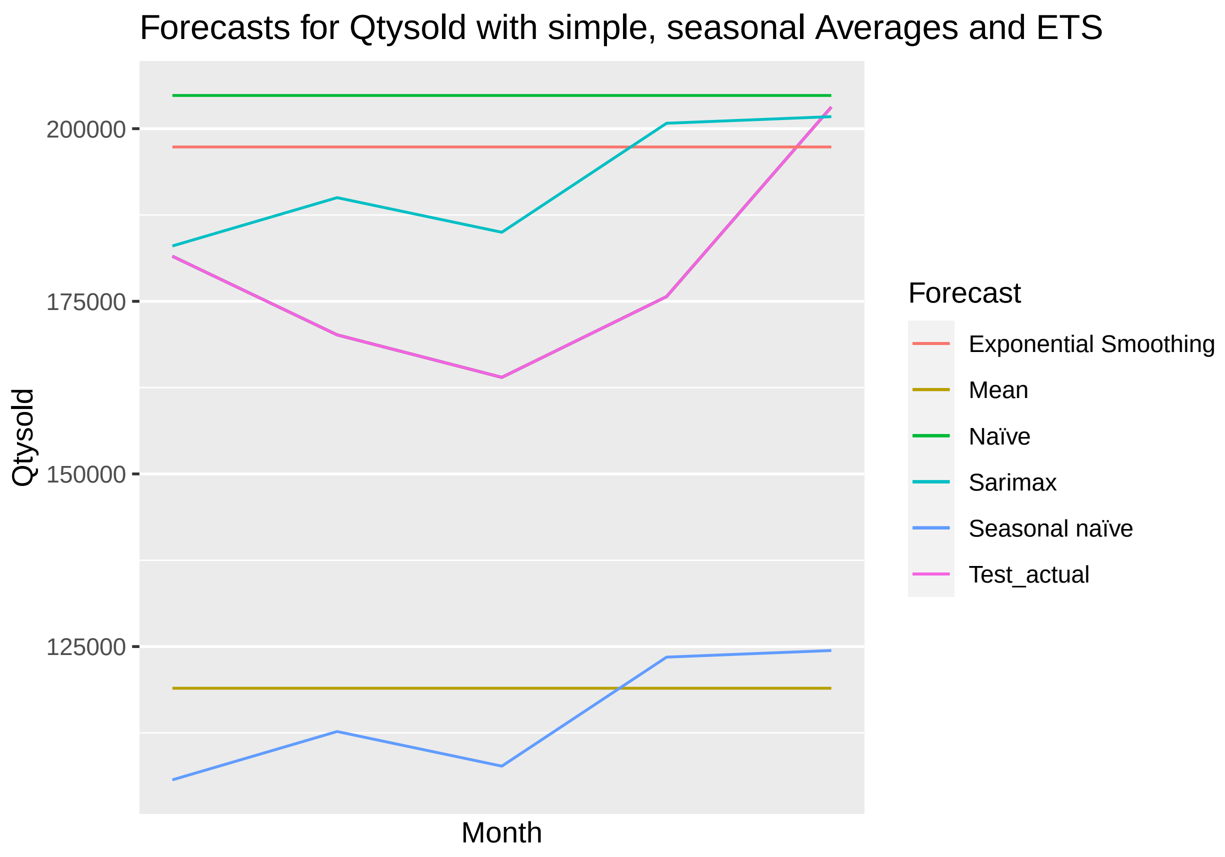
```
## Warning: Ignoring unknown parameters: PI
```

```
## Warning: Ignoring unknown parameters: PI
```

```
## Warning: Ignoring unknown parameters: PI
```

```
## Warning: Ignoring unknown parameters: PI
```

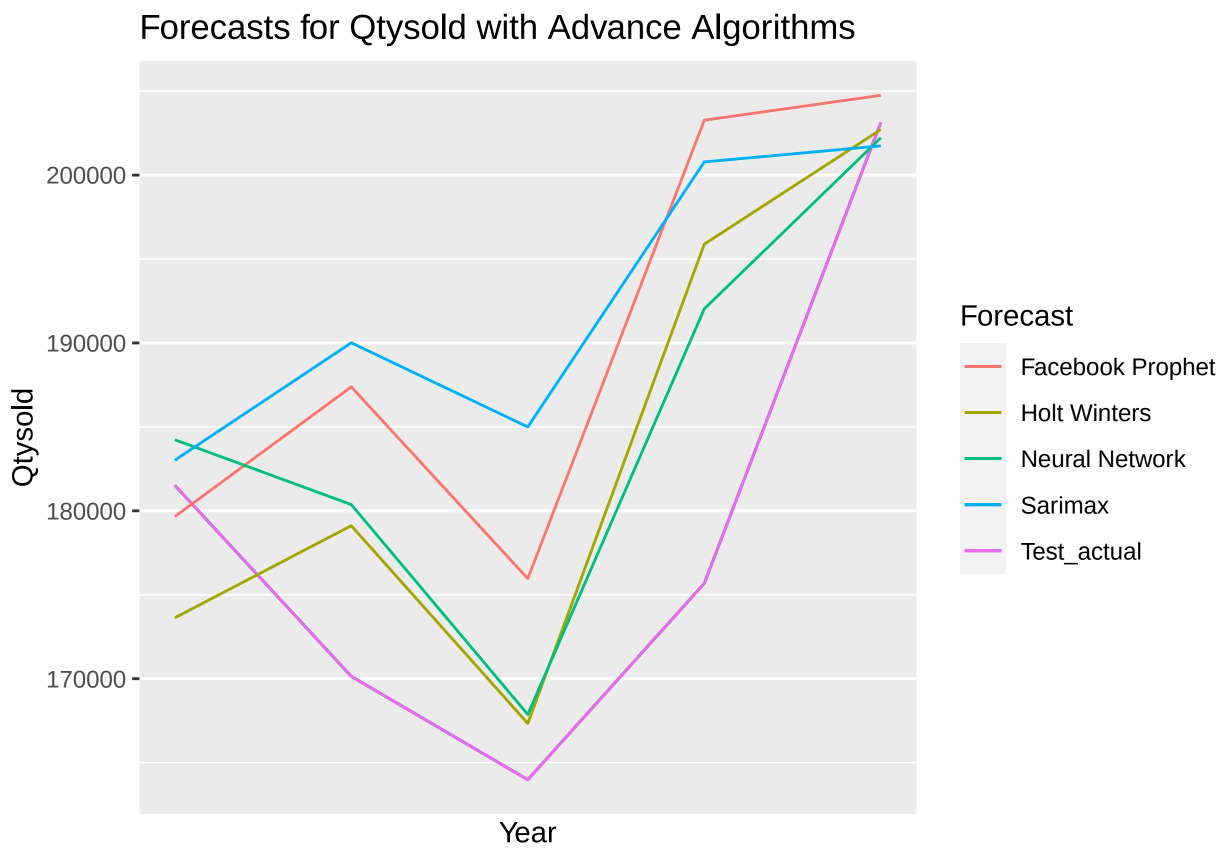
```
## Warning: Ignoring unknown parameters: PI
```



```
autoplot(testing) +
  autolayer(testing, series="Test_actual", PI=FALSE) +
  autolayer(forecast_prophet_subset_ts, series="Facebook Prophet") +
  autolayer(predictions_hw$mean, series="Holt Winters", PI=FALSE) +
  autolayer(prediction_nn$mean, series="Neural Network", PI=FALSE) +
  autolayer(fsct$mean, series="Sarimax", PI=FALSE) +
  xlab("Year") + ylab("Qtysold") +
  ggtitle("Forecasts for Qtysold with Advance Algorithms") +
  guides(colour=guide_legend(title="Forecast"))
```

```
## Warning: Ignoring unknown parameters: PI
```

```
## Warning: Ignoring unknown parameters: PI
## Warning: Ignoring unknown parameters: PI

## Warning: Ignoring unknown parameters: PI
```


Forecasts for Qtysold with Advance Algorithms
Qtysold
Year
Forecast
Facebook Prophet
Holt Winters
Neural Network
Sarimax
Test_actual