

# **TIME SERIES FORECASTING PROJECT --SPARKLING--**

**By Giridharan Velmurugan**

# Contents

<b>Serial No</b>	<b>Title</b>	<b>Page No</b>
<b>1</b>	<b>Problem statement</b>	<b>5</b>
<b>2</b>	<b>Read and plot the Dataframe</b>	<b>5</b>
<b>3</b>	<b>EDA and Decomposition</b>	<b>7</b>
<b>4</b>	<b>Train and test data</b>	<b>10</b>
<b>5</b>	<b>Building Predictive Time-Series models</b>	<b>11</b>
<b>6</b>	<b>Test of stationarity</b>	<b>17</b>
<b>7</b>	<b>Auto Regressive Integrated moving average(ARIMA)</b>	<b>18</b>
<b>8</b>	<b>Seasonal Auto regressive integrated moving average(SARIMA)</b>	<b>20</b>
<b>9</b>	<b>Summary of built models and Performance</b>	<b>22</b>
<b>10</b>	<b>Forecast - 17 months</b>	<b>23</b>
<b>11</b>	<b>Process Summary and Business Insights</b>	<b>23</b>

# List of Images

<b>Sl Number</b>	<b>Title</b>	<b>Page number</b>
<b>1</b>	<b>A simple plot of sparkle wines</b>	<b>6</b>
<b>2</b>	<b>A simple box plot</b>	<b>7</b>
<b>3</b>	<b>Boxplot of Sparkle wines Year on year</b>	<b>8</b>
<b>4</b>	<b>Sparkle wine sales</b>	<b>8</b>
<b>5</b>	<b>Decomposition Additive</b>	<b>9</b>
<b>6</b>	<b>Decomposition Multiplicative</b>	<b>10</b>
<b>7</b>	<b>Sparkling Sales, naïve method, train, test &amp; Predictions</b>	<b>11</b>
<b>8</b>	<b>Regression Train, test and predictions</b>	<b>12</b>
<b>9</b>	<b>Sparkling Sales, Regression, Moving averages 2, 4, 15</b>	<b>13</b>
<b>10</b>	<b>Sparkling, Simple exponential smoothing, train, test &amp; predictions</b>	<b>14</b>
<b>11</b>	<b>Sparkling, Double exponential smoothing, train, test &amp; predictions</b>	<b>15</b>
<b>12</b>	<b>Sparkling, Triple exponential smoothing, train, test &amp; predictions</b>	<b>16</b>
<b>13</b>	<b>Original data, Rolling mean and Std deviation</b>	<b>17</b>
<b>14</b>	<b>Original Data, rolling mean and std deviation - Differenced data</b>	<b>17</b>
<b>15</b>	<b>ACF with Difference of order 1</b>	<b>18</b>
<b>16</b>	<b>PACF with difference of order 1</b>	<b>19</b>
<b>17</b>	<b>Auto Arima Train, test and predictions</b>	<b>20</b>
<b>18</b>	<b>SARIMA Diagnostics</b>	<b>21</b>
<b>19</b>	<b>Sparkling, Forecast into 17 months</b>	<b>23</b>

## List of Tables

SL Number	Title	Page number
1	Year Month/ Sparkling	5
2	Forecast using moving averages trailing table	13
3	Parameter pdq/ Akaike value	19
4	Non-Seasonal 'pdq'/ Seasonal 'PDQs'/ Akaike value	20

## Problem Statement

For this assignment the data of different types of wine sales in the 20<sup>th</sup> century is to be analysed. As an analyst in the ABC Estate Wines, you are tasked to analyse and forecast wines sales in the 20<sup>th</sup> Century.

Data Dictionary –

- 1) Year Month: This feature holds the year and month in the format “YYYY-MM”
- 2) Sparkling: This is the feature that holds the sales units for the corresponding year month

The record of sales starts from “January of 1980” and ends on “July of 1995”. Totalling to 15 years and 7 months of recorded sales.

## Read and Plot the Dataframe

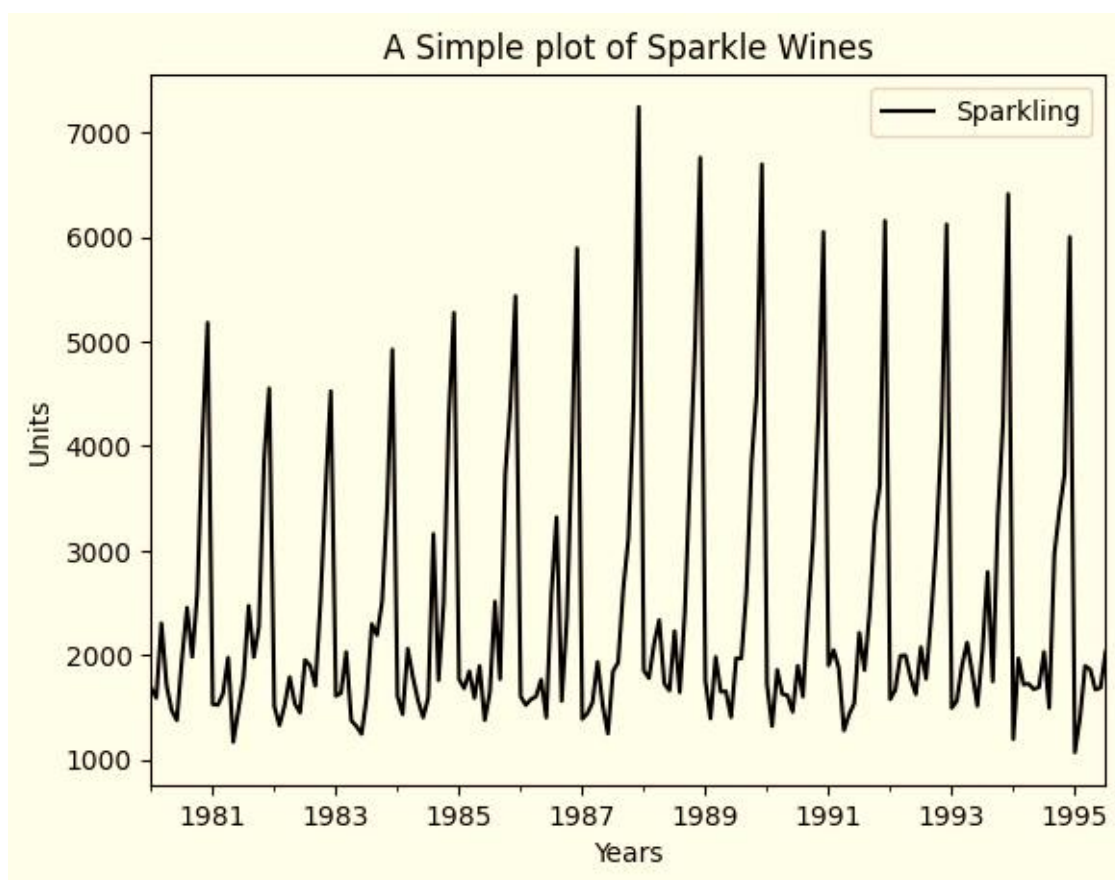
The dataframe has only two features, they are ‘YearMonth’ and ‘Sparkling’ as seen above. The dataframe is read in such a format that the feature ‘YearMonth’ is converted to a yyyy-mm-dd format and set as the index.

This would reduce the said dataframe from two features to one(which would be sales units ‘Sparkling’).

A peek is given in the table below.

YearMonth	Sparkling
1980-01-01	1686
1980-02-01	1591
1980-03-01	2304
1980-04-01	1712
1980-05-01	1471

**Plot of the dataframe.**



**From the plot, we can see that there is seasonality. The peaks and troughs clearly state that.**

**But is there trend in the data? is something which is not very clear, visual inspection suggest there to be a slight upward trend.**

## EDA and Decomposition

The data starts from January of 1980 and goes all the way till July of 1995, totalling 187 months. Fortunately, the data does not have any null values.

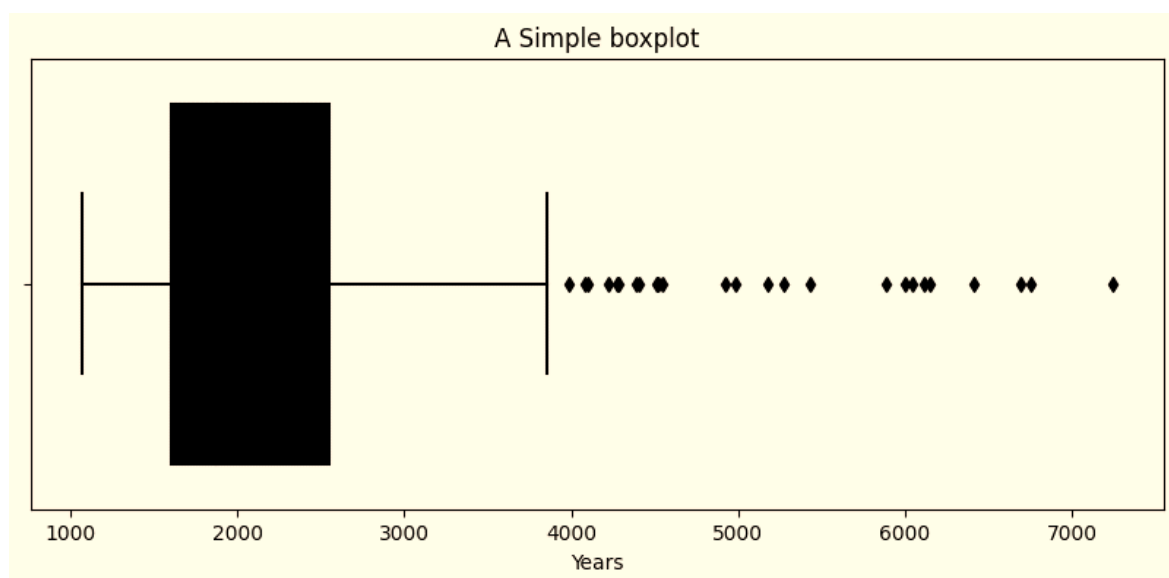
There are some duplicated values, this is nothing to worry as each value is tagged with a different timestamp, making each entry legitimate.

Below is a simple boxplot that is done on the entirety of the dataframe, overall, we do have some outliers, but these can be attributed to the peaks we saw earlier.

The mean sits at 2403 units.

The inter quartile range is between 1605 and 2549 units.

The minimum units sold in 187 months was 1070, and the maximum stands at a staggering 7242 units which is of course an outlier.

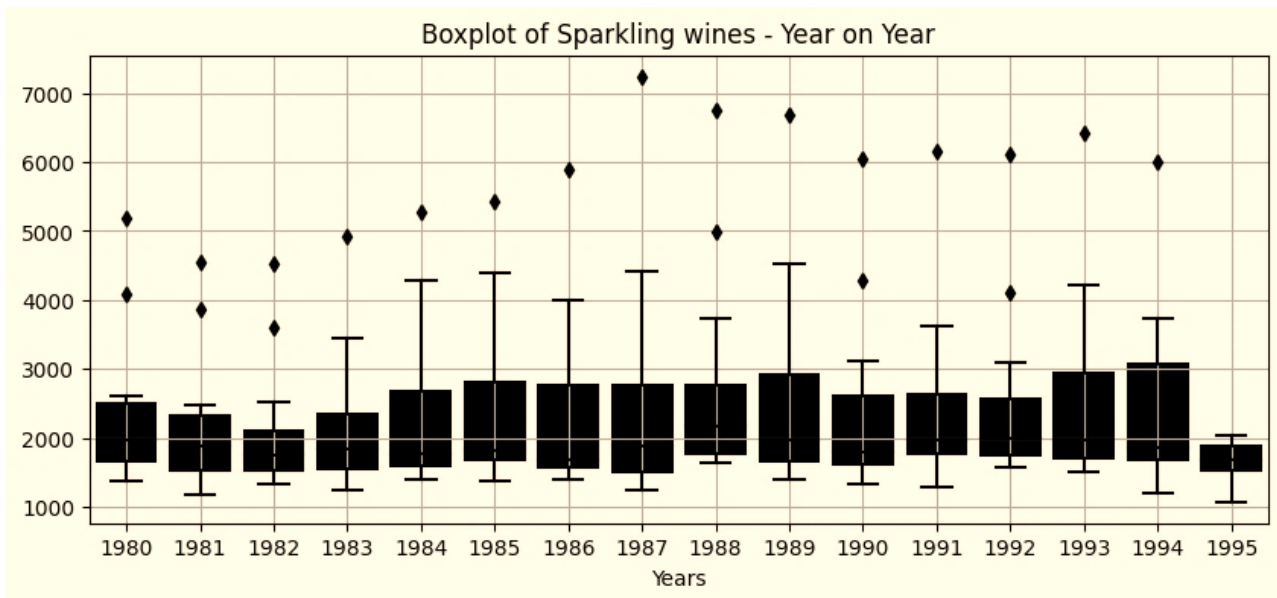


The next plot is also a boxplot, but one that is explained for each year.

We can see that each year has outliers, and some of them even have two. Again, these could be the peaks that we discussed previously.

1995 is an odd year, considering we only have data till July.

Out of 15 years, we have had 4 leaps years. – 1980, 1984, 1988 and 1992.



Next, we see a different plot that shows the variations between months of different years.



This plot shows a lot of information, but confirms that as we near holiday season the sales increase drastically,

On Average there is approximately, an increase of 268%, between January and December. We also see the most variation in December, around 2718 units. From January to June, the sales are constant, from July it picks up a growth trend, showing signs of trend within a year and seasonality.

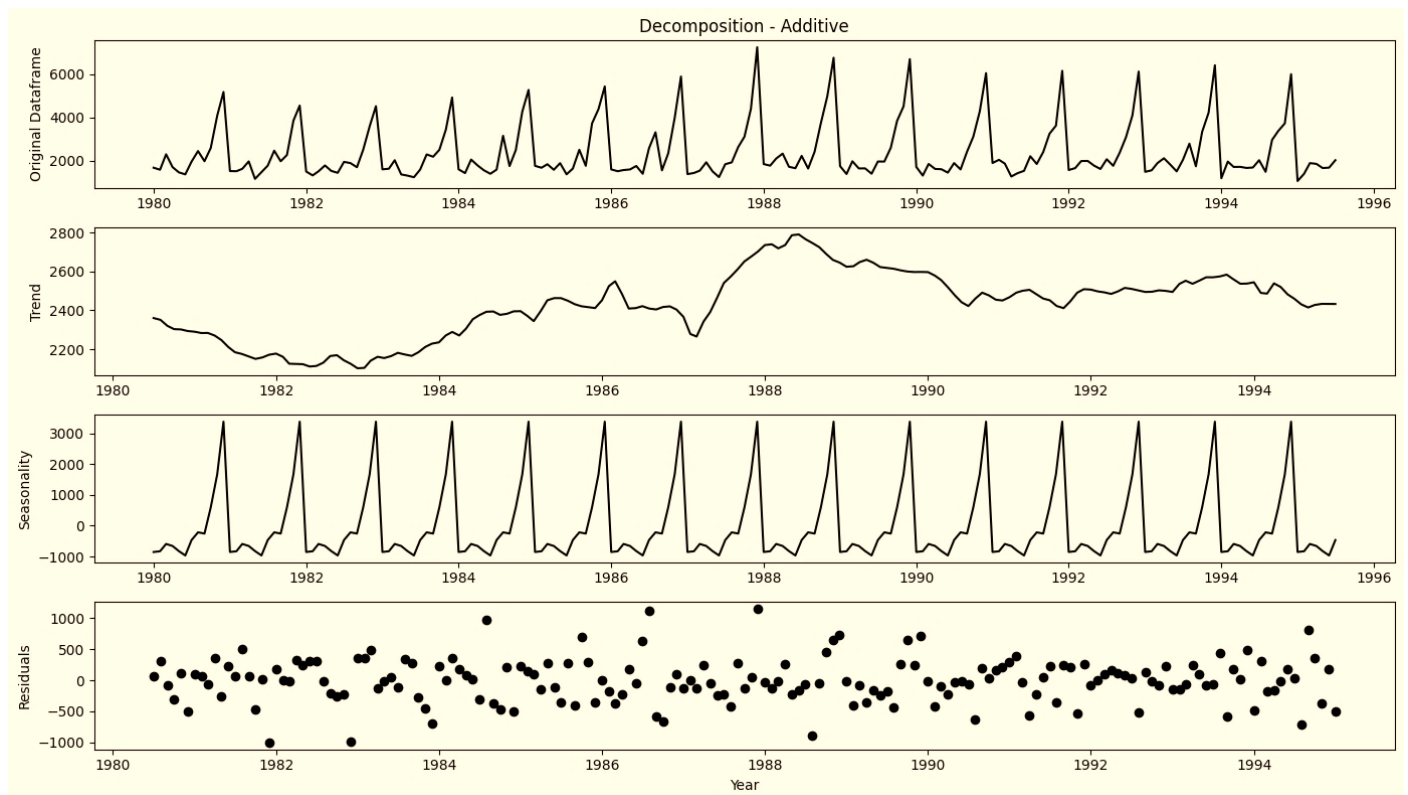
The months from August to December we see a lot of variances, even with multiple peaks in some months.



Now, let's decompose. Decomposition is the process where we dissect the data into multiple components. It helps us in understanding complex data patterns, improving predictive models, detecting anomalies, and preparing data for further analysis.

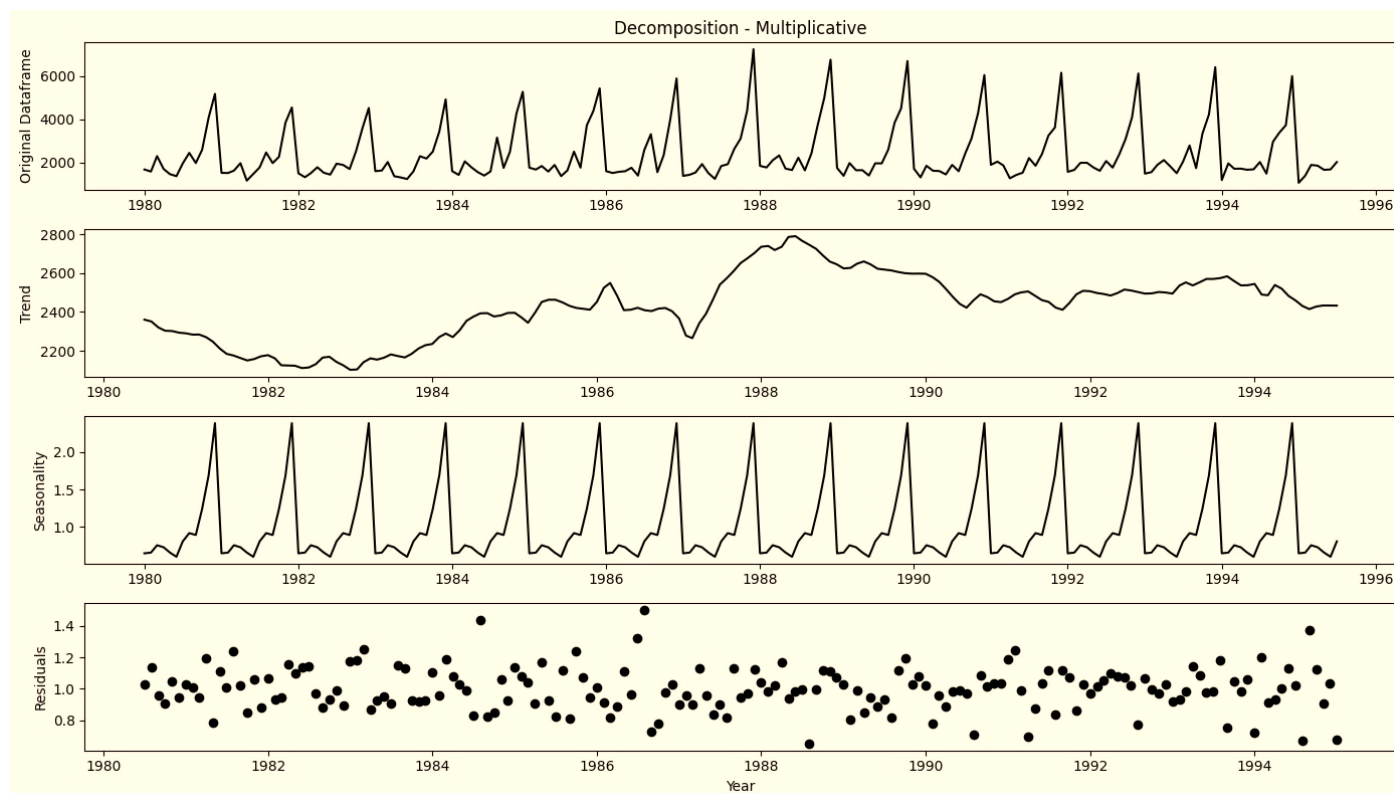
- **Trend:** Shows the long-term progression of the data, indicating an increase, decrease, or stability over time.
- **Seasonality:** Reveals patterns that repeat at regular intervals, like daily, weekly, monthly, or yearly patterns.
- **Residuals:** These are what remains after the trend and seasonal components have been removed, often providing insights into irregular factors or noise in the data.

Below is additive decomposition, where we assume data elements all add up to grow into a good and predictable fit.



From the plot trend is not ever increasing as there are dull moments, importantly trend does not show massive change over the years suggesting it to be additive. Seasonality is there and consistent which raises the question if info has been captured fully or not, if we compare it to the original data, we can see some data loss. The residuals are somewhat scattered based on the scales and do not form a pattern, this means that the fit is not perfect. This would ultimately lead to more error prone predictions.

Given below is multiplicative decomposition, where we assume data elements multiply and grow to become a better fit.



Trend seems to be similar like from additive decomposition. Seasonality seems to be better captured as multiplicative since it is keeping in line with trend.

Residuals are closely ranged between 0.8 to 1.4. the variation is better captured by this method. If we compare this to additive residuals which range between -1000 to +1000, multiplicative is far better.

## Train and Test data

Since we are dealing with a dataframe that is indexed based on Date-Time, we cannot randomize the train and test data split, we must segregate them in an orderly fashion. This is done so that we do not break continuity, furthermore time-series predictions work by predicting a portion of future timelines continuously and not by cherry picking.

Train dataframe to start from January of 1980 and end in December of 1994, totalling 132 months.

Test dataframe to start from January of 1991 totalling to 55 months.

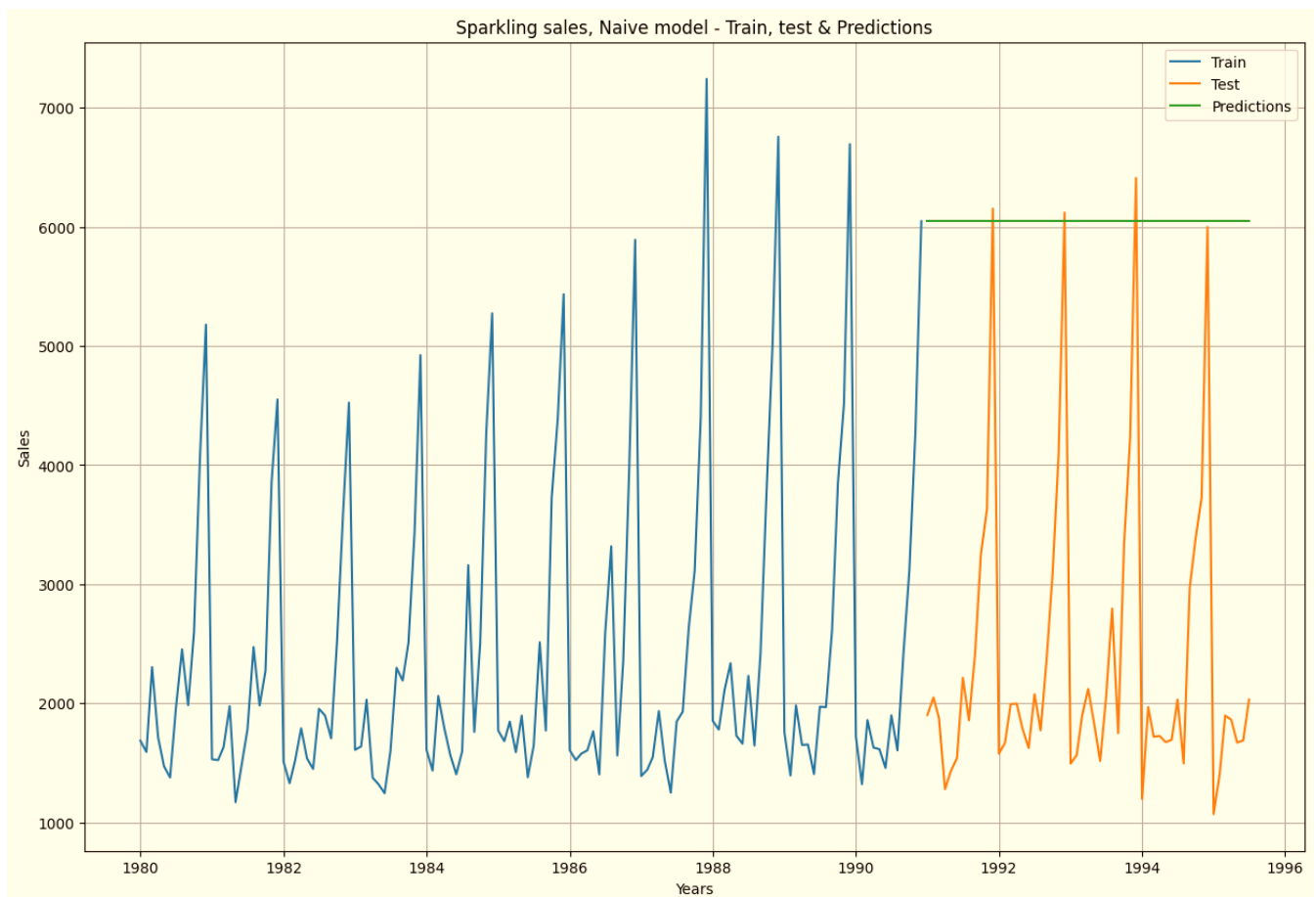
# Building Predictive Time-Series Models

## Forecast using Naïve method.

Naive forecasting is the simplest form of time series forecasting, based on the assumption that future values will be the same or very similar to the most recent observed value. The primary advantage of naive forecasting is its simplicity. It doesn't require any complex calculations or analysis; it is simply plug and play. We use the most recent month's sales as the prediction for the next month, making it easy to understand and apply.

In our case, we used the sales figure from the last available month (This would be December of 1990) as the forecast for each subsequent month. This method serves as a baseline to assess how much more accurate our sophisticated models are."

While naive forecasting is straightforward, it doesn't account for trends, seasonality, or any other changes in the data over time. Therefore, it's less accurate, especially in volatile markets or with seasonal sales data.



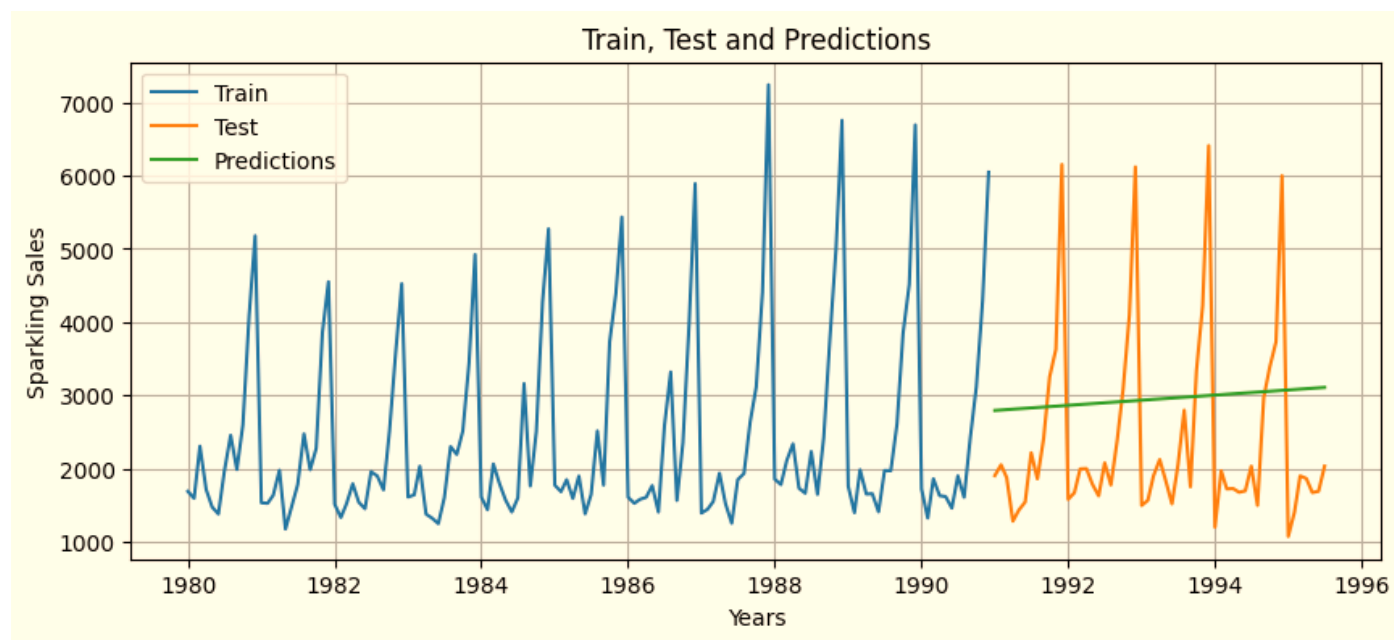
The green line in the plot above is the predictions, usually it is clear that it is not a good fit.

To measure how accurate our predictions are, we use a statistic called the Root Mean Square Error (RMSE). Think of RMSE as a ruler that measures the distance between our predicted sales and what happened. A smaller value means our predictions were closer to reality.

RMSE for naïve forecasting is "3864.279", considering the scales we are dealing with here, this is not particularly good. The benchmark has now been set.

## Forecast using Regression.

Linear regression helps us draw a line through based on historical sales data to project future sales. It's like connecting the dots in a way that shows where our sales are heading. Generally, regression employs dependent and independent variables(in various combinations). Here the dataframe being a univariate time series, the time index is the major factor.



The above plot shows the train, test, and linear regression predictions. The Green line cutting across the original test values in orange is the predictions, which shows a very poor fit. Regression has captured the trend mildly and the predictions mirror the same.

The RMSE is 1389.135, which is a high value considering the scale of the sales we are dealing with here. Linear regression clearly struggles to capture the seasonality we have in place.

Linear regression, from the looks of it, proves to be a poor and unreliable model, but still performs very well in comparison to naïve forecasting.

## Forecast using Moving Averages.

The moving averages method calculates the average sales over a set number of past months and uses this average to predict future sales. We analysed our sales using different 'trailing values' – 2, 4, 6, 9, 12, and 15 months. This means we calculated averages over these various time spans. A 2-month trailing value averages the last two months of sales, a 4-month value averages the last four months, and so on.

Using different trailing values allows us to understand how our sales forecasts change when we consider shorter versus longer historical periods.

The table below, shows a small peek of the data for 2 and 4 trailed averages.

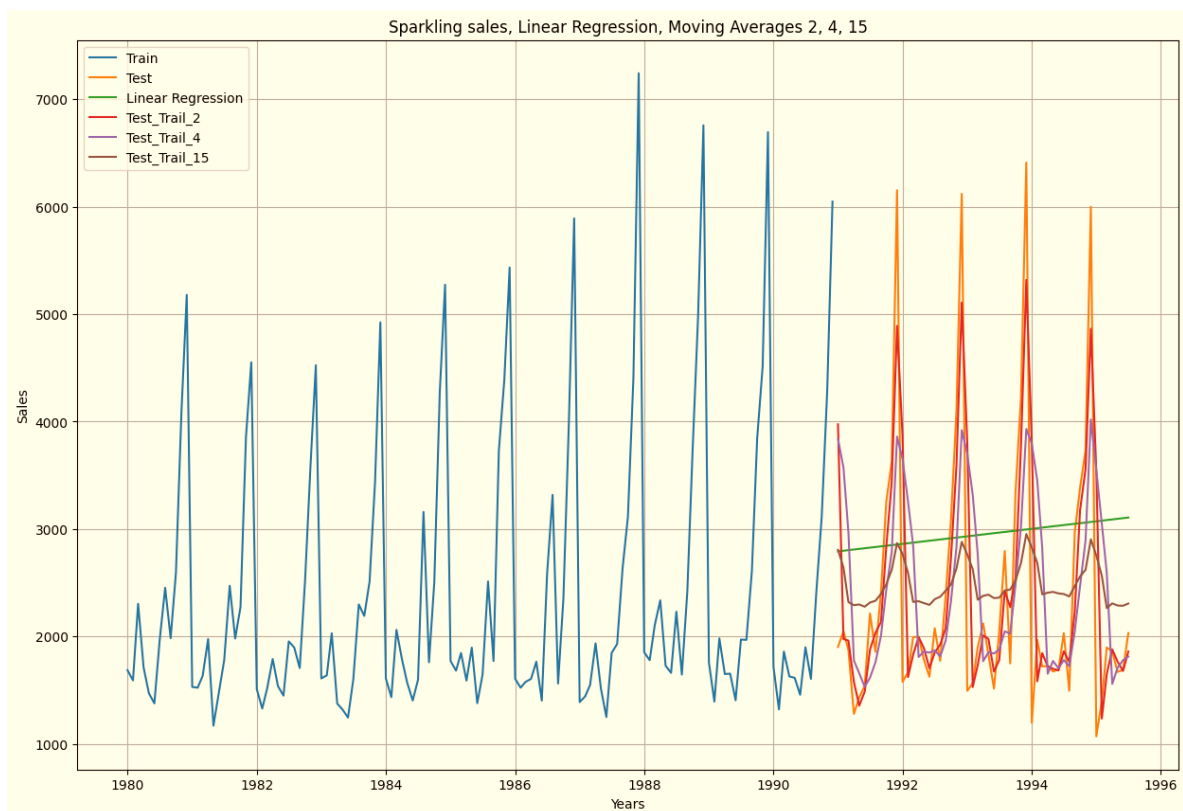
YearMonth	Sparkling	Sparkling - Trailing by 2	Sparkling - trailing by 4
1/1/1980	1686	NaN	NaN
1/2/1980	1591	1638.5	NaN
1/3/1980	2304	1947.5	NaN
1/4/1980	1712	2008	1823.25
1/5/1980	1471	1591.5	1769.5

Let's discuss the RMSE values,

Root mean square error for trail 2 is 813.401, Root mean square error for trail 4 is 1156.59,

Root mean square error for trail 6 is 1283.927, Root mean square error for trail 9 is 1346.278

Root mean square error for trail 12 is 1267.925, Root mean square error for trail 15 is 1173.813



The plot shows the train, test, and predictions for trailing 2, 4, 15 along with linear regression. The moving average method seems to perform better than linear regression, particularly the model where the trailing mean is done by 2 values.

The RMSE for trailing average of 2 is the best of the lot, even when comparing Linear regression. Second comes average trailed by 4 values, surprisingly average trailed by 15 values comes at third place, only then do the rest que up.

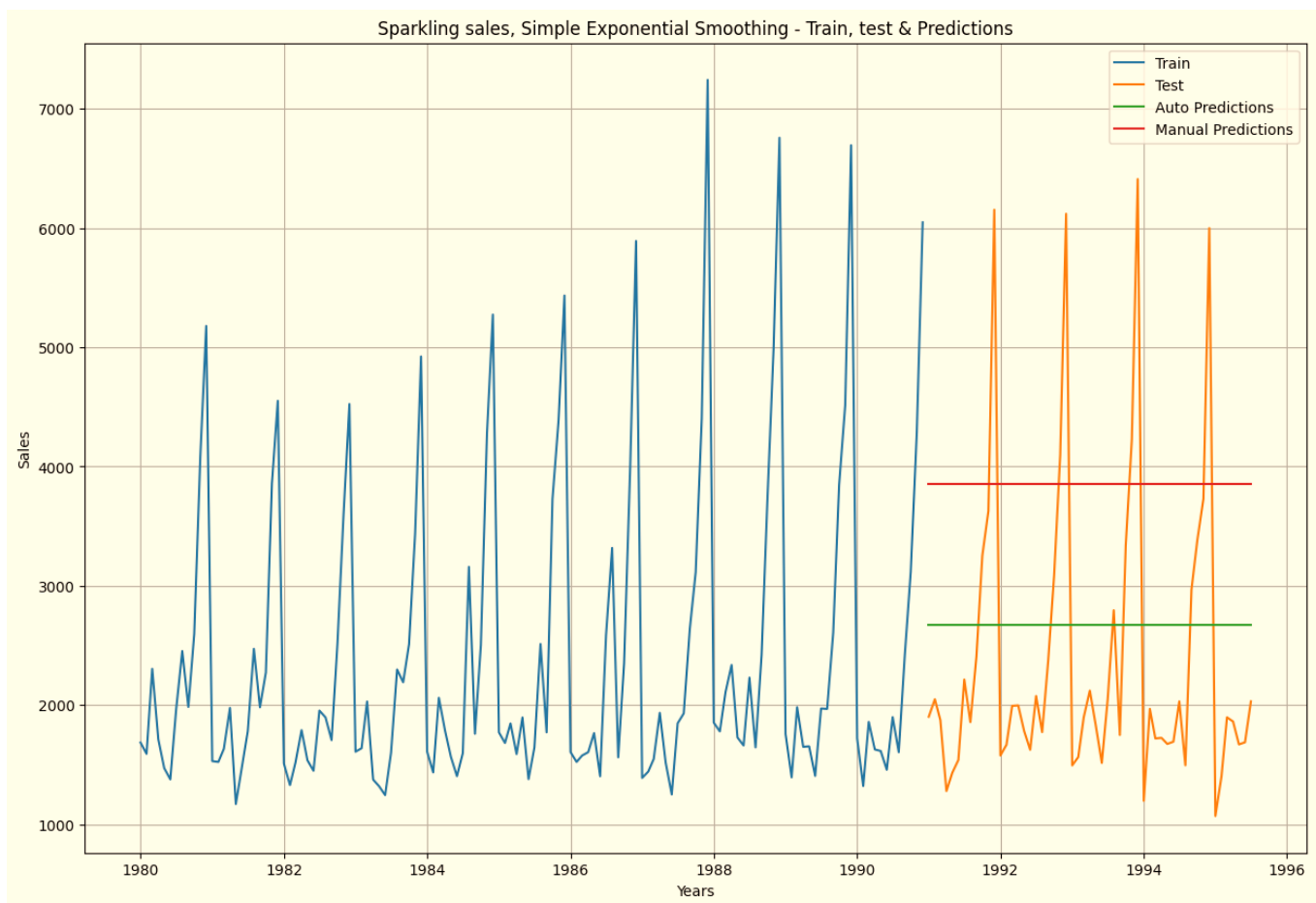
# Forecast using Simple Exponential Smoothing.

Simple Exponential Smoothing works by assigning exponentially decreasing weights to past sales data. This means more recent sales data is considered more heavily in predicting future sales.

In SES model, the key parameter we use is known as 'alpha' or the smoothing factor or level parameter. Alpha is a number between 0(lower) and 1(higher) that determines how much weight the model gives to the most recent sales data.

We autofit and let the package decide on the right alpha value initially, in this instance we get it to be around "0.03". The RMSE value we got was 1304.927. This is close to linear regression; hence this is not adequate and is a poor fit.

Finding the right alpha value is crucial in building a better model and understanding of the data. We tried multiple alpha values ranging from 0.3 to 0.9(in steps of 0.1).



The green and red lines going across are predictions done with alpha as "0.03(autofit)" and "0.3(manual)", provided in the plot above.

The alpha values we manually tested did not perform any better, they were way off their mark with the least being 0.3, with an RMSE value of 1935.507, very poor in comparison.

Simple exponential smoothing(SES) has not captured the trend or seasonality of the data provided. The performance of this model is less than moving averages model.

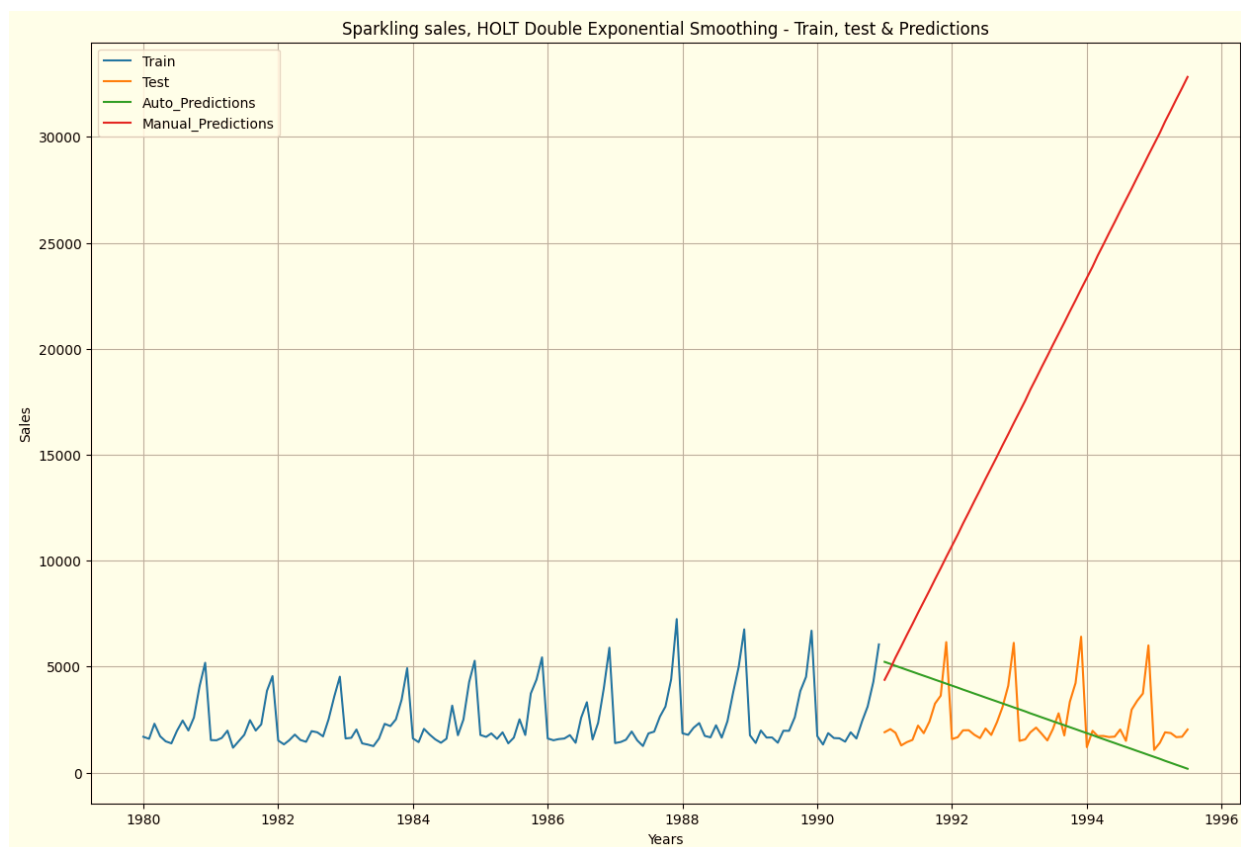
## Forecast using Double Exponential Smoothing(Holt's Method).

Double Exponential Smoothing is an extension of Simple Exponential Smoothing, designed to better handle data with trends, without considering seasonality. This method is ideal for data where trends, either increasing or decreasing, are present over time.

Like SES, alpha in Double Exponential Smoothing controls the level smoothing, we have another parameter called beta which controls the trend component. Higher beta value adapts quicker to recent change in trend and vice versa.

We autofit and let the package decide on the right alpha value initially along with the beta value, in this instance we get it alpha to be around "0.68" and beta to be '9.99e-05", this means beta is less than "0.00". **The RMSE value we got was 2007.239.** A very poor fit compared to SES.

Finding the right alpha and beta value is crucial in building a better model and understanding of the data. We tried multiple alpha values ranging from 0.3 to 0.9(in steps of 0.1) and likewise for beta too.



The red line in the plot above corresponds to a Manual prediction done with alpha as "0.3" and beta as "0.3"(the best of the manual combinations we tested). Similarly, the green line corresponds to Auto prediction done with as alpha as "0.68" and beta as "0.000".

The manual predictions performed abysmal in comparison. An **RMSE value of "18259.111"** was obtained.

Double exponential smoothing(DES) has not captured the trend coherently for the data provided. The performance of this model pales in comparison to moving averages.

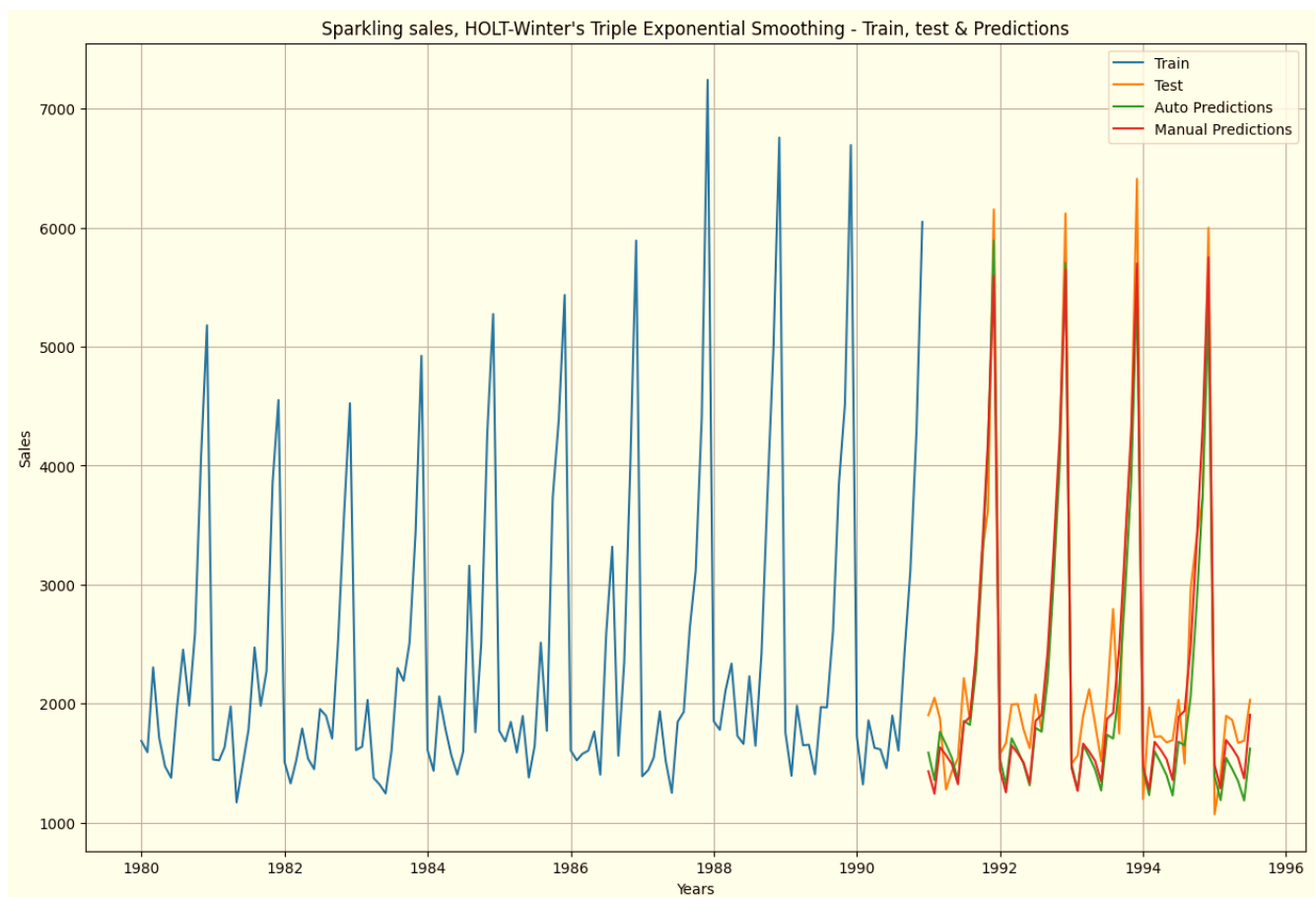
## Forecast using Triple Exponential Smoothing(Holt-Winter's Method).

Triple Exponential Smoothing takes it up a notch from DES by adding a seasonal component. It's particularly effective for data where both trend and seasonality are present, like the dataframe that we have with us. Here gamma is used to smooth the seasonality component.

Initially we let the package autofit the parameters, obtained are alpha as "0.11", beta as "0.049" and gamma as "0.36", The RMSE value we got was 403.706. As expected, TES has performed better than all the models until now.

Manual models were created in the same lines as we did for DES, gamma was also set within the same value ranges. The best model from this was where alpha, beta and gamma all as "0.3", with an RMSE value of "361.397". Manual predictions performed better than auto fit by a good margin.

The plot below shows the predictions, TES model has performed good to an extent that it is hard to distinguish test and predictions.





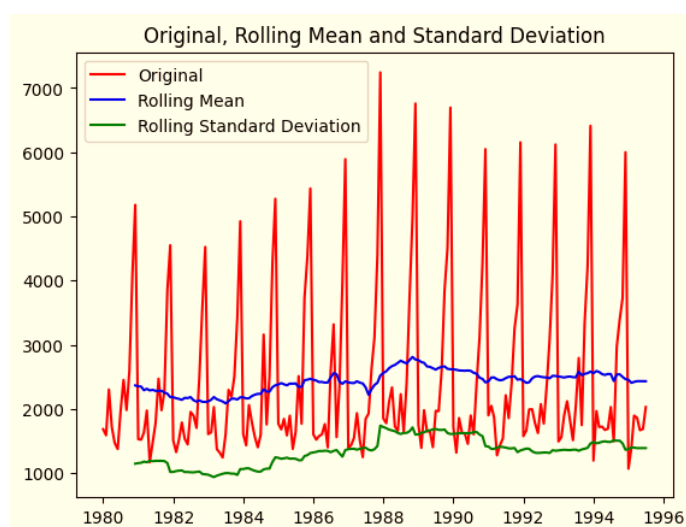
# Test of Stationarity

In time series analysis, it's important to know whether our data is “STATIONARY”, meaning that its statistical properties like mean, variance, and the like are constant over time. This is important for certain types of forecasting models to work effectively.

We can think of stationarity like levelling the playing field. We might have short term fluctuations here and there, but overall values remain similar.

One simple way to check for stationarity is by visually inspecting the plot of the time series. If the series appears to have a consistent mean and variance, and doesn't portray trend or seasonality, it may be stationary. But visual inspection might not be enough in most real-world scenarios, hence we use a statistical test called Augmented Dickey-Fuller (ADF) test. This test checks if a unit root is present in the series, which is a sign of non-stationarity. In simple terms, it helps us determine if the patterns in our data are influenced by time-based structure.

Below plot on the left shows the mean against the original values along with standard deviation.



Now let's run Augmented Dickey-Fuller Test on our timeseries(in its unadulterated form) and check for stationarity, below environment is set and conditions are determined.

Null Hypothesis  $H_0$  : Time Series is non-stationary.

Alternate Hypothesis  $H_a$  : Time Series is stationary.

If p-value(significance) < 0.05 then null hypothesis (i.e.) Time Series is non-stationary is rejected else the Time Series is non-stationary is failed to be rejected.

The obtained p-value “0.60” is greater than 0.05, hence we have failed to reject the null hypothesis.

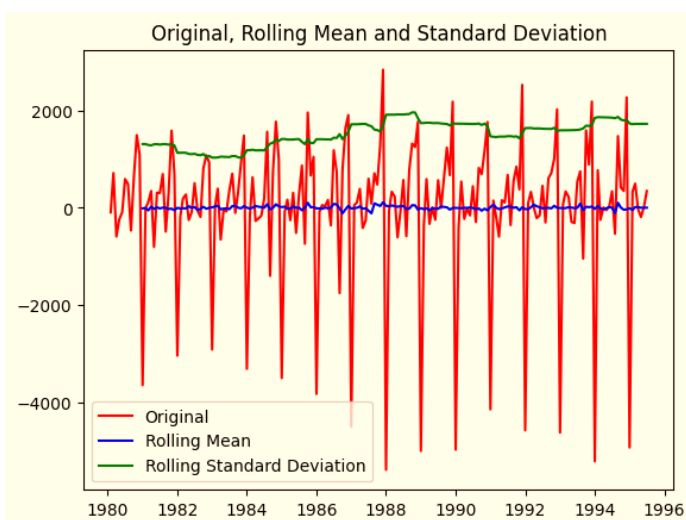
If our data is found to be non-stationary and, it is in our case, we need to transform it to make it stationary. Hence, we will involve differencing. Differencing is not the only method that we can employ there are other ways like employing logarithms transformations.

The plot on the right shows the mean and original values after differencing to the order of 1, now by referencing the scales we can see that the dataframe has achieved stationarity.

Applying Augment Dicky-Fuller test and basing the same conditions, we get the significance to be “0.000”, meaning we have successfully rejected the null hypothesis.

The dataframe has achieved stationarity after differencing it by an order of 1.

The above steps that we have done are very crucial to the models that are going to be built. Stationarity helps in setting the ground and providing reliable and robust forecasting methods and ensuring our business strategies are based.



# Auto Regressive Integrated Moving Average(ARIMA)

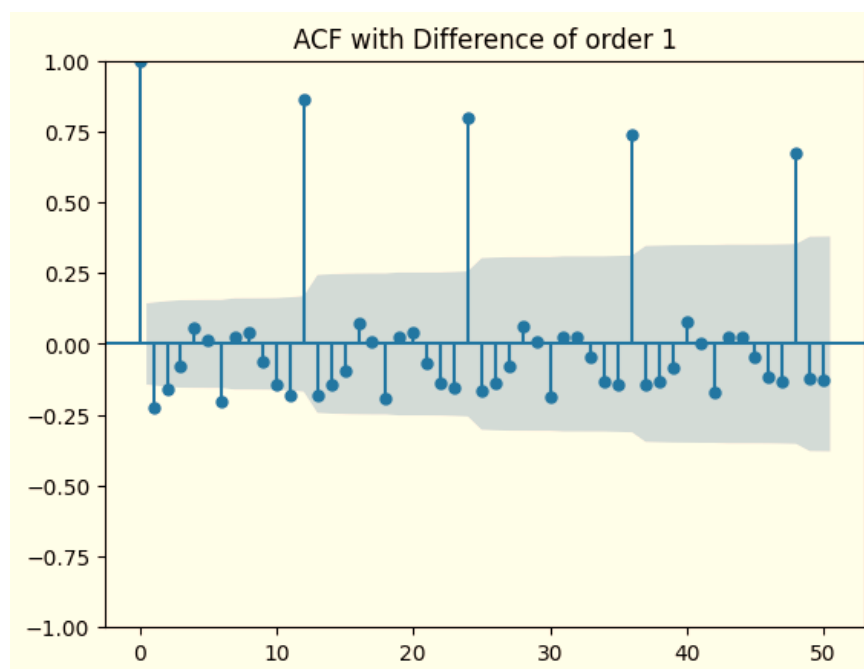
ARIMA is a sophisticated forecasting model that combines three key concepts: Auto Regression (AR), Integration (I), and Moving Averages (MA). AR looks at the relationship of current to be predicted values with its past counterparts, Integration(I) helps in making the series stationary and Moving averages(MA) incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.

The three concepts of ARIMA model have their unique parameters: (p, d, q). 'p' is the order of the Auto Regressive part, 'd' is the degree of differencing required to make the series stationary, and 'q' is the order of the Moving Average part.

Selecting the right (p, d, q) values is crucial. This is where the Akaike Information Criterion (AIC) comes in. AIC is a tool used to compare different ARIMA models on how well they fit the data. A lower AIC value generally indicates a better model. Our goal is to find the ARIMA model with the lowest AIC value, as it suggests an optimal balance between model fit and complexity.

Before we proceed with model building, we must first uncover what the values of “p” and “q” would be, “d” is already set a ‘1’, courtesy of differentiation. This is where two major plots come in, they are Autocorrelation Function(ACF) and Partial Autocorrelation Function(PACF).

**ACF for “q” (MA parameter):** The ACF plot helps us identify the optimal value for 'q', the MA (Moving Average) part of our ARIMA model.

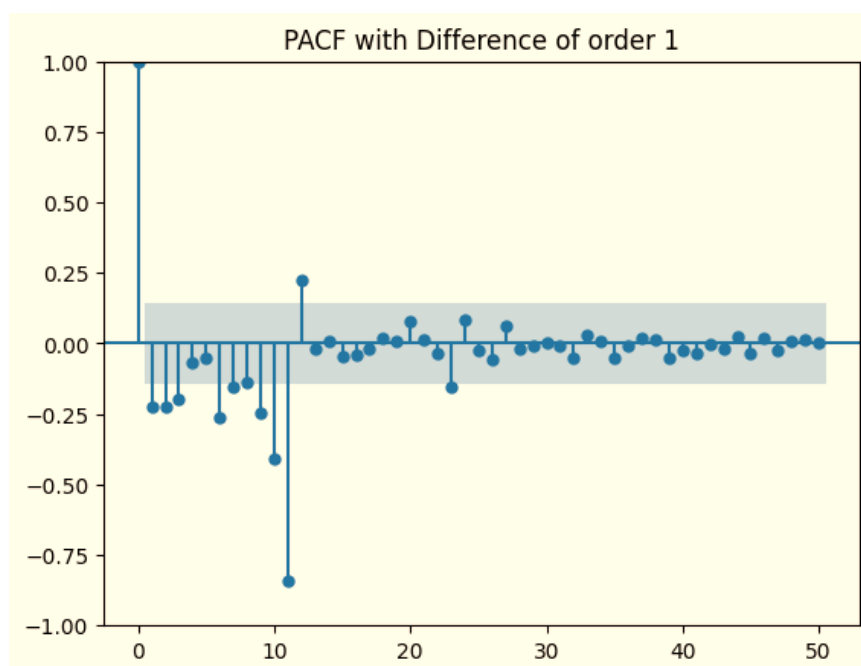


The plot above is used to find the optimal value(s). The shaded region represents the confidence bands, hence if a point(lag value) lies outside the band, then it indicates that it is statistically significant. It also means we can find meaningful relationships between observations at this lag.

Look for the lag where the correlation starts to become insignificant (bars start to fall within the shaded area). This point often suggests a good value for 'q', in this case being '3'.

Note that the first point is a value gauged against itself therefore it will always be 1.

**PACF for “p” (AR parameter):** Similarly, the PACF plot is used to determine 'p', the AR (Auto Regressive) part.



The same guidelines from ACF apply here to, hence 4 would be a good value for ‘p’ as start.

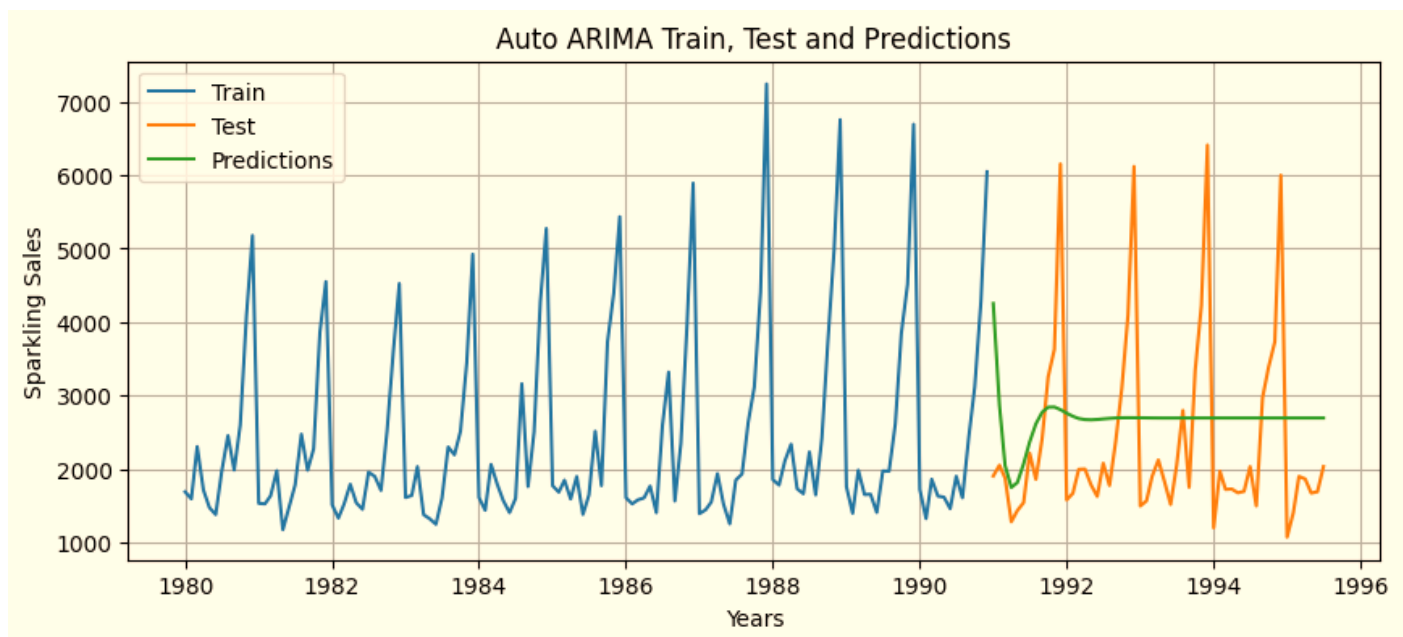
Now let’s model ARIMA, instead of choosing one value for p and q we will use this range 0, 1, 2, 3, 4 to test different parameter combinations (insignificant values are also chosen to determine how they perform). ‘d’ will remain constant at 1.

Please find the top 5 parameter combinations that worked best.

Parameter (p, d, q)	Akaike_Value
(4, 1, 4)	2213.04
(2, 1, 2)	2213.51
(3, 1, 4)	2220.01
(3, 1, 3)	2221.46
(4, 1, 3)	2222.90

Surprisingly, parameters “pdq” having insignificant values has performed marginally better than significant ones. But if we just dissect the values a bit further, we can see that “pdq” combination of 2,1,2 has performed almost equally (just differing by 4-tenths) to 4,1,4. This blatantly shows how insignificant their contributions are to our predictions.

**We built ARIMA with ‘pdq’ values as (2, 1, 2). The RMSE value that we obtained was “1299.979”.** ARIMA has produced a model that is average in comparison to the models we have built until now.



The plot above shows the train, test, and predictions. We can see that the performance of ARIMA is not up to par.

## Seasonal Auto Regressive Integrated Moving Average(SARIMA)

Seasonal Auto Regressive Integrated Moving Average is an extension of ARIMA model by adding a seasonal component. It's particularly useful for time series data with clear seasonal patterns like the one discussed now.

A SARIMA model is typically denoted as SARIMA(p, d, q)(P, D, Q, s). Here, p, d, q are the non-seasonal ARIMA parameters, and P, D, Q are the seasonal components of the model, with 's' representing the length of the seasonal cycle.

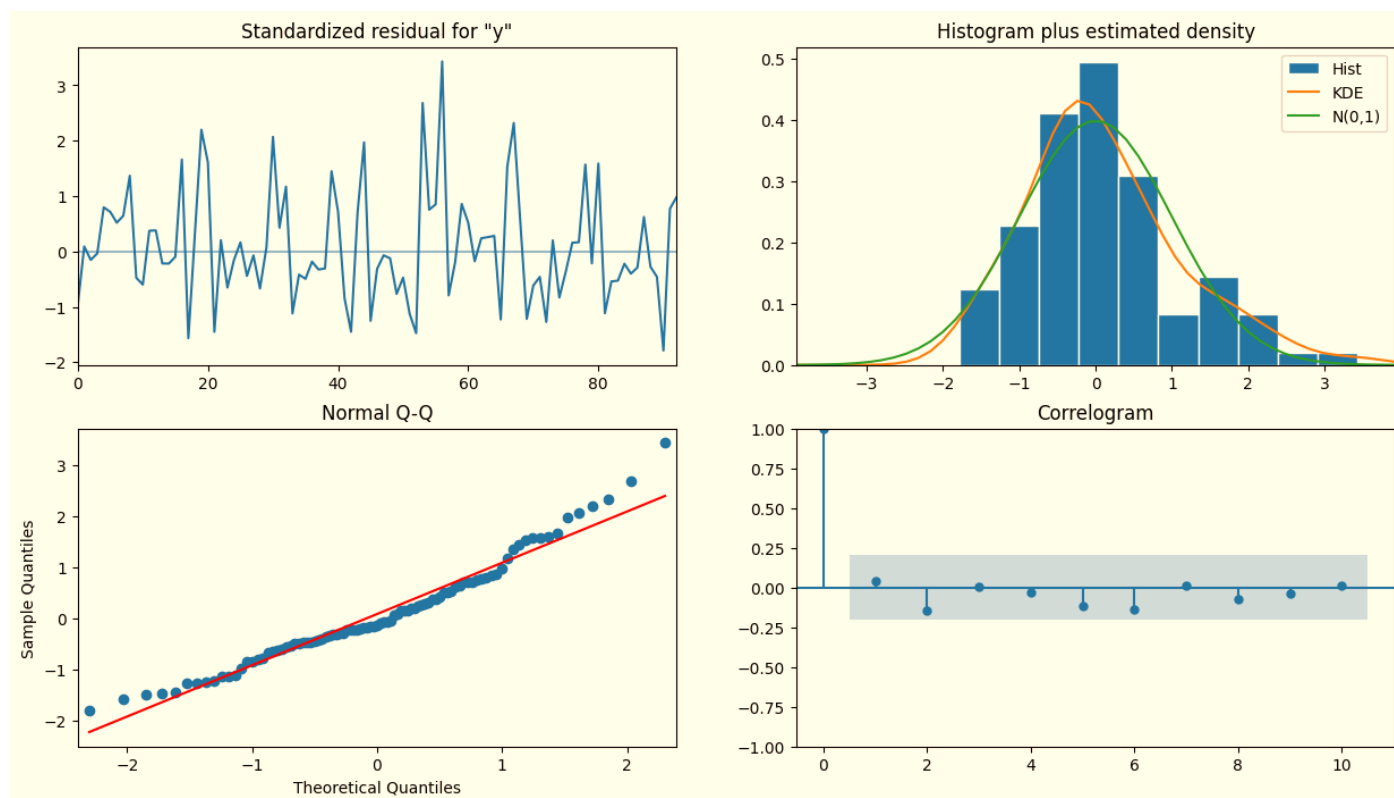
For example, SARIMA(1, 1, 1)(1, 1, 1, 12) would mean a model with one non-seasonal AR term, one differencing, one MA term, and similarly one seasonal AR term, one seasonal differencing, and one seasonal MA term, with a seasonal period of 12 (which could be 12 months in a yearly cycle).

Now let's model SARIMA, we will use the same 'pdq' value range from ARIMA for both non-seasonal and seasonal components, seasonality 's' alone will have the value as '12' which is consistent with our data.

Please find the top 5 parameter combinations that worked best.

Non-Seasonal 'pdq'	Seasonal 'pdq' with 's'	Akaike_Value
(1, 1, 1)	(0, 0, 3, 12)	12
(1, 1, 3)	(0, 0, 3, 12)	16
(2, 1, 2)	(0, 0, 3, 12)	16
(1, 1, 4)	(0, 0, 3, 12)	18
(3, 1, 1)	(2, 1, 3, 12)	20

The RMSE obtained from the top pick parameter combinations above is “1072.745”, which is better than ARIMA and linear regression but still not good as TES.



**Top-Left: Standardized Residual plot** shows the residuals of the model over time. You would want to see no direct pattern or trends, which would suggest that the model has captured the time series structure well. The residuals seem to be fluctuating around zero without systematic patterns, which is a good sign.

**Top-Right: Histogram plus Estimated Density plot** compares the distribution of the standardized residuals to a normal distribution. The blue bars represent the histogram of the standardized residuals, showing the frequency of residual values. The orange line is a kernel density estimate (KDE), which is a smoothed version of the histogram. The green line represents a standard normal distribution. You would want the KDE (orange line) to follow the normal distribution (green line) closely. In our plot there seems to be a slight deviation, implying it is not perfect enough. This indicates that the model could be improved.

**Bottom-Left: Normal Q-Q plot** is used to assess if the residuals are normally distributed. The dots represent the quantiles of the residuals, and the line represents a normal distribution and following that would mean we have a normal distribution at hand. As seen, there are some deviations, especially at the ends, suggesting some deviations.

**Bottom-Right: Correlogram (ACF Plot)** shows the autocorrelation of the residuals at different lags. Ideally, you would want to see that the autocorrelations are within the blue shaded area. This would suggest that the model has captured the time series' autocorrelation structure well. In our plot, all the autocorrelation coefficients are within the blue area, which suggests that there's no significant autocorrelation in the residuals.

## Summary of built models and performance

Below is a table that shows the performance of all the models that built and predicted. For our dataframe triple exponential smoothing manual approach followed by Auto-fit have taken first and second spots respectively. Furthermore, these were the only models to have RMSE less than 500. Third is Moving average by using two points with 814 as RMSE.

We have considered Naïve method to be our benchmark, considering the performance characteristics, **TES manual approach is relatively 90.7% better than naïve method.**

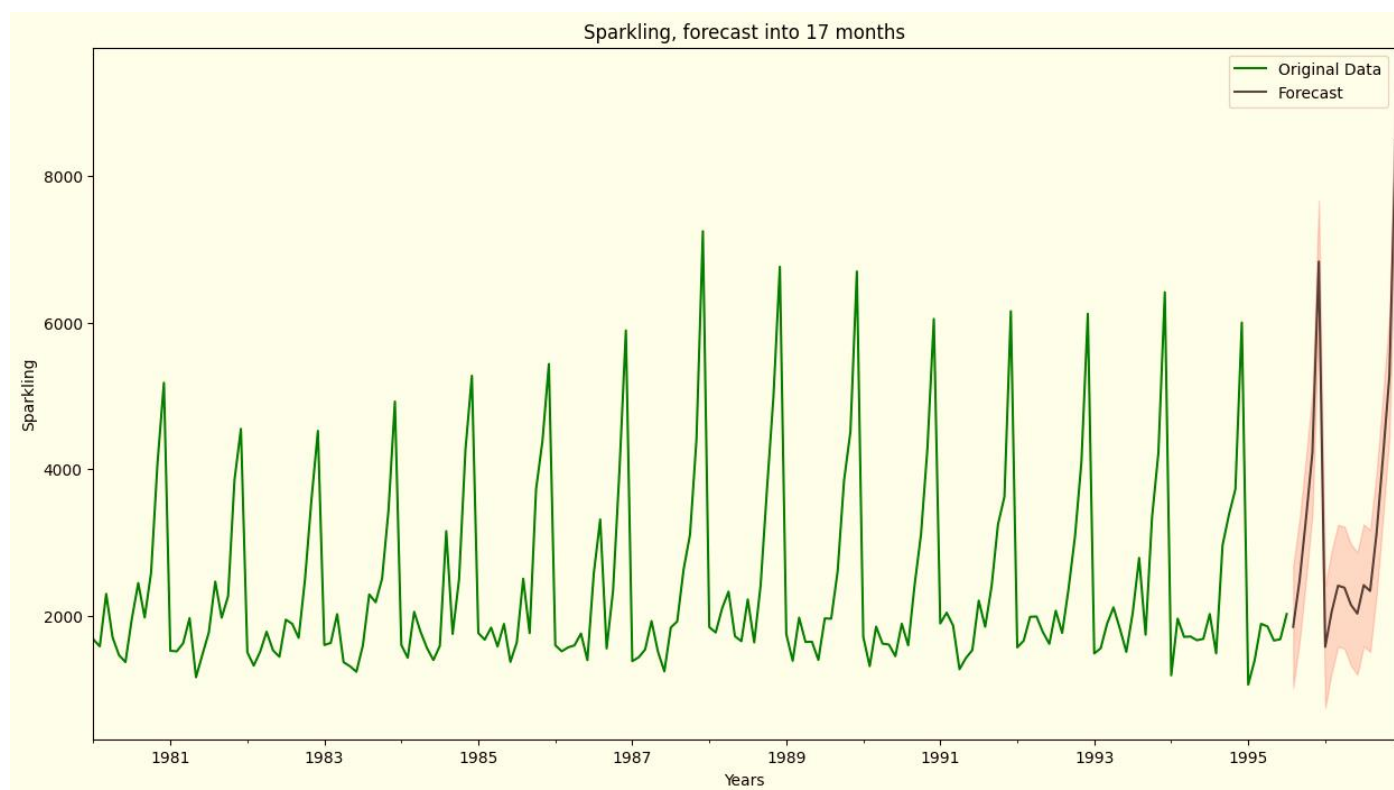
Model	Root Mean Square Error
Triple Exponential Smoothing(Manual) where alpha, beta and gamma all are 0.3.	361.397
Triple Exponential Smoothing(Auto-fit) where alpha, beta and gamma are 0.11, 0.04, 0.36	403.706
Moving Average Trailing by 2 points	813.401
SARIMA with “pdq” as (1, 1, 1) and “PDQs” as (0, 0, 3, 12)	1072.745
Moving Average Trailing by 4 points	1156.590
Moving Average Trailing by 15 points	1173.813
Moving Average Trailing by 12 points	1267.925
Moving Average Trailing by 6 points	1283.927
ARIMA with “pdq” as (2, 1, 2)	1299.980
Simple Exponential Smoothing(Auto-fit) with Alpha as 0.03	1304.927
Moving Average Trailing by 9 points	1346.278
Linear Regression	1389.135
Simple Exponential Smoothing(Manual) with Alpha as 0.3	1935.507
Double Exponential Smoothing(Auto-fit) where Alpha is 0.68 and beta as 0.000	2007.239
Naïve Method	3864.279
Double Exponential Smoothing(Manual) with alpha and beta as 0.3	18259.111

The worst model was Double exponential smoothing manual approach that has a whopping 372.5% performance degrade to our benchmark.

## Forecast - 17 month's

The best model we have is Triple exponential smoothing where alpha, beta and gamma are “0.3”. Now we will use this model to forecast 17 months from 1995 August to 1996 December.

The below plot shows the forecasts(the black line), since these are predictions there can be some errors, hence the shaded region around the forecast are the confidence bands set at 95% confidence.



As the value start to peak, we can see that the confidence bands are becoming tighter, but on the other hand when it is the off season, we have much variability.

The forecast also suggests that we might see a new all time high in sales, in December of 1996.

## Process Summary and business insight's

We have been tasked with forecasting sales of wine and the dataframe was provided for use.

Plotting the dataframe showed us hints of mild trend, enormous seasonality. Decomposing the dataframe confirmed the same. We segregated the data into train and test.

After completing pre-processing of data, it was used to build time series forecasting models. The benchmark model was set by using the naïve method, we used Linear regression, moving averages trailing by various values, Exponential smoothing techniques(simple, double, and triple).

The data was also used to build complex models like Autoregressive integrated moving averages(ARIMA) and Seasonal Autoregressive integrated moving averages(SARIMA).

To make ARIMA and SARIMA a reality we had to perform “test of stationarity” to see if the dataframe has constant mean and variations throughout. Augmented Dicky–fuller test was used to achieve stationarity.

ARIMA and SARIMA were built upon multiple combinations of ‘pdq’, but we kept seasonality as ‘12’ since we had strong evidence to support the claim. Akaike Information criterion(AIC) was used to gauge the parameter combinations and get the best of the lot.

Comparing the performance of the model throughout was done with the help of Root mean square error(RMSE). The lower RMSE value the better the model performed.

Triple exponential smoothing with a huge performance difference over other models took the top spot on the podium. Thereby becoming the best performing model.

### Business insights.

- First and foremost, we can expect an increase in sales of “Sparkling” wine, hence we must prepare to meet demand.
- The sales will see an increase post July and we will see a huge surge in the holiday season(the months of November and December).
- The initial months(from January to June) we can see improvement, but the road is bumpy with minimal boost.
- Forecast suggests that we can see a new all-time sales high in December of 1996.
- There seems to be no downtime in the coming months, hence we must make use of every opportunity.