Case Study

Time Series Forecasting for Sales

Industry – Retail

Learning Outcomes:

After studying this case, the students will be able to:

1. Elucidate time series and its main components.
2. Understand the requirements and challenges of forecasting for a consumer market organization.
3. Apply various forecasting techniques of time series models.
4. Interpret and decide the output.

Getting started with the case

The case study discusses how KPMG India helped a consumer market organization create a sales forecast mechanism that can make weekly forecast for sales up to 12-weeks. The forecast had to be done for an organization which had hundreds of products in multiple warehouses across the country. Each warehouse stocked hundreds of products**.** According to the initial requirement, KPMG India understood that this case requires multiple time series models with each one of them having different data ranges, different trends, different seasonality spikes etc. The number of items for forecast was quite large and the model was supposed to be able to update, train and predict the values every week within a given time frame. It needed a system which was capable enough to automatically detect spikes and follow business rules to avoid over stocking or understocking.

Note: Before jumping into the scenario of the case, we would first discuss 'what is Time Series Forecasting' in order to help you understand the reasons behind using one forecasting technique over the other.

Introduction to Time Series Analysis

What is Time Series forecasting?

Time series forecasting has become one of the most important application of predictive analytics in an organization. Therefore, we need to do proper planning and preparation of both range and long range. This had a direct impact on both the top and bottom lines in the organization. Forecasting in general, needs to be done for budget allocation – marketing promotion, advertisement, manpower planning, capacity planning, material requirement planning etc. Proper forecast is important as it can have a direct impact on the on the profit and revenue.

Time series forecasting might appear simple but is one of the most complex predictive analytics techniques. Forecasting can become challenging due to many factors which influence the demand and the scale of the business. Forecasting for e-commerce and manufacturing becomes more challenging as the SKU turns out to be in thousands-lakhs. For instance, amazon sells more than 300 million products through its e-commerce portal. Amazon itself sells about 13 million SKU's and has more than 2 million retailers selling their products. Predicting demand of these products is an important science. Overstocking will have a direct impact on the bottom line and under stocking can cause customer dissatisfaction.

Time Series Data and Components of Time Series Data

A Time Series Data is a data on a response variable Y(t). The data points are usually collected at regular intervals and are arranged in a chronological order.

There are two types of time series analysis data:

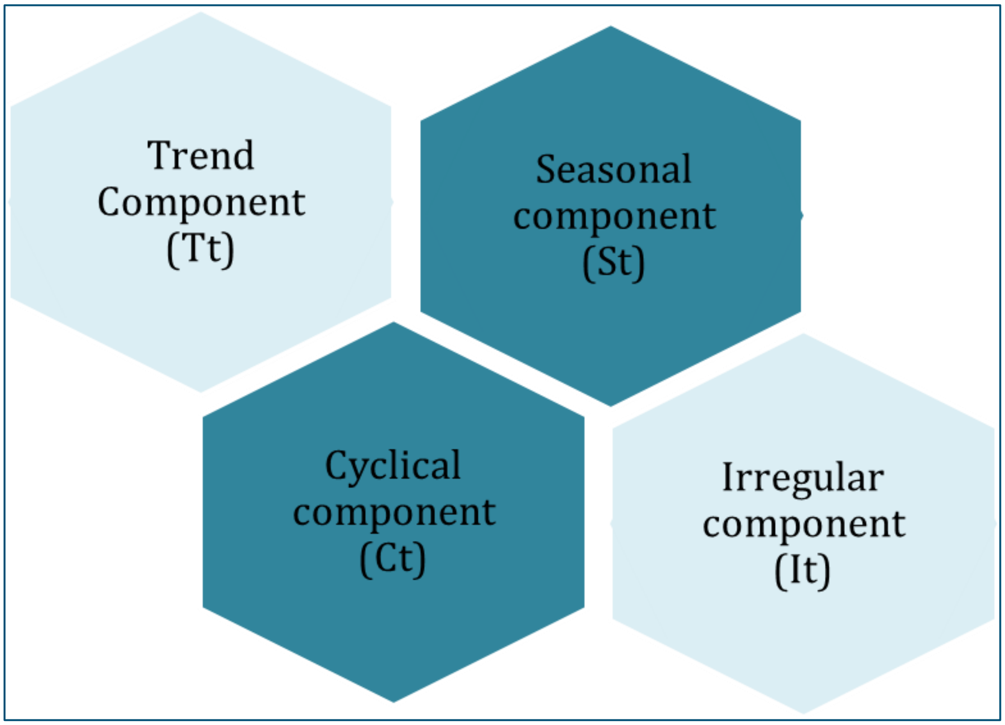
UNIVARIANT

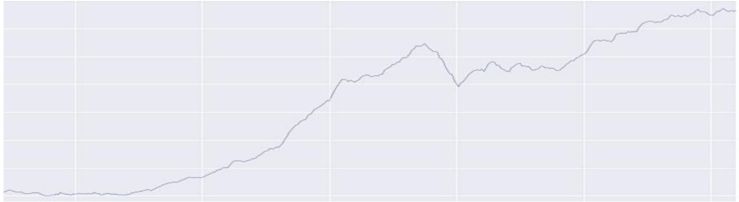
MULTIVARIANT

If the time series data consists of a single variable it is called univariant time series data. Example - demand of a product at time t.

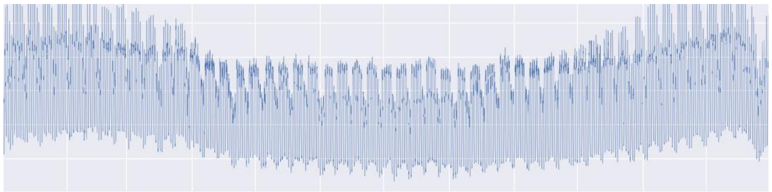
If the time series data consist of more than one variable, it is called multivariant time series data. Example - data about demand, price, money spent on promotion/advertisement at time t.

For a forecast perceptive, a time series data can be broadly broken down into 4 components.

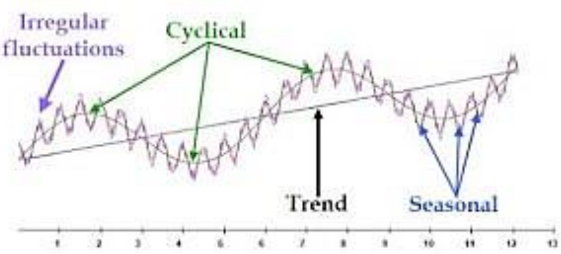




A **trend component** consists of a consistent upward or downward movement of variable w.r.t. time scale.



Seasonal component (measured w.r.t seasonality index) is the repetitive upward or downward movement forming a trend that occurs within a specific calendar year period such as festival, vacations etc. Example - the sales of products might show an upward trend during the month of December and October.



Cyclical component is a trend or fluctuation around the trend line that mostly occurs due to macro-economic changes such as recession. There exists a major difference between seasonal component and cyclical component trend. Seasonal trend occurs within a calendar year whereas the periodicity of cyclical component trend is varying. Irregular component can be noise in the data or uncorrelated changes.

Note: Cyclical component is better to be negated if you have a small or a medium size dataset. For estimating the cyclical component we require large amounts of data.

A time series model can be a result of an addition function of the already mentioned components is called an additive time series model;

$$Yt= Tt + St + Ct + It$$

Similarly, a time series model created using a product function is called a multiplicative time series model

$$Yt= Tt * St * Ct * It$$

Additive time series model cannot be used in a case such as ours as the seasonal component is independent of the trend. Our problem is not of additive nature.

To understand this better, let us look into a simple example; your college provides a specific course on weekends, a fixed number of suppose 60 students are enrolled in it and they are provided with lunch by the college itself. This shows additive nature in data - the demand of food will on weekends by 60+ students- the seasonal component is dependent on the trend.

Let us consider a case of e-commerce; the demand of the product may increase with trend which may or may not be independent of the season. So, the increase will be in multiplicative in nature.

In mathematical terms; the additive model is appropriate if the seasonal component remain mostly unchanged, above the level of local mean and doesn't vary with the level of the series.

For our case multiplicative model becomes more appropriate as the seasonal component is correlated with the level of local mean.

There have been many techniques built based of different logics and approaches ranging from simple techniques like moving average and exponential smoothing to a complex regression-based logic like auto-regressive, auto-regressive moving average (ARMA) & auto-regressive integrated moving average (ARIMA). ARIMA has become the rock star of time series analysis or forecasting and everyone wants to learn and implement it. But that's not how everything works! Using a complex logic will not always guarantee you better results, sometimes simple moving average techniques provide better results than complex techniques like ARIMA. Multiple forecasting techniques like moving average, exponential smoothing and ARIMA should be used and checked for better accuracy before jumping to a final model. But the assumption of stationary process remains intact.

Please refer https://otexts.com/fpp2/expsmooth.html to learn more about these techniques and different logics.

Refer https://www.kdnuggets.com/2019/08/stationarity-time-series-data.html to learn about stationarity of data.

Why did we go for ARIMA over other forecasting techniques?

We used ARIMA model and not ARMA model as ARMA model can only be used for stationary data whereas ARIMA model can be used for non-stationary data. We surely need to convert the non-stationary data to stationary one but this can be done by directly calling the components of ARIMA i.e. ARIMA(p,d,q)

- P Auto regressive component with p lags, AR(p)
- Integration component (d)
- Moving average with q lags, MA(q)

The integration component converts the non-stationary time series data to stationary time series data so that AR and MA process can be used for forecasting. On using a forecasting technique directly the non-stationary data it was observed that the ACF showed a very slow decline which on further R&D was because the model parameter values were greater than the one resulting in non-convergence of the time series causing the ACF value to decline. In our case the non-stationarity was a result of stochastic trends. The de-trending can be done either by using trend stationarity or difference stationarity process. It is always recommended to find the nature of your time series data using either ACF plot, Dickey-Fuller test etc.

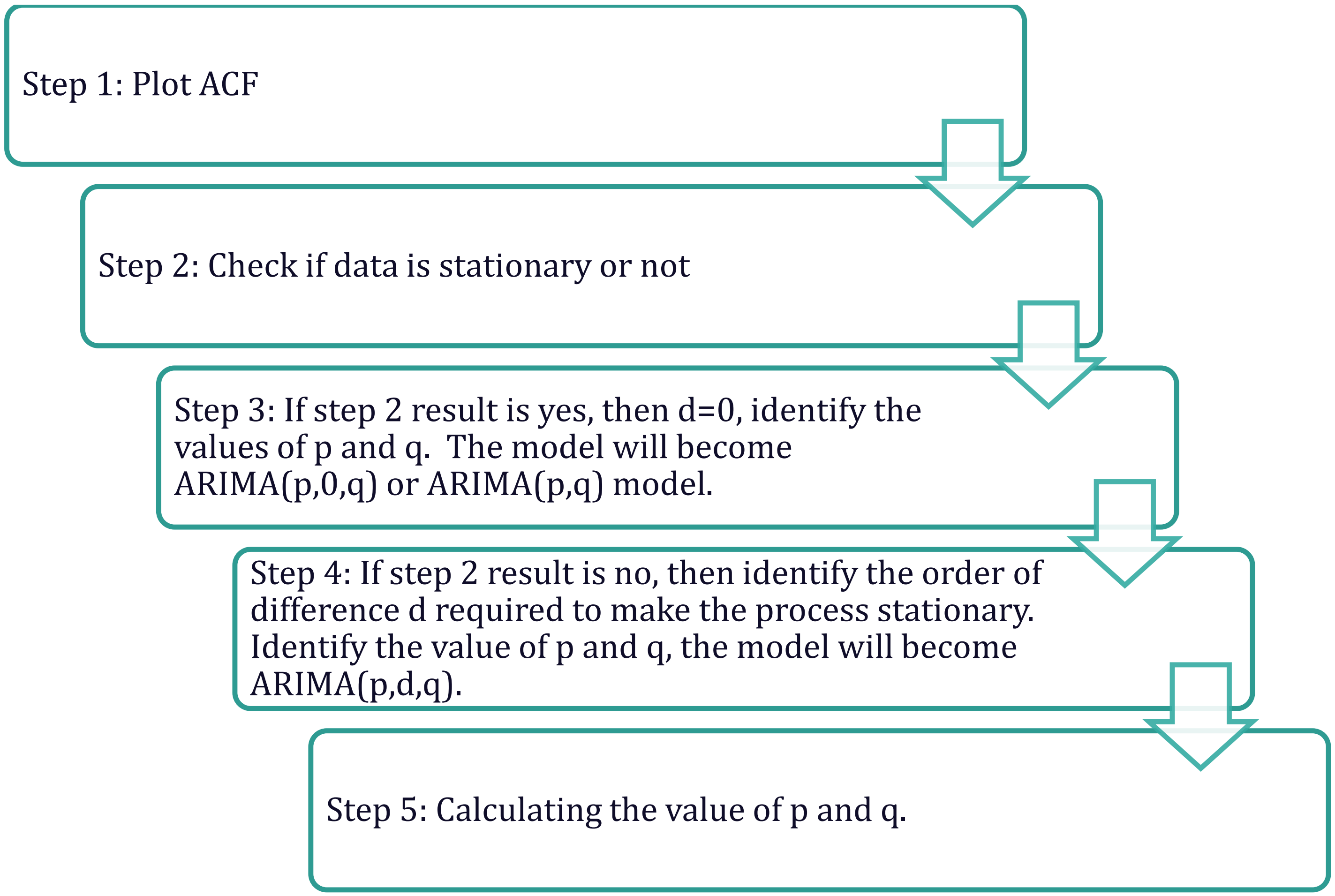
Phases of Time Series Model Development

We divided our time series model development into three phases; model identification, parameter estimation- model selection and model validation. These phases also help calculate the ARIMA (p,d,q) values.

Phase 1: Model Identification

Our objective was to identify the value of auto regressive lags, order of differencing moving average lags. This figure will help identify the stages. The entire process starts with plotting ACF, again saying you should never skip the step of plotting the data. This step will help you identify whether the time series data provided is stationary or not. In our case the data was not stationary therefore we had to find the value of d. If the data would have been stationary the value of d would have become 0.

The flow chart stating this condition is given below:



Step 1: Plot ACF

Step 2: Check if data is stationary or not

Step 3: If step 2 result is yes, then d=0, identify the values of p and q. The model will become ARIMA(p,0,q) or ARIMA(p,q) model.

Step 4: If step 2 result is no, then identify the order of difference d required to make the process stationary. Identify the value of p and q, the model will become ARIMA(p,d,q).

Step 5: Calculating the value of p and q.

Phase 2: Parameter estimation and model selection

After step 5 we have the value of p, d and q, next step we need to do is estimation of the AR and MA. This was achieved using ordinary least square. After model development the model selection is needed to be done, using RMSE, Akaike Information criteria or Bayesian Information Criteria. AIC and BIC were calculated using log likelihood and lesser the value better is the model.

Phase 3 Model Validation

Model validation of an ARIMA model is similar as for any regression technique. We need to make sure the only residual in the model is the white noise, we also went one step ahead to test the model using Ljung Box test. You can also perform Theil's coefficient to calculate the power of the forecasting model.

Refer to https://robjhyndman.com/hyndsight/ljung-box-test/

Model Design and Development

On understanding the requirement; we split the processing engine into 3 smaller independent modules. The final design model was a hybrid model design using both statistical and heuristic techniques and can be split into the following components:

Series Spike Analyzer

Regression-Time Series Model

Safety Stock - Rule Engine

Spike Analyzer

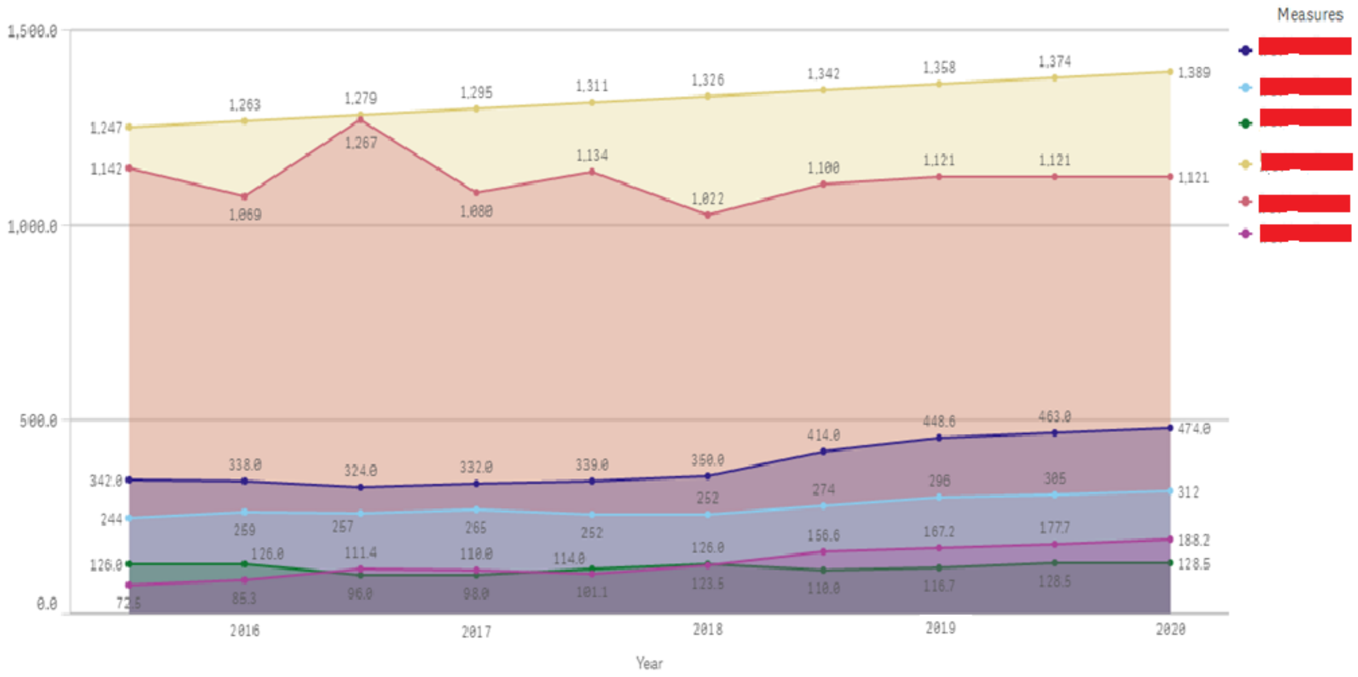
It was designed to raise alerts if there would be a skipped demand of products, initially it was just designed for forecasting seasonality trends based on historical data. The time period was broken into multiple buckets such that each bucket has 45 days in itself (this is a standard approach in the industry). Based on some business logic; this creation of buckets is done so that we can we can easily track in which bucket a higher demand is expected.

To explain this better, let us take an example of the e-commerce industry. If there is a Diwali or Christmas sale, the items on discount or new products in the market will be different. The spikes will be different for different product category. If there comes in Great Indian sale on e-commerce site, the products bought will be different. These buckets are allocated different priorities and have different product categories in them. There are some ground rules in order to raise an alert: the demand spike must have occurred in multiple years; and it must have occurred in the year corresponding to the most recent previous season. These two rules lay a ground for marking the spikes.

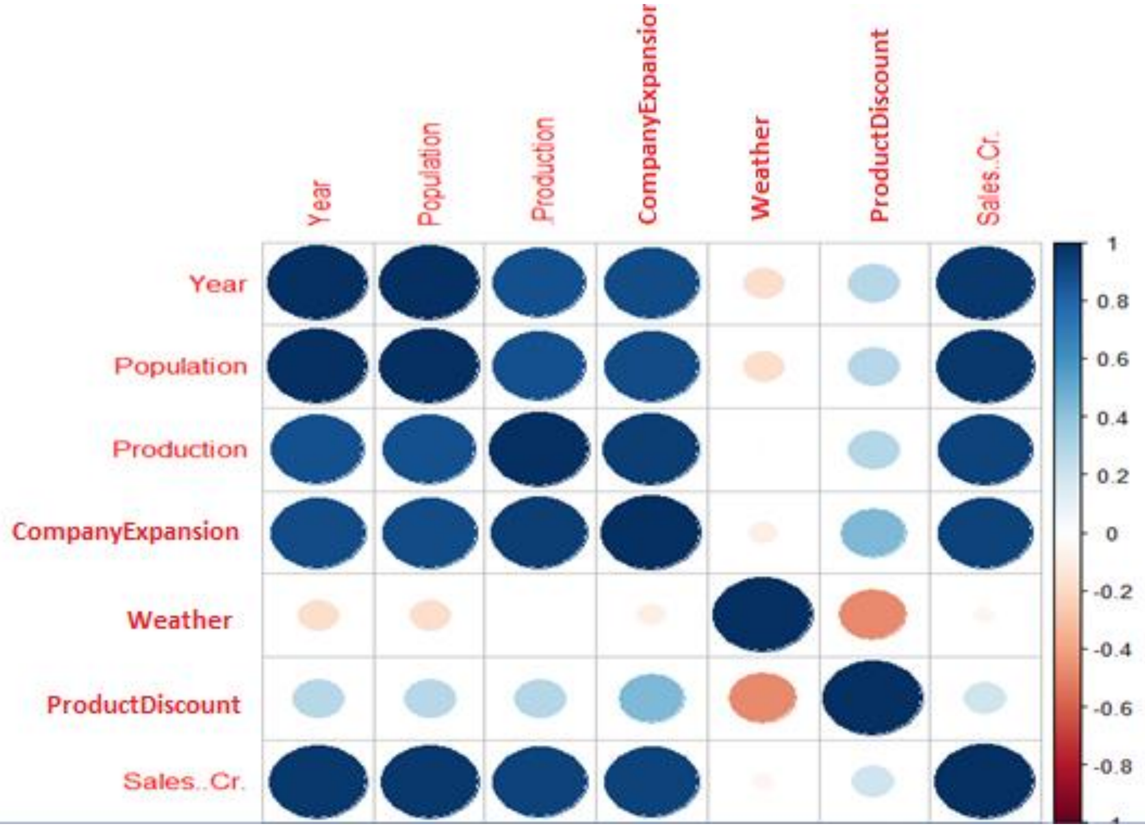
There comes another challenge - many festivals like Holi, Eid, Diwali, Christmas do not come on the same date. There exist chances that some of these spikes might not fall into the same bucket. We had to add a feature which maps holidays/sales with buckets. This was a long and painful process to map historical data, but this cannot be missed.

This is the trend analysis for the reliable products of the client. In this image we can see we divided every individual year into 2 buckets. Client usually like to treat their most reliable products or the product categories which have not seen any alterations over the years separately.

Following is the trend analysis of these products, these products have mostly similar pace of sale.



In the next figure you can see the correlation of overall variables:



Regression-time series model

With so many product categories we had to create subsets of product demand time series initially for analysis. With so many description, categories and seasonal events the accuracy of the model gets directly hampered. In order to increase the accuracy of our model we had to go along using mini dummy variables. These variables are usually having categorical values to provide the presence of an event which can directly impact the demand of the products. This surely means increase in the processing time. It is really hard to miss the limitations of seasonal differencing and mini seasonal dummy variables for always shifting seasonal events. We came up with a solution by synchronizing values corresponding to the same event. Which resulted in creation of a time series where one year's value is directly altered to match those of another year in such a way that it guarantees values associated with a particular event are "in-sync". Meaning the two data points fall in the same time period interval.

To make it simpler; if the data point of an event of this year and the data point for the same event of previous year are not in an association, then we will have to scan the previous year data points until we are able to find the data point associated with the event. After finding it replace the previous year data point value with the value of the event associated data point. Now what will you do for a scenario when we are not able to find the data point associated with a particular event in the previous year? In such a case we try finding the two nearest data points of the previous year and calculate the average value of the two non-event data points.

Safety Stock Rules Component

This is essential to make sure that the products do not overflow in the warehouse or go out of stock especially when their demand comes in. The safety rule component contains multiple rules designed by the SME and businesspeople to make sure the minimum number of safety stock is maintained.

This system used multiple safety rule such than in situations like up trend, low trend and recent demand the product does not go out of stock in the warehouse at least.

Note: In the e-commerce industry an unusual circumstance is not unusual – sudden drop below a significant level is very common. This is what drops the accuracy of the model in real world and no one can stop it. The extent of loss can be reduced only when there is a strong research team. No predefined rules can avoid such a scenario, you should always explain the scope and limitations to the client beforehand.

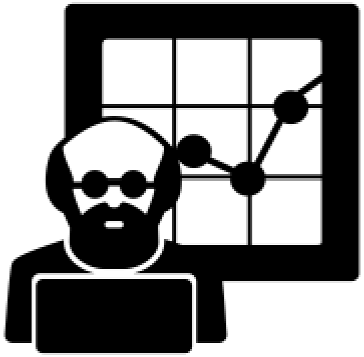
There is a perception in the market that data science can solve anything in the real world, this is wholly not true. Take the case of medical mask shortage!

No data science model could have predicted the seriousness of COVID initially. Even if the warehouses had maintained the minimum stock, they couldn't have predicted the rise in demand.

Now back to the safety rule component; the base forecast by the ARIMA model is processed by this component. The forecast is done without manual intervention. But the safety rule component required manual intervention, we can't let everything be controlled by the algorithm. This component had inbuilt rules, but we had to develop this component such that any new rule can be easily created, and the existing ones can be tweaked easily.

Project Development & Challenges

Before deciding the number of people to be put onto this project we had to send a data scientist, a senior developer resource and a manager to understand the complexity of the problem before stating the timeline.



A real-world project is very different as compared to the projects done in the academic course. You have to sit multiple times with the client representatives and SME to understand what is expected before even drafting a proposal. During the client meeting, the designing of the system had already begun. Splitting the expectation into modules, what output should be expected from which module, all the minute details are mentioned and placed in which module, etc.

Once the designing of the system is completed, we begin to estimate the human effort that will be required by each profile. For the first module we required 2 data scientists, 1 data engineer and 1 manager/lead. This prototype was provided a 3-month timeline. You might have gone through multiple projects where different modules are worked upon by different teams simultaneously but in situation like ours it becomes very tough as the output of a time series project. This project has 3 component and each component is dependent on one or the other output. The output format was not fixed therefore we started the design and development of the first module where we detected spikes- demand on testing and validating the results by the SME's we moved forward with the forecasting technique.
If we would have simultaneously started with the development of multiple modules together, we might have ended redoing the second and third module if the SME had not accepted the results of the first module. That way engineers are asked to design a POC before moving forward with the prototype.

There was a different set of challenge for the second module. It was the volume of data available.

There is a saying:

"Keep in mind the curse of dimensionality"

It's a situation when the number attributes are so many and the total number of records for training the data is not sufficient enough.

With so many products, categories, description, SKU data, labelling, and mini dummy variables for each event, the number of variables can easily reach to a few hundreds and each variable has its significance in the forecast. Above that we knew we might have to deal with situation as we might have to work with data describing some new product category whose previous records are not known. In such a situation one needs to consult the top line what they would want. There can be many solutions to such a situation;

1. First situation - If the product is similar to the old one or their usage is similar to an existing one then we can merge them together. We could have used clustering but the output was not accepted by the SME.
2. Second situation -If the product is totally new then the SME would want you to skip the forecast and add special rules to deal with it.

Forecasting projects provide a great challenge and opportunity for the developers as these type of projects don't directly go with the standard approach. With such a requirement we need to think outside the box in order to design a solution. Initially, for data engineer and data scientists the project can be tough as the designing and development keeps on rapidly changing. Our developers did face a case when module one got to be redone as the SME was not satisfied with the output. But in such a case you should not lose hope, not all the data science projects have a straight map. The path is complicated. You might have to redo something you developed from scratch multiple times. Take it as an opportunity! You, as a data scientist, will have to do additional research and brainstorming.

Conclusion

Our project/prototype was a success and our clients were very happy with the results. The alert system was able to accurately detect the spikes and provide forecast for the demand with an accuracy of 74% for already existing products.

Please refer: Intelligent techniques for forecasting multiple time series in real-world systems (https://cs.adelaide.edu.au/~zbyszek/Papers/p51.pdf)