STATS5000 Project Report

Darsh Shah

12/5/2021

1. Problem Statement

Global Mart is online store super giant having worldwide operations. It takes orders and delivers across the globe and deals with all the major product categories - consumer, corporate & home office. The purpose of this project is to understand the what categories of products are the most profitable and during which months they are the most profitable. I also want to understand what market segment is most profitable and what kind of products are the most profitable in those market segments. It would also be interesting to understand what days are the most profitable for the store across the world and also in particular market segments. All these analyses should help the store get a broader idea of their profitability across the world and what products are bringing the most revenue.

Dataset URL: https://raw.githubusercontent.com/vaibhavpalkar21/Time-Series-Retail-Giant-Sales-Forecasting/master/Global%20Superstore.csv

Github URL: https://github.com/darsh-shah4548/Inventory-Demand-Forecasting

2. Importing the dataset

The data set has been imported from a github repository. I was browsing for project ideas and I found the Readme of this github repo quite interesting and wanted to work on this data set. Since they provided this data set in their github repo, i decided to simply use this data set.

```r
#Importing the dataset.
global <- read.csv("https://raw.githubusercontent.com/vaibhavpalkar21/Time-Series-Retail-Giant-Sales-Fo

glimpse(global)
```

```
## Rows: 51,290
## Columns: 24
## $ Row.ID         <int> 32298, 26341, 25330, 13524, 47221, 22732, 30570, 31192,~
## $ Order.ID       <chr> "CA-2012-124891", "IN-2013-77878", "IN-2013-71249", "ES~
## $ Order.Date     <chr> "31-07-2012", "05-02-2013", "17-10-2013", "28-01-2013",~
## $ Ship.Date      <chr> "31-07-2012", "07-02-2013", "18-10-2013", "30-01-2013",~
## $ Ship.Mode      <chr> "Same Day", "Second Class", "First Class", "First Class~
## $ Customer.ID    <chr> "RH-19495", "JR-16210", "CR-12730", "KM-16375", "RH-949~
## $ Customer.Name  <chr> "Rick Hansen", "Justin Ritter", "Craig Reiter", "Kather~
## $ Segment        <chr> "Consumer", "Corporate", "Consumer", "Home Office", "Co~
## $ City           <chr> "New York City", "Wollongong", "Brisbane", "Berlin", "D~
## $ State          <chr> "New York", "New South Wales", "Queensland", "Berlin", ~
## $ Country        <chr> "United States", "Australia", "Australia", "Germany", "~
## $ Postal.Code    <int> 10024, NA, NA, NA, NA, NA, NA, NA, 95823, 28027, 22304,~
## $ Market         <chr> "US", "APAC", "APAC", "EU", "Africa", "APAC", "APAC", "~
## $ Region         <chr> "East", "Oceania", "Oceania", "Central", "Africa", "Oce~
## $ Product.ID     <chr> "TEC-AC-10003033", "FUR-CH-10003950", "TEC-PH-10004664"~
## $ Category       <chr> "Technology", "Furniture", "Technology", "Technology", ~
## $ Sub.Category   <chr> "Accessories", "Chairs", "Phones", "Phones", "Copiers",~
## $ Product.Name   <chr> "Plantronics CS510 - Over-the-Head monaural Wireless He~
## $ Sales          <dbl> 2309.650, 3709.395, 5175.171, 2892.510, 2832.960, 2862.~
## $ Quantity       <int> 7, 9, 9, 5, 8, 5, 4, 6, 5, 13, 5, 5, 4, 7, 12, 4, 9, 14~
## $ Discount       <dbl> 0.0, 0.1, 0.1, 0.1, 0.0, 0.1, 0.0, 0.0, 0.2, 0.4, 0.0, ~
## $ Profit         <dbl> 762.1845, -288.7650, 919.9710, -96.5400, 311.5200, 763.~
## $ Shipping.Cost  <dbl> 933.57, 923.63, 915.49, 910.16, 903.04, 897.35, 894.77,~
## $ Order.Priority <chr> "Critical", "Critical", "Medium", "Medium", "Critical",~
```

3. Data Preprocessing

Let's first check for missing values

```
global %>%
  dplyr::select(everything()) %>%
  summarise_all(funs(sum(is.na(.))))
```

```
##   Row.ID Order.ID Order.Date Ship.Date Ship.Mode Customer.ID Customer.Name
## 1      0        0          0         0         0           0             0
##   Segment City State Country Postal.Code Market Region Product.ID Category
## 1       0    0     0       0       41296      0      0          0        0
##   Sub.Category Product.Name Sales Quantity Discount Profit Shipping.Cost
## 1            0            0     0        0        0      0             0
##   Order.Priority
## 1              0
```

Only the Postal code is the column that has missing values. Postal codes are not important for my analyses, so i can remove that column from the dataset.

```
global <- global[,-12]
```

```
#Removing the duplicate entries
global <- global[!duplicated(global[c(2,15)]),]
```

Let's convert the order date and ship date into default date format.

```
global$Order.Date <- as.Date(global$Order.Date,"%d-%m-%Y")
global$Ship.Date <- as.Date(global$Ship.Date,"%d-%m-%Y")
```

4. Exploratory Data Analysis

For the EDA, I mainly wanted to extract as much information as i can from the data set. Therefore, i decided to ask as to what questions would the CEO of the store like to know the answers of considering the data on hand. The questions that i tried answering are:

- What is the trend of orders received by the store globally across 2011 to 2014? - Fig 4.1
- what are the most profitable market segments across the world? - Fig 4.2
- What are the most profitable market segments population per million wise? - Fig 4.3
- During which months of the year, are these market segments most profitable? - Fig 4.4 & 4.5
- what are the most profitable sub categories of products across the world? - Fig 4.6
- How much does each sub-category contribute towards the total profit? - Fig 4.7
- What are the most profitable sub categories in the most profitable market segments? - Fig 4.8 and 4.10
- How are the most profitable sub categories doing over time in those market segments? - Fig 4.9 and 4.11

Let's look at how has the frequency of orders progressed for the store from 2011 to 2014.

```r
#Formatting the order_month_year to only have month and year.
global$order_month_year <- as.factor(format(global$Order.Date,"%m-%Y"))

ggplot(global,aes(order_month_year))+geom_histogram(stat = "count") +
  theme(axis.text.x = element_text(angle = 90, vjust=0.5, hjust=1)) +
  labs(x="Month and Year of Orders", y = "Count of Orders")
```

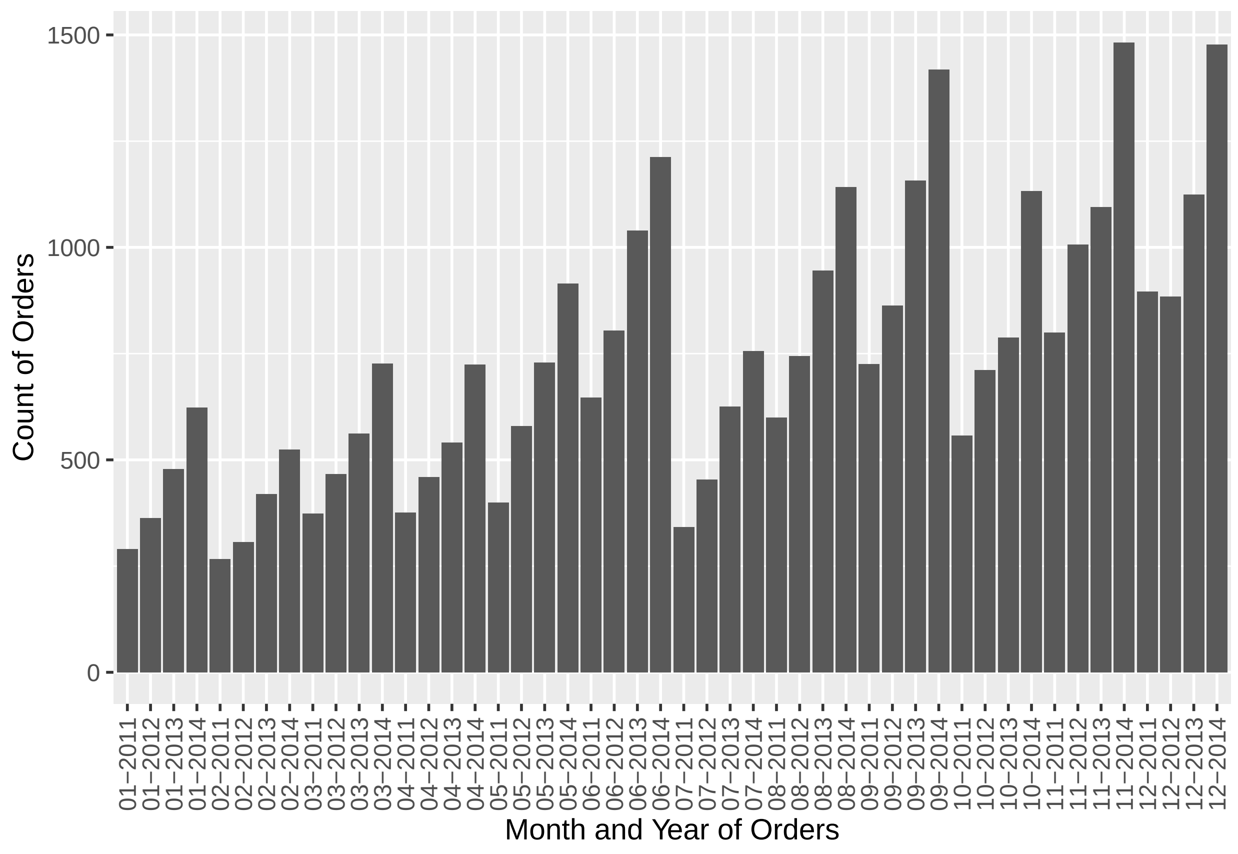


Fig 4.1: Trend of Orders received by the store worldwide from 2011 to 2014

From the graph, It can be seen that overall there has been an increase in orders for every month from 2011 to 2014. This also suggests that the store should have the most orders for the month of November and December of 2015. The increase in orders during those months could be because of the Holiday period (Thanksgiving, BlackFriday, Cyber-Monday and Christmas). We can also see a trend that there is a decrease in orders from 12-2011 to 01-2012. This could also be because people might spend a lot during the Christmas break.

Since, the dataset has a **Segment** and **Market** Column, I am keen on understanding which market segments are more profitable and what sort of products are the most profitable in that market segment. Therefore, I created a new column, **Market_Segment**

```
global$Market_Seg <- paste(global$Market, global$Segment, sep = "_")
global$Market_Seg <- as.factor(global$Market_Seg)
```

Now, Let's look at which Market_Segments are profitable

```
market_segments <- global %>%
  group_by(Market_Seg) %>%
  summarise(sales = sum(Sales),
            profit = sum(Profit),
            quantity = sum(Quantity))

ggplot(market_segments,aes(Market_Seg, profit))+
  geom_bar(position = "dodge", stat="identity")+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  labs(x="Market Segments", y="Profits")
```

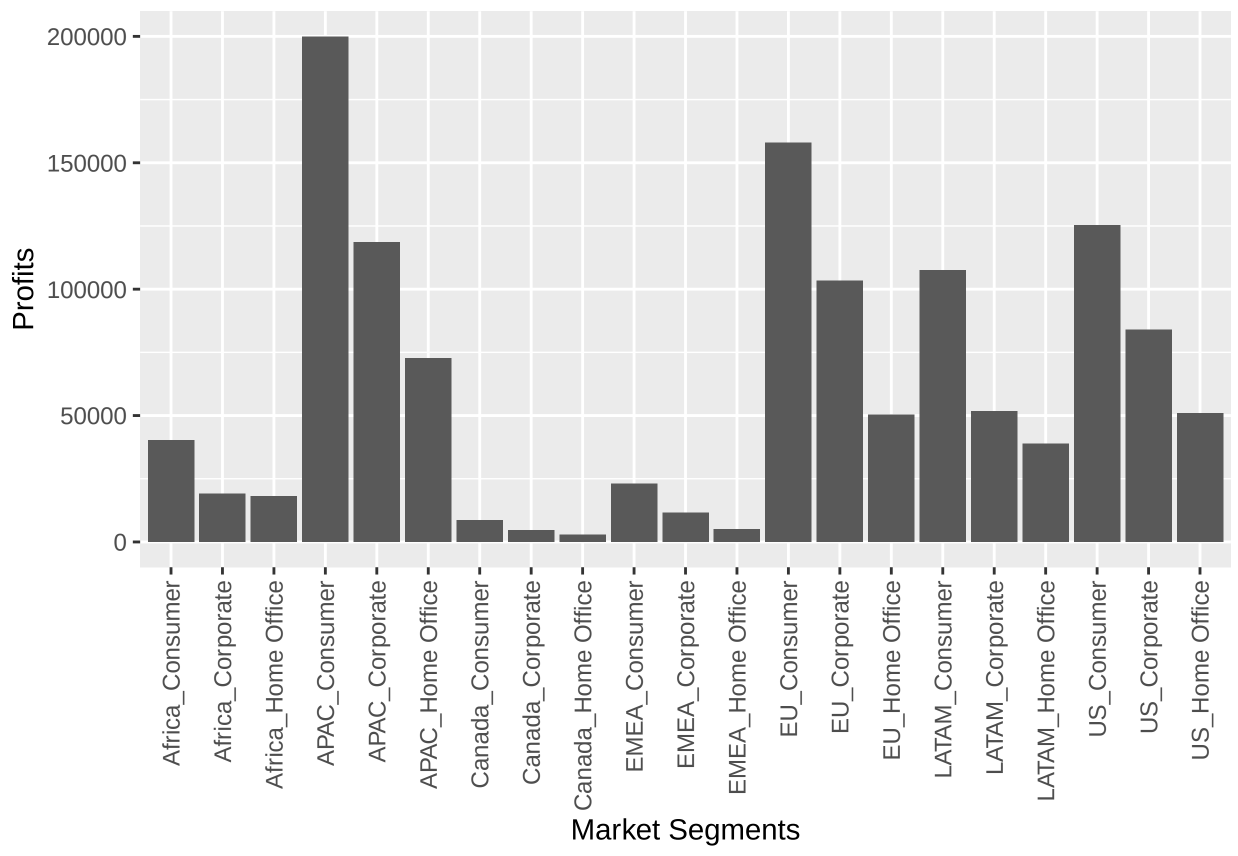


Fig 4.2: Most Profitable Market Segments World wide

From the graph, we could say that *APAC_Consumer* and *EU_Consumer* would be the Most profitable Market Segments. But, we could also look count of orders against the population and see how are these Market Segments really across the entire population.

```r
#Making a dataframe of the different Markets.
Market = c('Africa', 'APAC', 'Canada', 'EMEA', 'EU', 'LATAM', 'US')
population_in_million = c(1216, 4300, 38.01, 746.4, 447.7, 662, 329.5)

population_df <- data.frame(Market, population_in_million)


market <- global %>%
  group_by(Market) %>%
  summarise(profits = sum(Profit))

market_and_population = merge(market, population_df, by="Market")

market_and_population <- market_and_population %>%
  mutate(profit_per_million = profits/population_in_million)

ggplot(market_and_population,aes(Market, profit_per_million))+
  geom_bar(position = "dodge", stat="identity")+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  labs(x="Market", y="Profits per million")
```

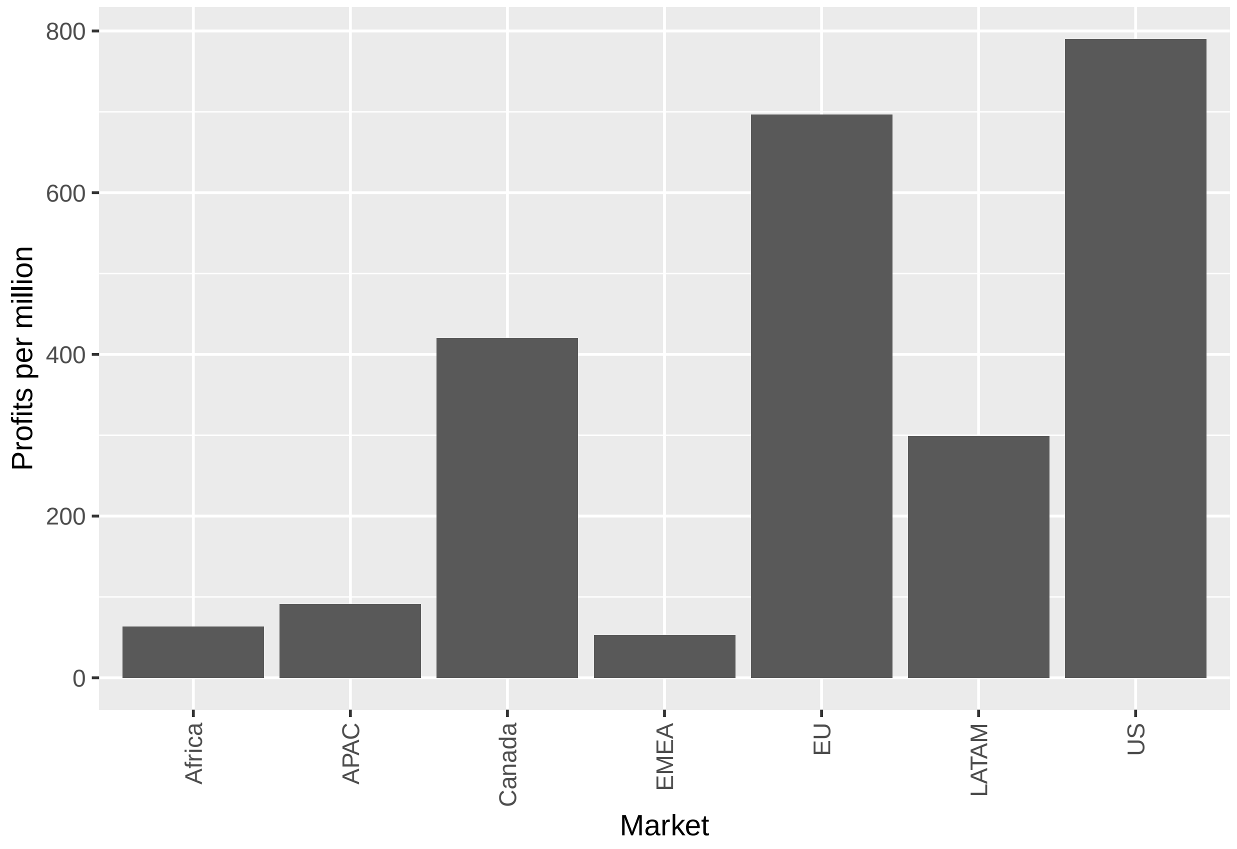


Fig 4.3: Most Profitable Markets populaton per million wise.

In the previous graph, we saw that Canada was doing bad. But when considering the population of the Markets and plotting against the profits per million, we find that Canada is doing good and APAC is actually not doing good in Profits per million. Although EU has been doing good in both the graphs. We also see that US is generating the most profits per million for the Store.

So now that we know that EU and US are the most profitable market segments, let's look at during what time of the months are these markets generating the most profits.

```
us_market <- global %>%
  filter(Country == "United States") %>%
  dplyr::select(Market, Segment, order_month_year, Profit, Sales, State, Sub.Category)

us_market <- separate(us_market, order_month_year, c("Month", "Year"))


us_market_over_time <- us_market %>%
  group_by(Month) %>%
  summarise(Profits = sum(Profit))

us_profit_plot <- ggplot(us_market_over_time, aes(x=Month, y=Profits, group = 1)) +
  geom_line( color="steelblue") +
  geom_point() +
  xlab("Months") +
  theme(axis.text.x=element_text(angle=60, vjust=0.5, hjust=0.5))
us_profit_plot
```

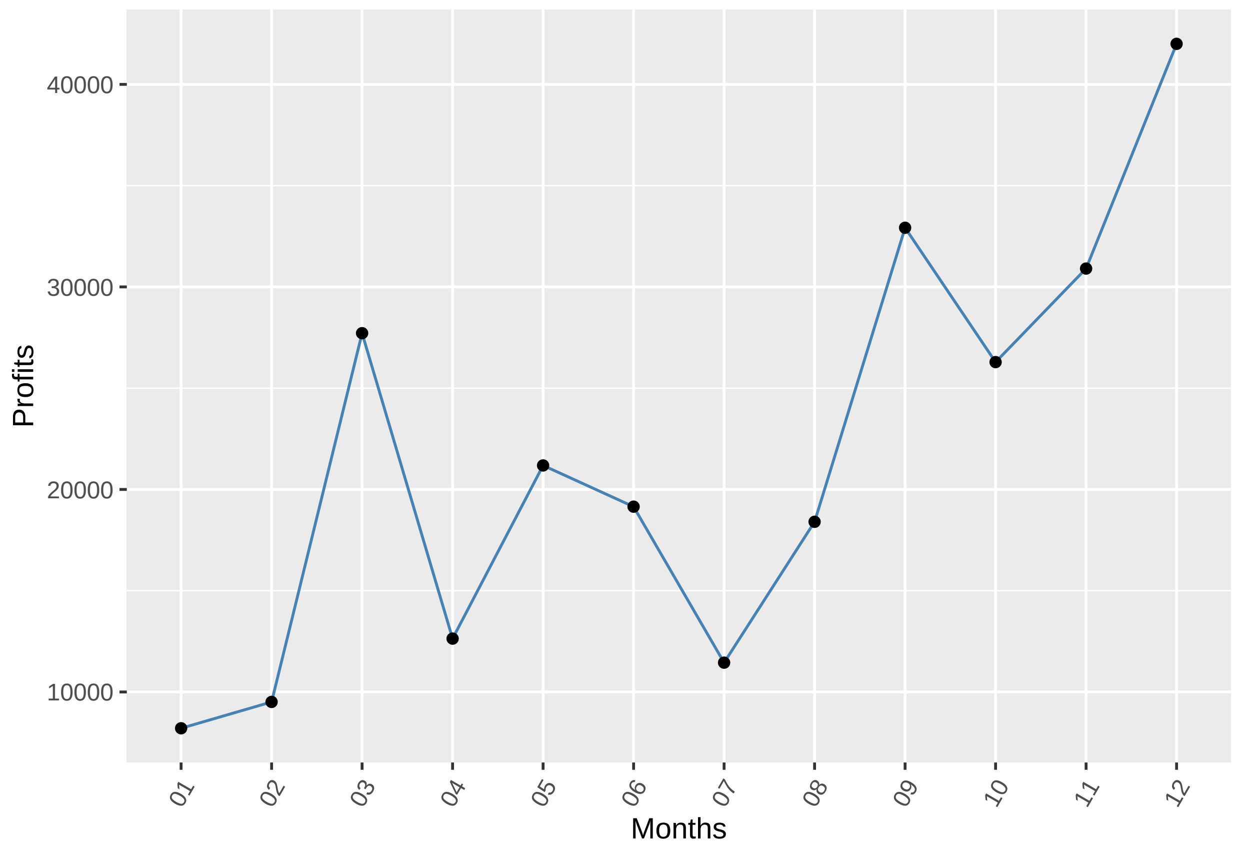


Fig 4.4: Trend of the US Market across the year.

We can see that the US Market becomes the most profitable during the end of the year. This could be because of the Thanksgiving holiday and the Christmas holidays.

Is this the same trend for the EU Market?

```r
EU_market <- global %>%
  filter(Market == "EU") %>%
  dplyr::select(Market, Segment, order_month_year, Profit, Sales, State, Sub.Category)

EU_market <- separate(EU_market, order_month_year, c("Month", "Year"))


EU_market_over_time <- EU_market %>%
  group_by(Month) %>%
  summarise(Profits = sum(Profit))

EU_profit_plot <- ggplot(EU_market_over_time, aes(x=Month, y=Profits, group = 1)) +
  geom_line( color="steelblue") +
  geom_point() +
  xlab("Months") +
  theme(axis.text.x=element_text(angle=60, vjust=0.5, hjust=0.5))
EU_profit_plot
```

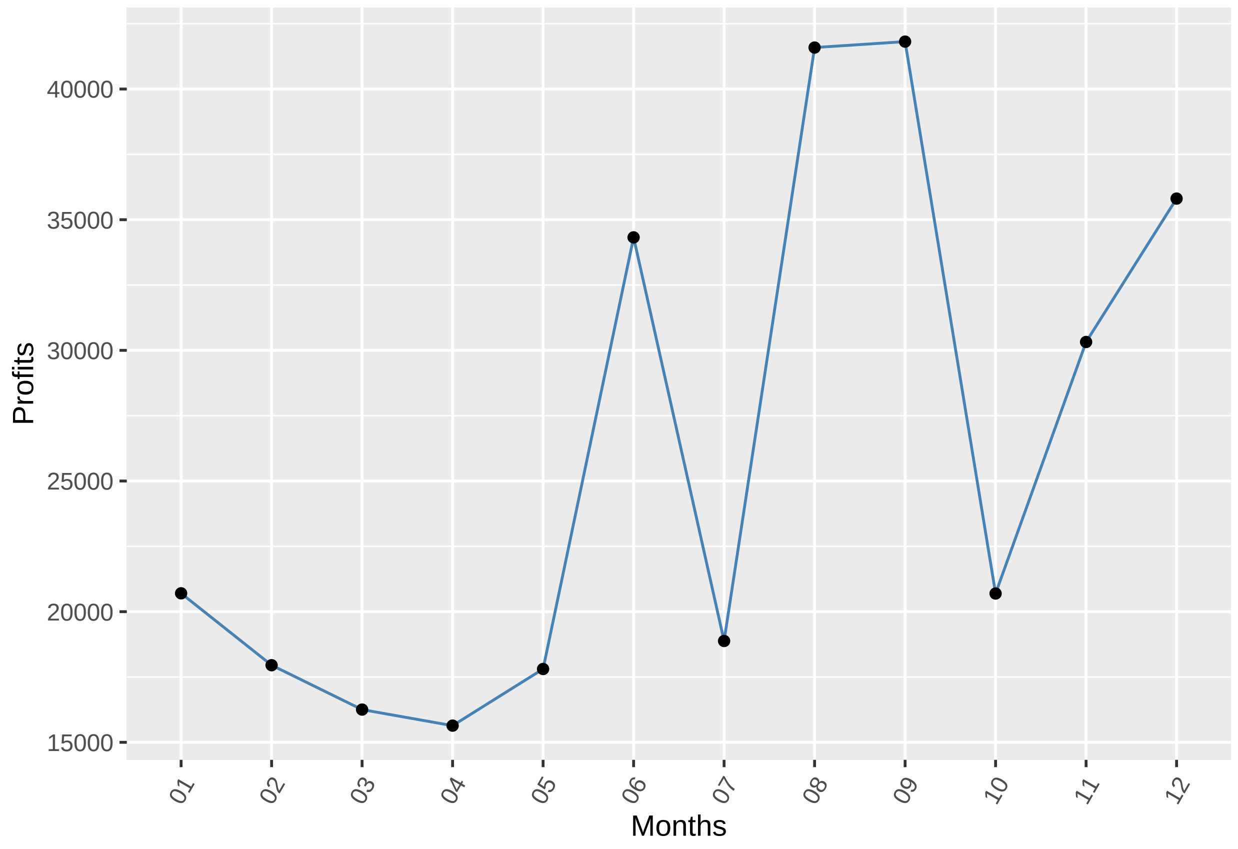


Fig 4.5: Trend of the EU Market across the year.

The EU Market generates the most profits during August and September. This could be due to the opening of schools and colleges which contribute to people moving and buying lots of stuffs for move-ins. The profits take a dip during October but there is again an increase during the Christmas breaks in EU as well. In fact, the EU Market makes more profit during December than the US Market.

Now that we have looked at which Markets are Profitable, Let's look at which Sub Categories of products are the most profitable and how are those products doing over time.

```
sub_category_sales <- global %>%
  group_by(Sub.Category) %>%
  summarise(sales = sum(Sales),
            profit = sum(Profit),
            quantity = sum(Quantity))

ggplot(sub_category_sales,aes(Sub.Category, profit))+
  geom_bar(position = "dodge", stat="identity")+
  theme(axis.text.x = element_text(angle = 90, vjust=0.3, hjust=1)) +
  labs(x = "Sub Categories of Products", y="Profits")
```

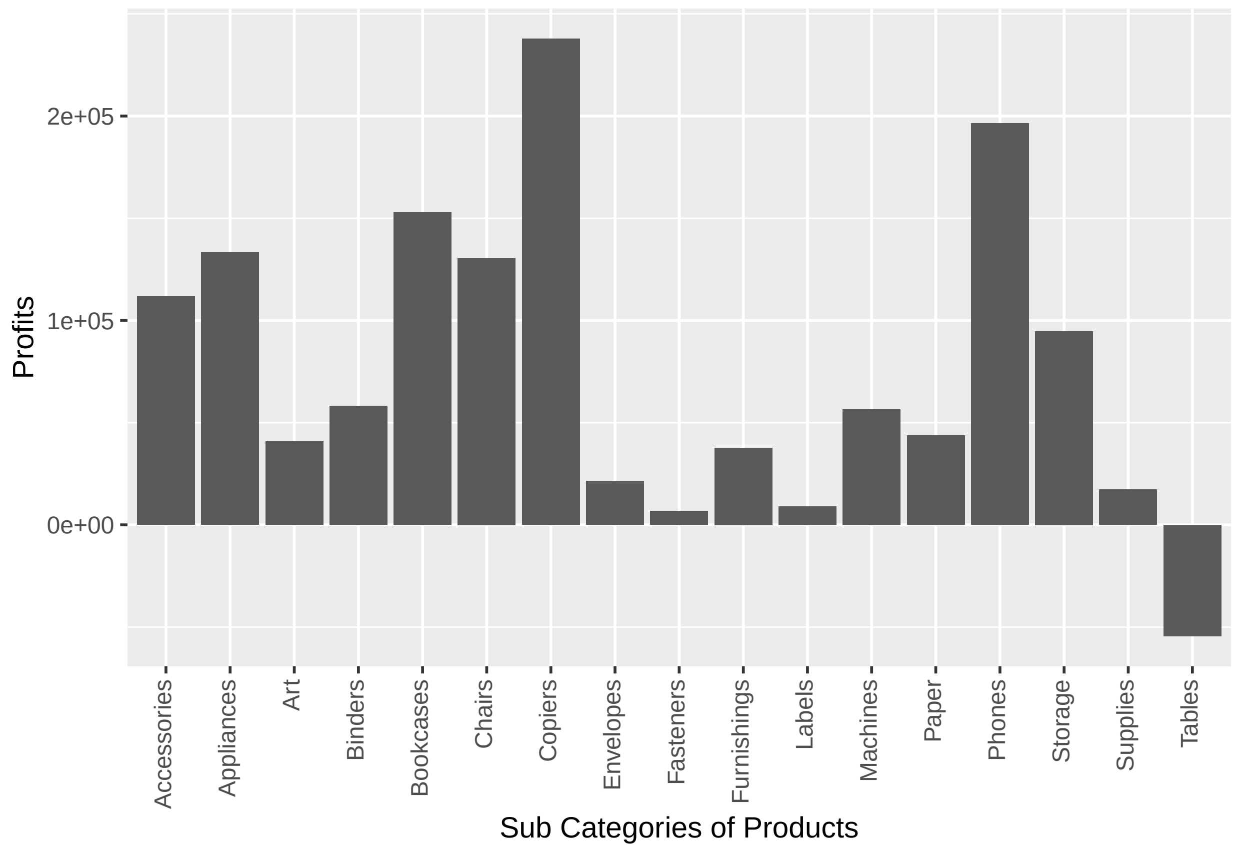


Fig 4.6: Most profitable sub-categories of products.

Overall, across the globe, Copiers, Phones, Bookcases, Accessories are the most profitable. Tables have been incurring losses world wide.

If we disregard, Tables, let's look at how much each sub category is contributing towards the total profit of the store.

```
no_table_sub_category <- sub_category_sales[-c(17),]

no_table_sub_category <- no_table_sub_category %>%
  mutate(prop = profit / sum(profit)*100)

dataset <- no_table_sub_category %>%
  mutate(Sub.Category = fct_reorder(Sub.Category, prop))


ggplot(dataset, aes(x = "", y = prop, fill = Sub.Category)) +
  geom_col(width = 1) +
  coord_polar(theta = "y") +
  xlab(NULL)
```

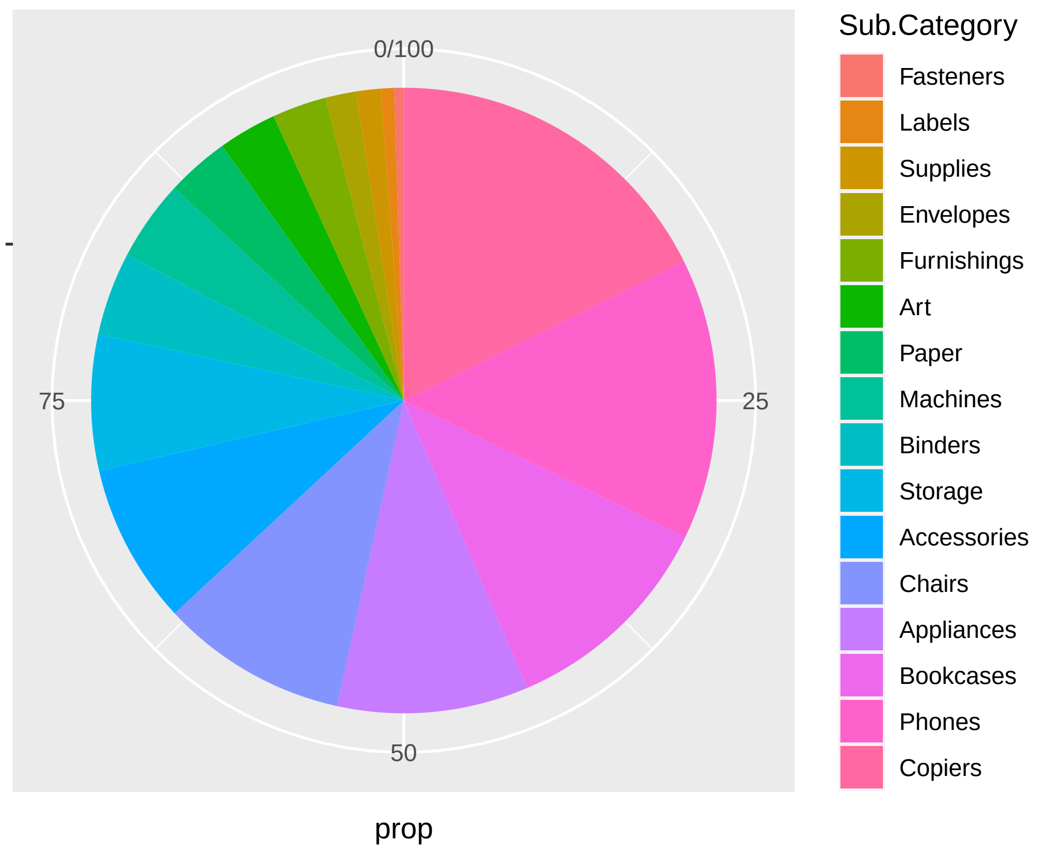


Fig 4.7: Contribution of each sub-category towards the total profit.

Phones, Copiers and Bookcases have the major contribution towards the overall Profit of the Store.

Let's take a look at the which sub-categories are the most profitable in the EU and US Markets and we shall also see during which months are these sub-categories the most profitable.

```
sub_category_EU <- global %>%
  filter(Market == "EU") %>%
  dplyr::select(Sub.Category, Market_Seg, order_month_year, Sales, Profit) %>%
  group_by(Sub.Category) %>%
  summarise(sales = sum(Sales),
            profits = sum(Profit))

ggplot(sub_category_EU,aes(Sub.Category, profits))+
  geom_bar(position = "dodge", stat="identity")+
  theme(axis.text.x = element_text(angle = 90, vjust=0.3)) +
  labs(x="Sub Categories", y="Profits")
```

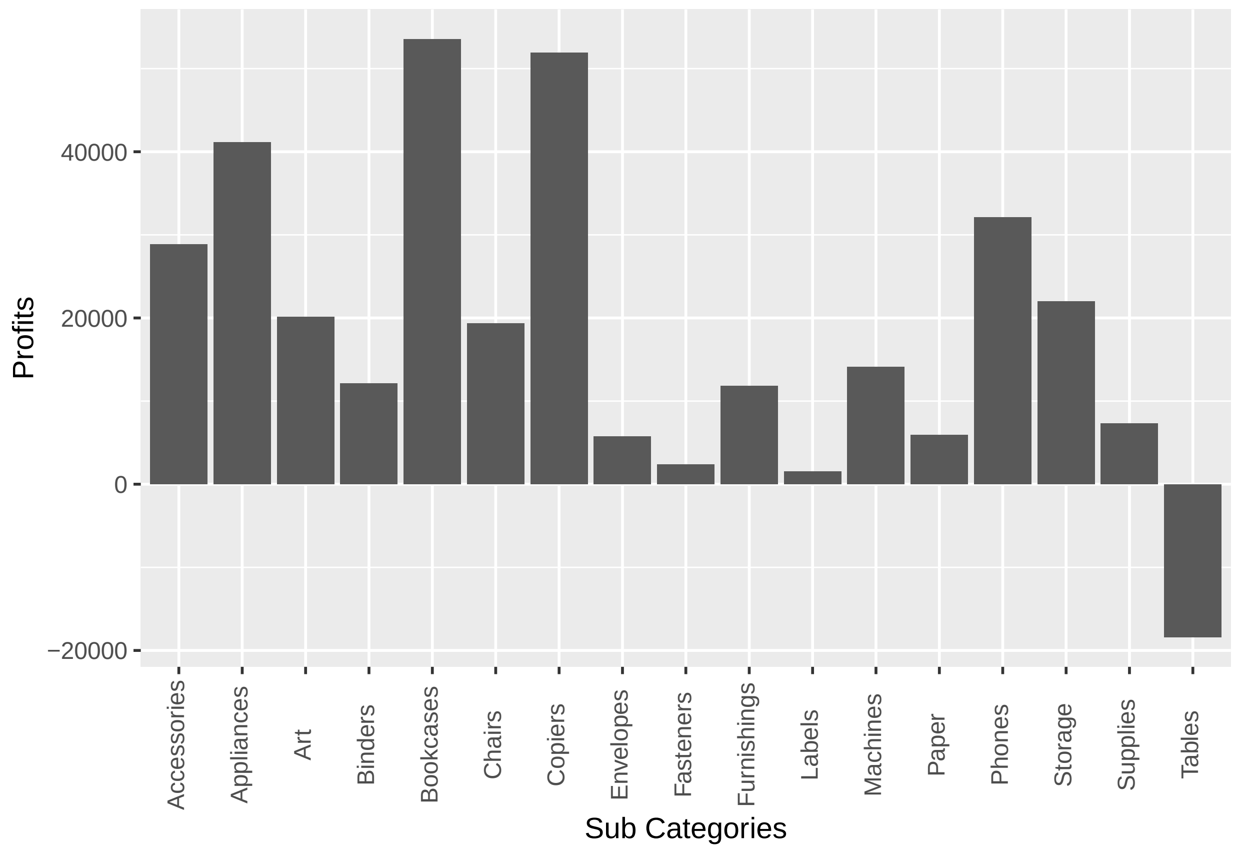


Fig 4.8: Most profitable sub-categories in the EU Market

We can see that Appliances, Bookcases and Copiers are the Most profitable in the EU Market. Let's look at how these sub-categories are doing over time in the EU Market.

```
Appliances_EU <- global %>%
  filter(Market == "EU") %>%
  dplyr::select(Sub.Category, Market_Seg, order_month_year, Sales, Profit)%>%
  filter(Sub.Category == "Appliances")

Appliances_EU <- separate(Appliances_EU, order_month_year, c("Month", "Year"))

Appliances_EU_Monthly_Sales <- Appliances_EU %>%
  group_by(Month) %>%
  summarise(sales = sum(Sales),
            profits = sum(Profit))
```

```r
##############################################################################

Bookcases_EU <- global %>%
  filter(Market == "EU") %>%
  dplyr::select(Sub.Category, Market_Seg, order_month_year, Sales, Profit)%>%
  filter(Sub.Category == "Bookcases")

Bookcases_EU <- separate(Bookcases_EU, order_month_year, c("Month", "Year"))

Bookcases_EU_Monthly_Sales <- Bookcases_EU %>%
  group_by(Month) %>%
  summarise(sales = sum(Sales),
            profits = sum(Profit))
##############################################################################

Copiers_EU <- global %>%
  filter(Market == "EU") %>%
  dplyr::select(Sub.Category, Market_Seg, order_month_year, Sales, Profit)%>%
  filter(Sub.Category == "Copiers")

Copiers_EU <- separate(Copiers_EU, order_month_year, c("Month", "Year"))

Copiers_EU_Monthly_Sales <- Copiers_EU %>%
  group_by(Month) %>%
  summarise(sales = sum(Sales),
            profits = sum(Profit))
##############################################################################

p <- ggplot(Appliances_EU_Monthly_Sales, aes(x=Month, y=profits, group = 1)) +
  geom_line( color="steelblue") +
  geom_point() +
  xlab("Monthly Sales of Appliances in EU") +
  theme(axis.text.x=element_text(angle=60, hjust=1))

q <- ggplot(Bookcases_EU_Monthly_Sales, aes(x=Month, y=profits, group = 1)) +
  geom_line( color="green") +
  geom_point() +
  xlab("Monthly Sales of Bookcases in EU") +
  theme(axis.text.x=element_text(angle=60, hjust=1))

r <- ggplot(Copiers_EU_Monthly_Sales, aes(x=Month, y=profits, group = 1)) +
  geom_line( color="red") +
  geom_point() +
  xlab("Monthly Sales of Copiers in EU") +
  theme(axis.text.x=element_text(angle=60, hjust=1))

ggarrange(p, q, r, ncol = 1, nrow = 3)
```

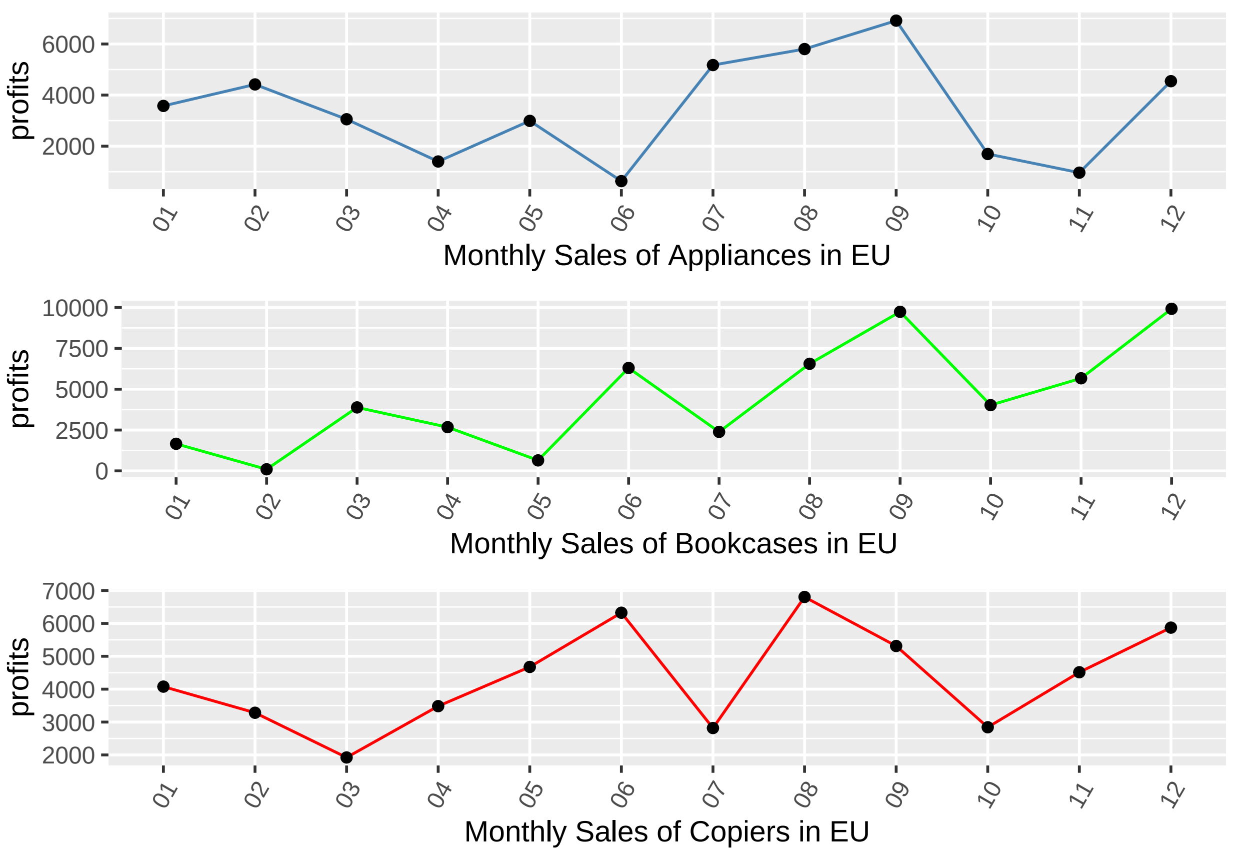


Fig 4.9: Trend of the 3 most profitable sub-categories of products in EU.

Let's look at the most profitable sub-category in USA.

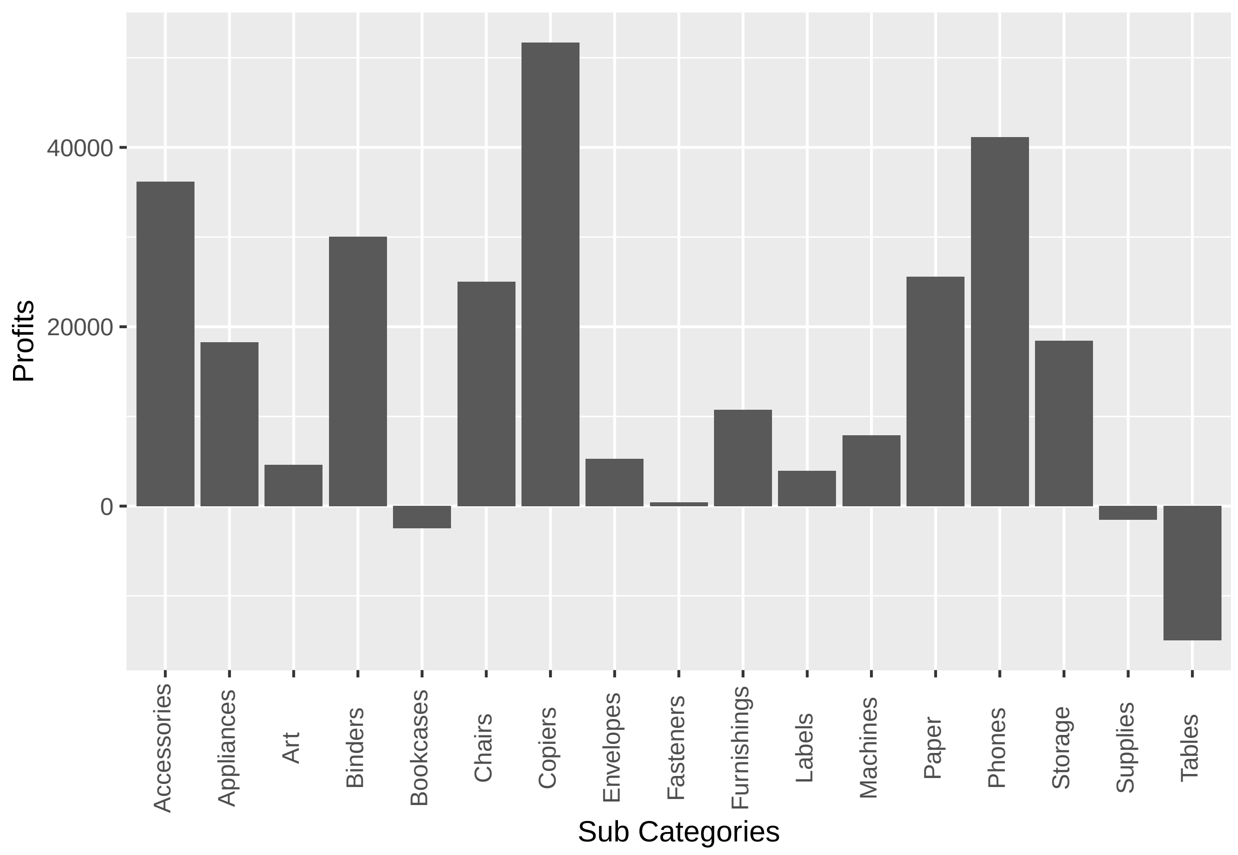


Fig 4.10: Most profitable sub-categories in the US Market

The store is incurring a losses in Bookcases, Supplies and Tables. Copiers, Accessories and Phones are the most profitable in USA. Let’s take a look at their trend with time also.

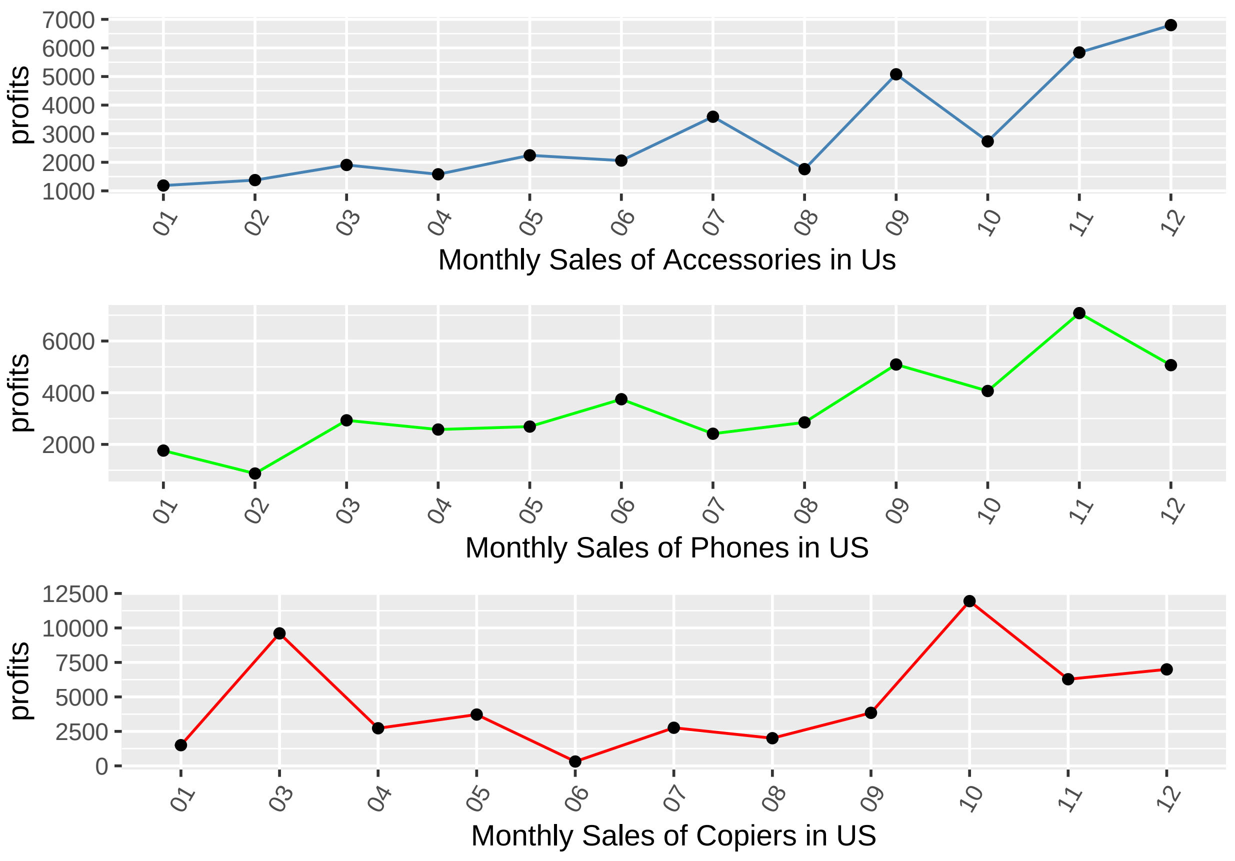


Fig 4.11: Trend of the 3 most profitable sub-categories of products in US.

5. Modelling

Since i am confident the EU Market has been the most profitable globally and profit per million wise, I want to focus on modelling the EU Market for this project. An Extension of this project would also be modelling for the APAC Market which is also profitable worldwide and the US Market which is profitable population per million wise.

Let's aggregate the value for 21 market segments by the order month and year.

```
segmented <- global %>% mutate(month = as.numeric(format(Order.Date, "%m")),
                               year = as.numeric(format(Order.Date, "%Y"))) %>%
  group_by(Market_Seg, year, month) %>%
  summarise(monthly_sales = sum(Sales),
            monthly_qty = sum(Quantity),
            monthly_profit = sum(Profit))
```

Adding a new column in Segmented data frame with the month and year together.

```
segmented$month_year <- paste(segmented$month,segmented$year,sep = "-")
```

Since the EU_Consumer Market segment is the most profitable, I am filtering out that Market segment.

```
EU_Consumer <- segmented %>%
  filter(Market_Seg == "EU_Consumer")

EU_Consumer$month_year <- seq(1:48)
```

Creating a generalize function for smoothing

```
smoothing_fun <- function(x){
  # Smoothing the series - Moving Average Smoothing
  w <- 1
  smoothedseries <- stats::filter(x,
                                  filter=rep(1/(2*w+1),(2*w+1)),
                                  method='convolution', sides=2)
  # Smoothing left end of the time series
  diff <- smoothedseries[w+2] - smoothedseries[w+1]
  for (i in seq(w,1,-1)) {
    smoothedseries[i] <- smoothedseries[i+1] - diff
  }

  # Smoothing right end of the time series
  n <- length(x)
  diff <- smoothedseries[n-w] - smoothedseries[n-w-1]
  for (i in seq(n-w+1, n)) {
    smoothedseries[i] <- smoothedseries[i-1] + diff
  }
  return(smoothedseries)
}
```

Upon receiving feedback from the Professor regarding that my function was smoothing the data too much, i tried different combinations to try to consider the extreme peaks in the time-series. However, this turned out to be the best combination as the final predictions were the most accurate.

5.1 Modelling on EU Consumer Sales Data

5.1.1 General Additive Model

Training the Model on first 42 months.

I'll create a model on the first 42 Months and then test the model on the last 6 Months.

I first generated a time series object because it will have additional attributes, including time indices for each observation, the sampling frequency and time increment between observations, and the cycle length for periodic data.

The advantage of creating and working with time series objects of the ts class is that many methods are available for utilizing time series attributes, such as time index information.

After plotting the time series, we can see that there is a slight upward trend in the sales.

To model the seasonality, sinusoidal function was used on the Month column simply because thanks to Fourier, we can write any wave as an infinite sum of sine and cosine waves.

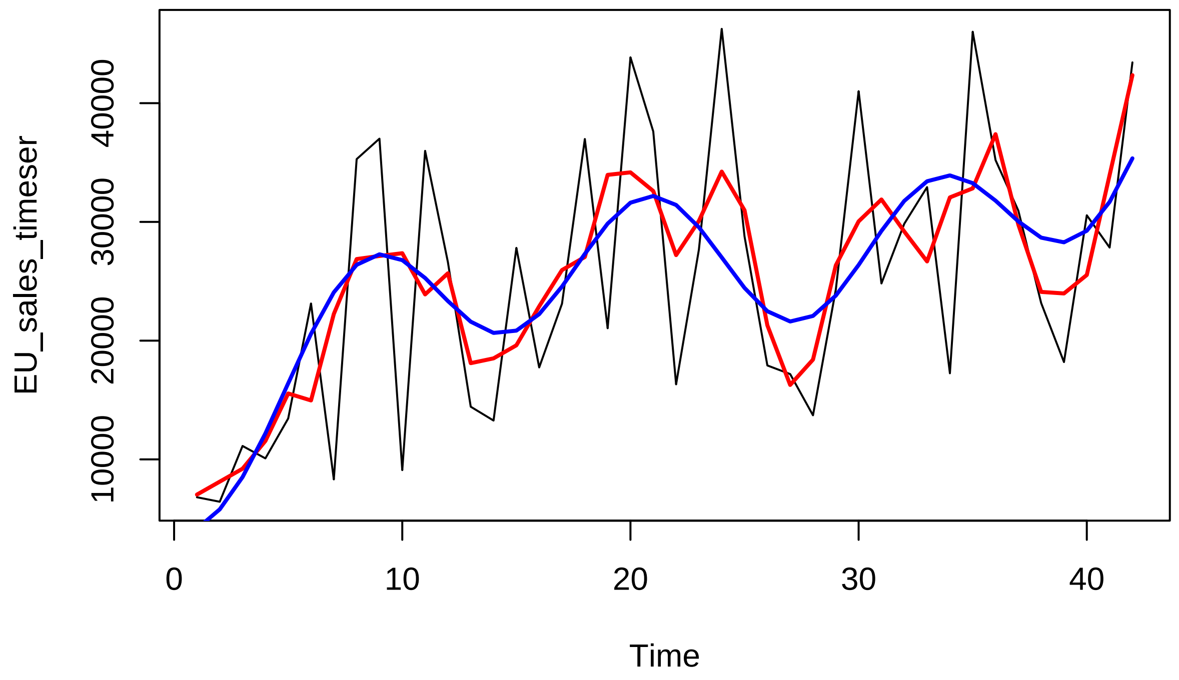
I did a linear model fit on the Smoothed Sales Data frame. Then i predicted the global EU sales by using the linear model fitted sales on the 42 months. Those predictions can be seen by the blue line against the red line on the graph (red line being the Smoothed Sales).

Then i computed the residuals. Plotting the residuals, seems like the residuals are stationary but we verify that by doing the ADF and KPSS tests.

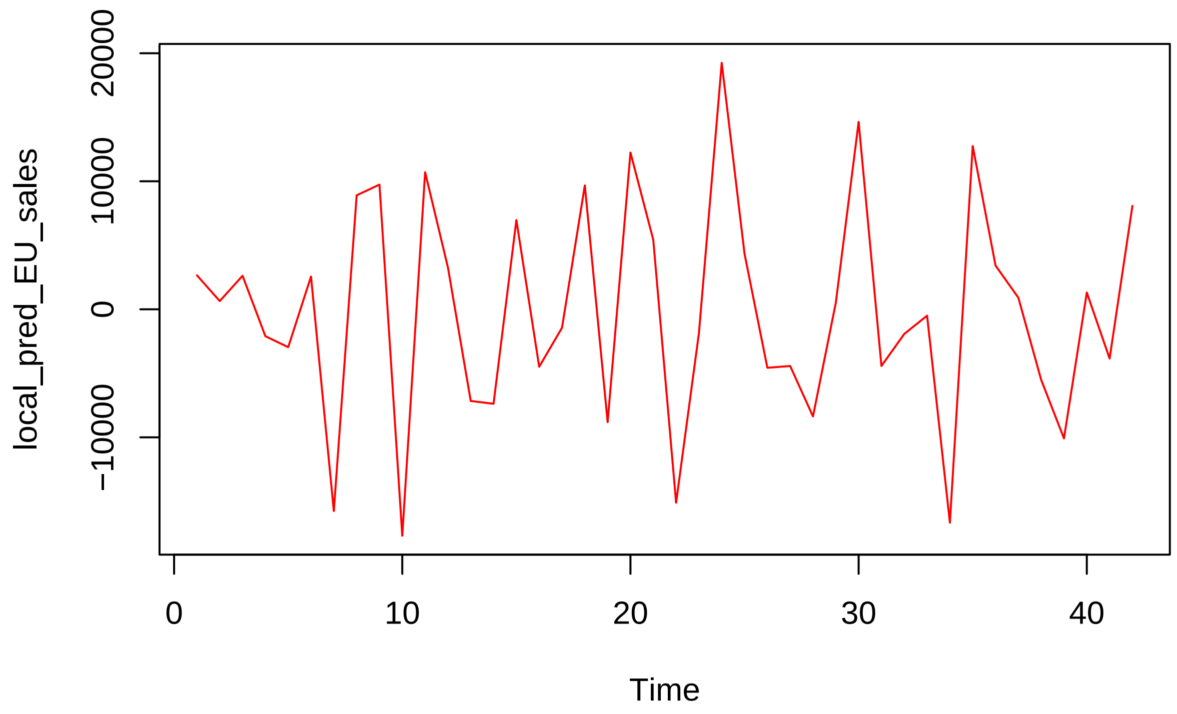
```r
# Creating the total time series
EU_Con_sales_ts <- ts(EU_Consumer$monthly_sales)

# Dividing the train and test data
EU_Con_sales_ts_indata <- EU_Consumer[1:42,]
EU_Con_sales_ts_outdata <- EU_Consumer[43:48,]


#Plotting the time series

EU_sales_timeser <- ts(EU_Con_sales_ts_indata$monthly_sales)
plot(EU_sales_timeser)


#Smoothing the EU_Consumer_sales
smoothed_EU_sales <- smoothing_fun(EU_sales_timeser)

#Plotting the smoothed time series of EU_Consumer_sales
EU_sales_timevals_in <- EU_Con_sales_ts_indata$month_year
lines(smoothed_EU_sales, col="red", lwd=2)


#Converting the smoothed time series into a data frame
EU_sales_smoothed_df <- as.data.frame(cbind(EU_sales_timevals_in,
                                            as.vector(smoothed_EU_sales)))

colnames(EU_sales_smoothed_df) <- c('Month','Sales')

#Now, let's fit a additive model with trend and seasonality to the data
#Seasonality will be modeled using a sinusoidal function

lmfit_EU_sales <- lm(Sales ~ sin(0.5*Month) + cos(0.5*Month) + poly(Month,3),
                     data=EU_sales_smoothed_df)
global_pred_EU_sales <- predict(lmfit_EU_sales, Month=EU_sales_timevals_in)
summary(global_pred_EU_sales)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4157   22119   26584   25167   29995   35353
```

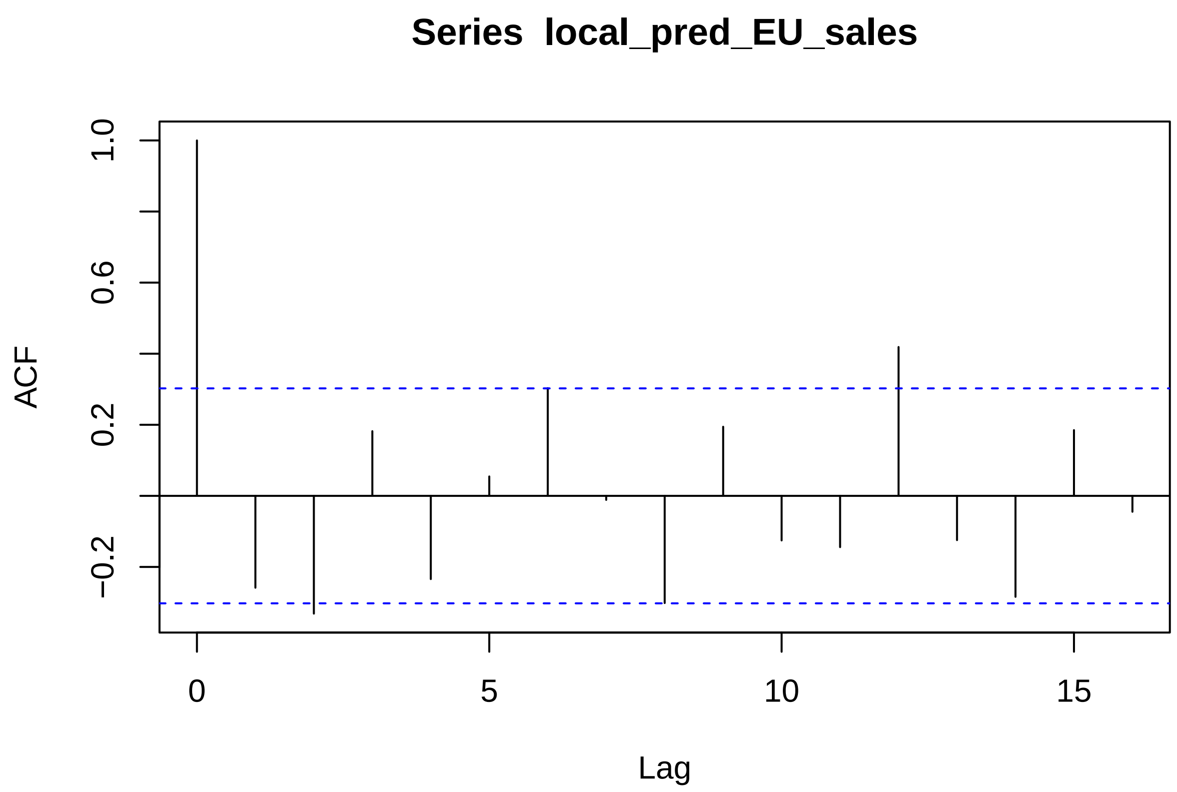
```r
lines(EU_sales_timevals_in, global_pred_EU_sales, col='blue', lwd=2)
```



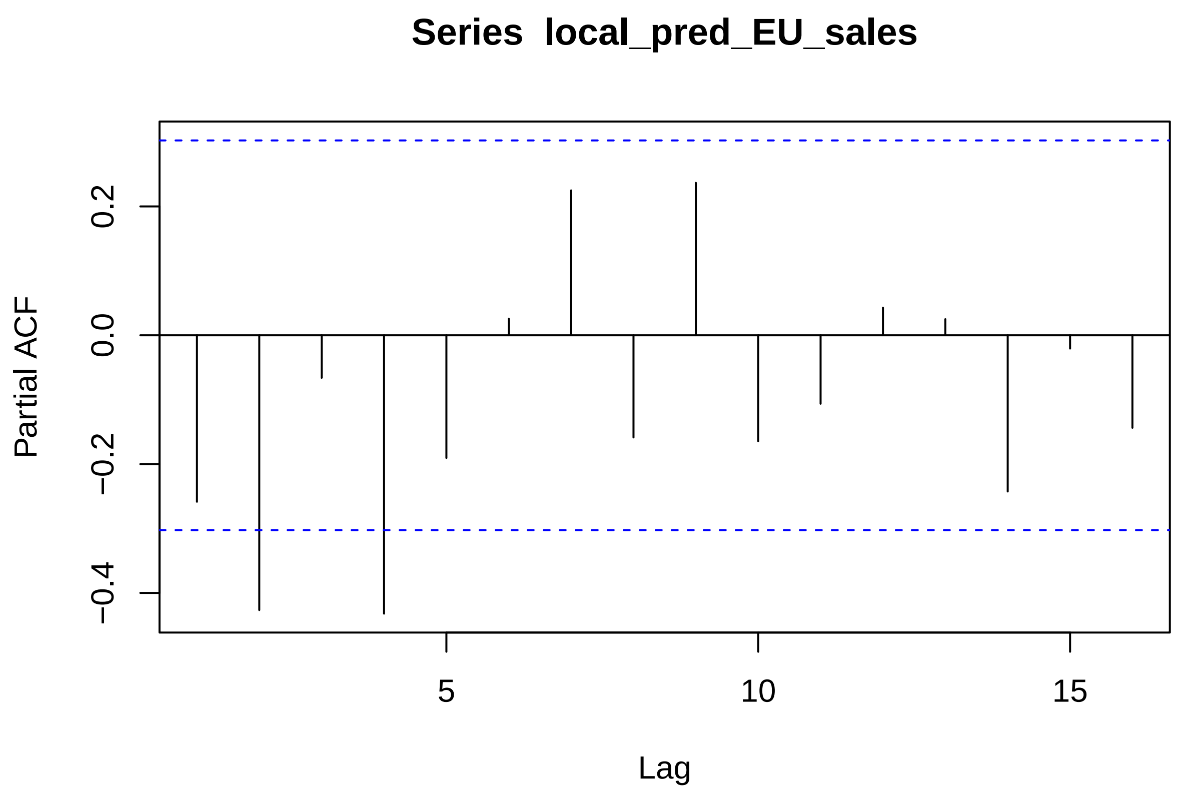
```
local_pred_EU_sales <- EU_sales_timeser - global_pred_EU_sales
plot(local_pred_EU_sales, col='red', type = "l")
```



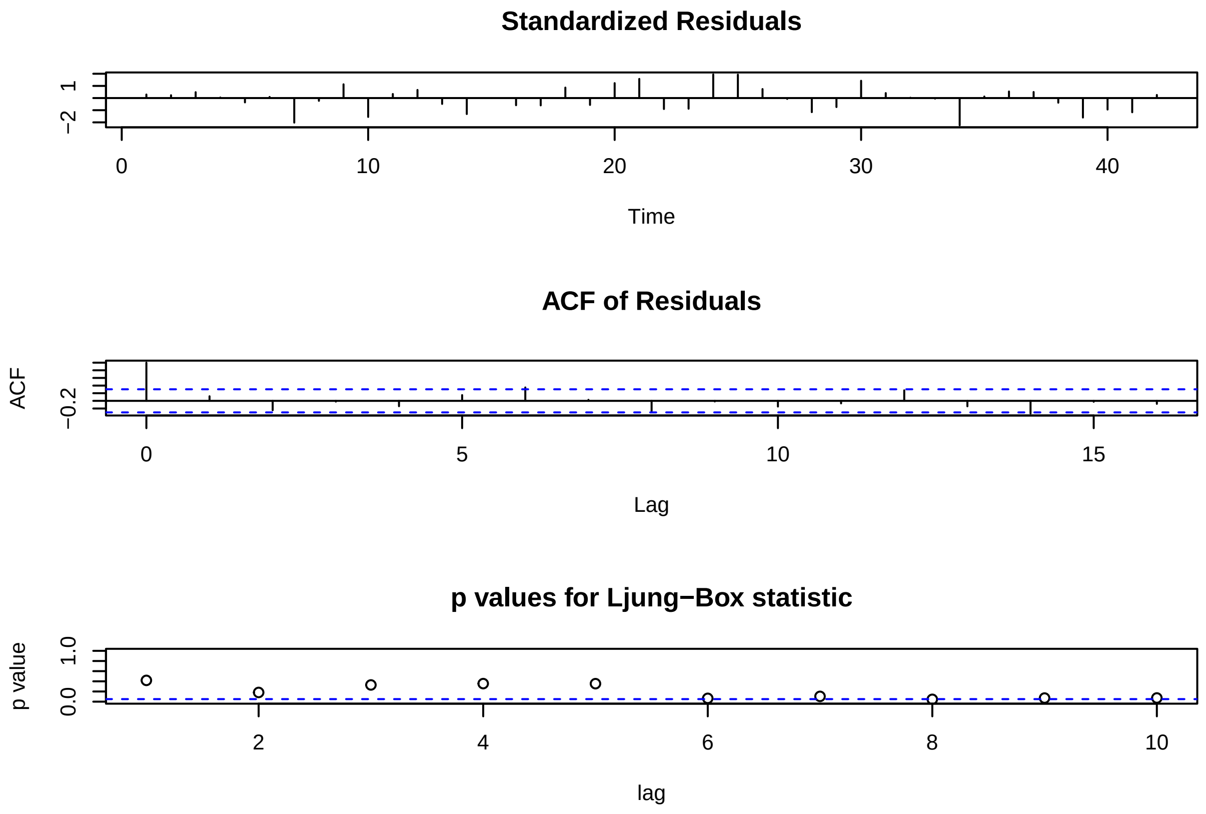
```
# ACF & PACF plot of local_pred
acf(local_pred_EU_sales)
```



```
pacf(local_pred_EU_sales)
acf(local_pred_EU_sales, type="partial")
```



```
#Modelling the local pred with auto arima
armafit_EU_sales <- auto.arima(local_pred_EU_sales)
tsdiag(armafit_EU_sales)
```



```r
armafit_EU_sales
```

```
## Series: local_pred_EU_sales
## ARIMA(0,0,1) with zero mean
##
## Coefficients:
##           ma1
##       -0.6981
## s.e.   0.1247
##
## sigma^2 estimated as 56859376:  log likelihood=-434.4
## AIC=872.8   AICc=873.11   BIC=876.28
```

The "Residuals" in a time series model are what is left over after fitting the model. For many time series models, the residuals are equal to the difference between the observations and the corresponding fitted values.

Residuals are useful in checking whether a model has adequately captured the information in the data. A good forecasting method will yield residuals with the following properties:

- The residuals are uncorrelated. If there is correlation between the residuals, then it would suggest that there is information left in the residuals that the model could use in forecasting
- The residuals have zero mean.

From the ACF plot, we can see that majority of the residuals are below the confidence interval, this could suggest that there is zero correlation. This could mean they are white noise. To check if the residuals are white noise we'll do the ADF and KPSS tests.

Performing ADF and KPSS tests to test the Stationarity of the Residuals

```r
resi_EU_sales <- local_pred_EU_sales-fitted(armafit_EU_sales)

adf.test(resi_EU_sales,alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  resi_EU_sales
## Dickey-Fuller = -3.8883, Lag order = 3, p-value = 0.02364
## alternative hypothesis: stationary
```

```r
kpss.test(resi_EU_sales)
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  resi_EU_sales
## KPSS Level = 0.12773, Truncation lag parameter = 3, p-value = 0.1
```

So, from the above tests, we can say that the residual is stationary.

Testing and Evaluating the model

Let's evaluate the model using Mean Absolute Percentage Error (MAPE) Firstly, we'll make a prediction on the remaining 6 months of our test data

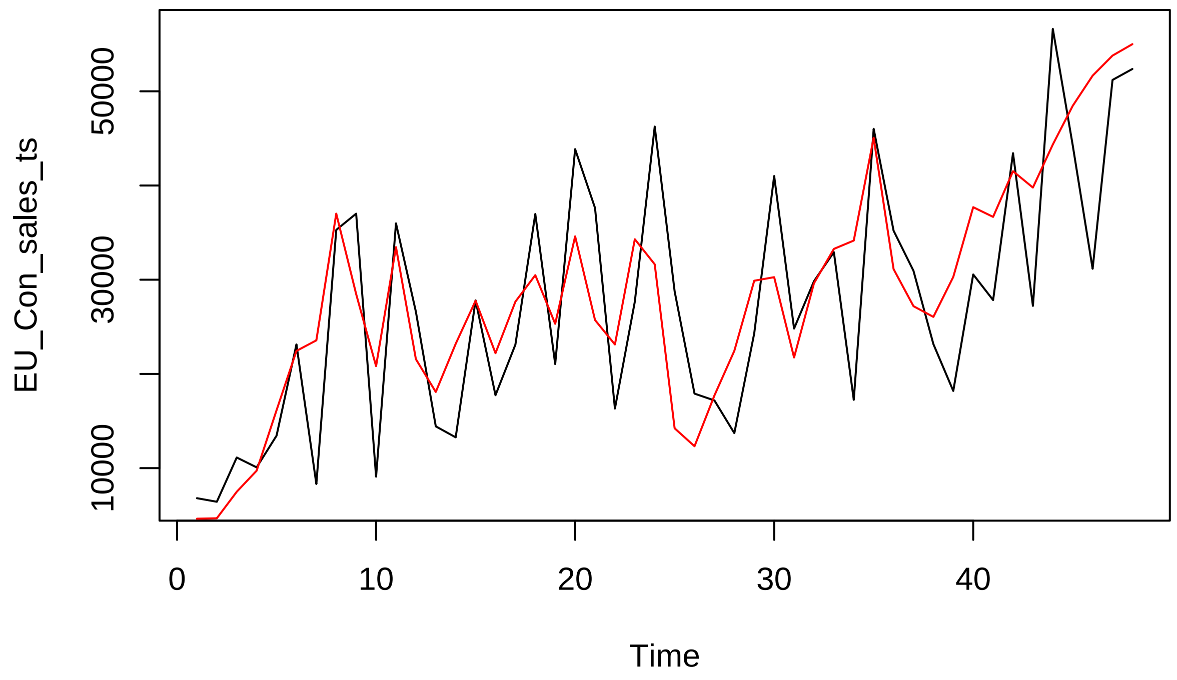
```r
timevals_out_EU_sales <- EU_Con_sales_ts_outdata$month_year

global_pred_out_EU_sales <- predict(lmfit_EU_sales,data.frame(Month =timevals_out_EU_sales))

global_pred_EU_sales_combined <- global_pred_EU_sales+fitted(armafit_EU_sales)

fcast_EU_sales <- global_pred_out_EU_sales


arma_pred=predict(armafit_EU_sales,n.ahead = 6)$pred
fcast_total=fcast_EU_sales+arma_pred

#Now, let's compare our prediction with the actual values, using MAPE

MAPE_class_dec_EU_sales <- accuracy(fcast_total,EU_Con_sales_ts_outdata$monthly_sales)[5]
MAPE_class_dec_EU_sales
```

```
## [1] 24.70029
```

I got at 24% Error which suggests that the model isn't the best but its still alright.

Let's also plot the predictions along with original values, to get a visual feel of the fit

```r
class_dec_pred_EU_sales <- c(ts(global_pred_EU_sales_combined),ts(global_pred_out_EU_sales))
plot(EU_Con_sales_ts, col = "black")
lines(class_dec_pred_EU_sales, col = "red")
```



After plotting predictions along with the original values, we can see that the model hasn't done that bad of a job since it has taken into the considerations some of the peak months of sales. However, after the 40th Month, the model has not shown any major dip in sales whereas in reality there have been some serious dips in sales.

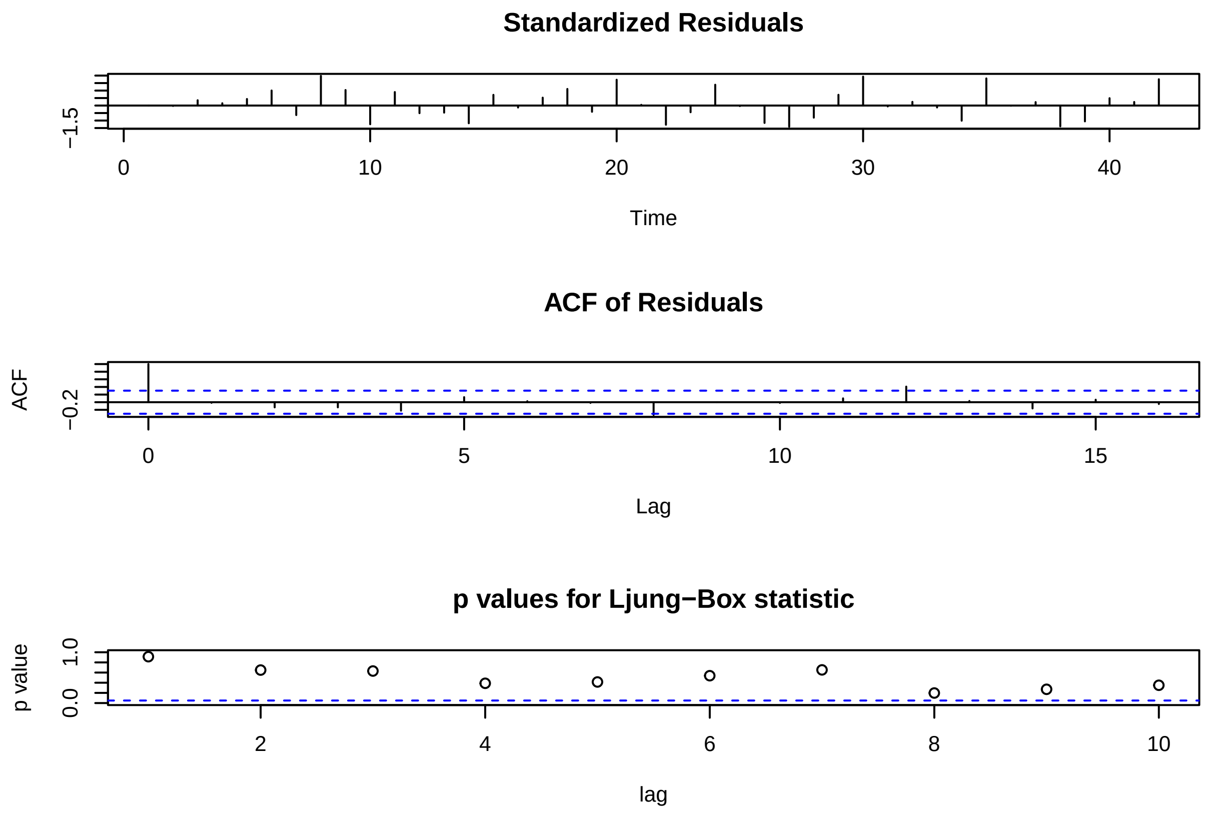
This was a General Additive Model which gave us an accuracy of around 76%. Let's do an ARIMA Model and see how much accuracy we get.

5.1.2 ARIMA Model

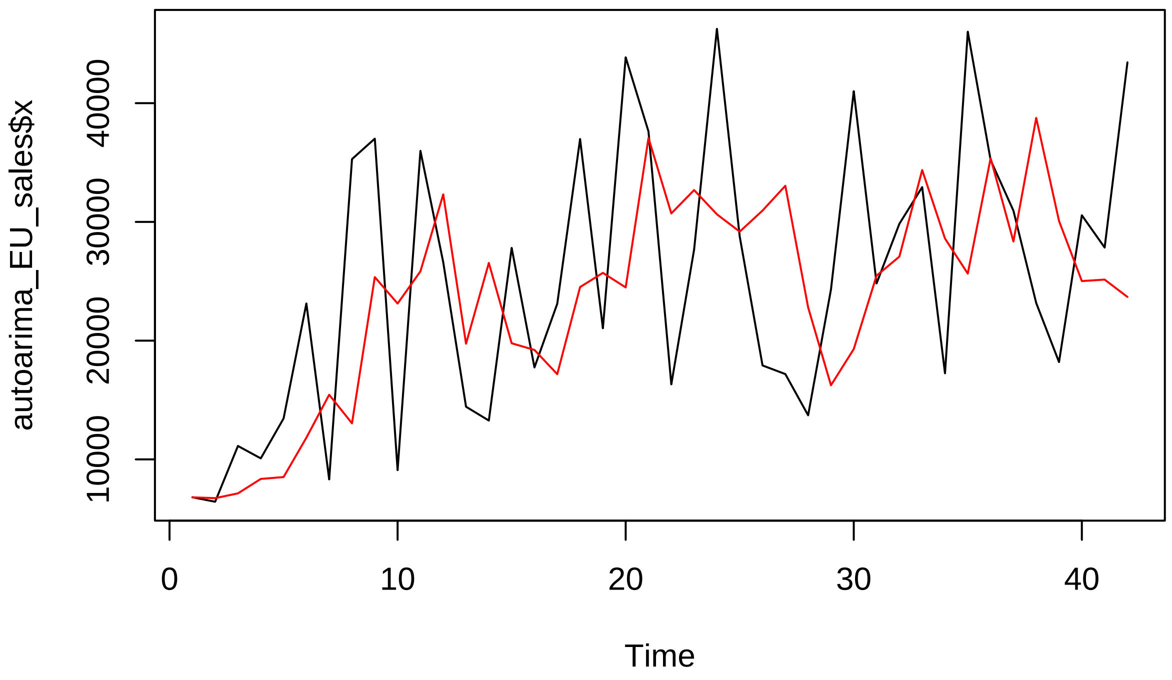
```r
#now let's do an ARIMA fit

autoarima_EU_sales <- auto.arima(EU_sales_timeser)
autoarima_EU_sales
```

```
## Series: EU_sales_timeser
## ARIMA(2,1,0)
##
## Coefficients:
##           ar1      ar2
##         -0.6291  -0.4747
## s.e.    0.1372   0.1338
##
## sigma^2 estimated as 126533396:  log likelihood=-439.96
## AIC=885.91   AICc=886.56   BIC=891.05
```

```
tsdiag(autoarima_EU_sales)
```



```
plot(autoarima_EU_sales$x, col="black")
lines(fitted(autoarima_EU_sales), col="red")
```



```r
#Again, let's check if the residual series is white noise

resi_auto_arima_EU_sales <- EU_sales_timeser - fitted(autoarima_EU_sales)

adf.test(resi_auto_arima_EU_sales,alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  resi_auto_arima_EU_sales
## Dickey-Fuller = -4.4949, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

```r
kpss.test(resi_auto_arima_EU_sales)
```

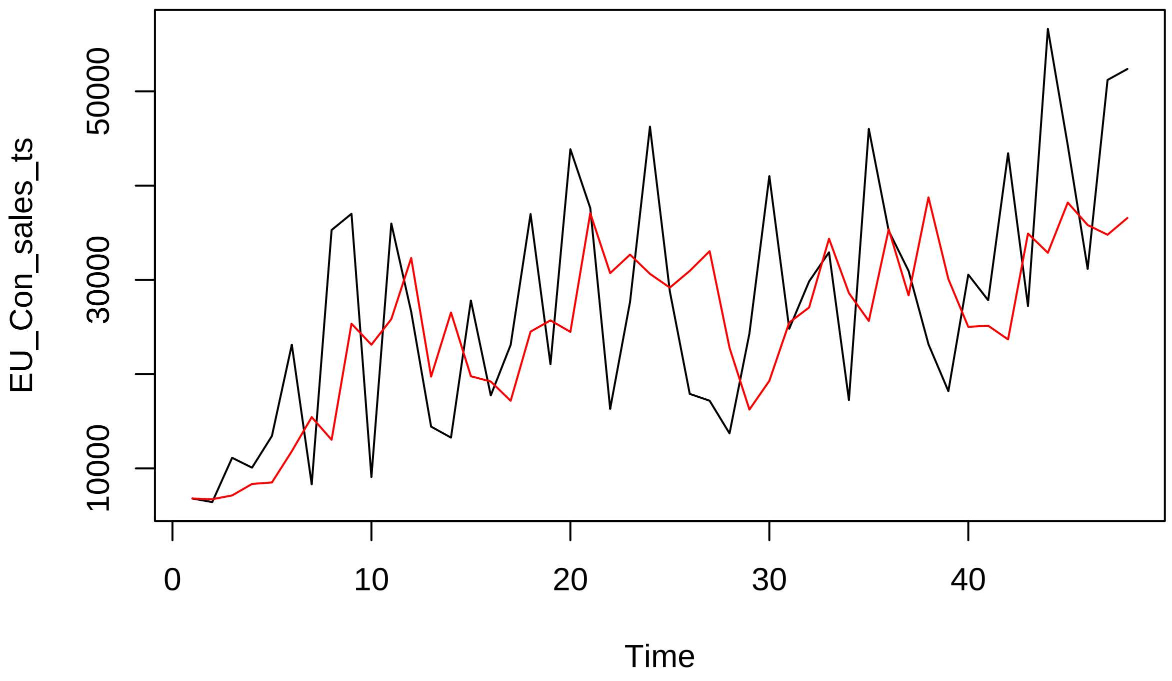
```
##
##  KPSS Test for Level Stationarity
##
## data:  resi_auto_arima_EU_sales
## KPSS Level = 0.065419, Truncation lag parameter = 3, p-value = 0.1
```

```r
# Showing it is staionary

#Also, let's evaluate the model using MAPE
```

```r
fcast_auto_arima_EU_sales <- predict(autoarima_EU_sales, n.ahead = 6)

MAPE_auto_arima_EU_sales <- accuracy(fcast_auto_arima_EU_sales$pred,
                                     EU_Con_sales_ts_outdata$monthly_sales)[5]
MAPE_auto_arima_EU_sales
```

```
## [1] 26.85028
```

```r
#Lastly, let's plot the predictions along with original values, to
#get a visual feel of the fit

auto_arima_pred_EU_sales <- c(fitted(autoarima_EU_sales),ts(fcast_auto_arima_EU_sales$pred))
plot(EU_Con_sales_ts, col = "black")
lines(auto_arima_pred_EU_sales, col = "red")
```



We get a MAPE score of 26% which is slightly higher than the GAM Model.

From the plot of predictions against the original values, we can see that the ARIMA Model has not done a good job in predictions. Unlike the GAM Model, the ARIMA model has never come close to the peaks of the original values. Also, there seems to be a lag in the predictions against the original values. Again, the ARIMA model also has not shown the increase in sales after the 40th month. It has shown a slight increase but the original sales were much higher than what was predicted.

So, from the above two models we can conclude that the General Additive Model is better than the ARIMA Model.

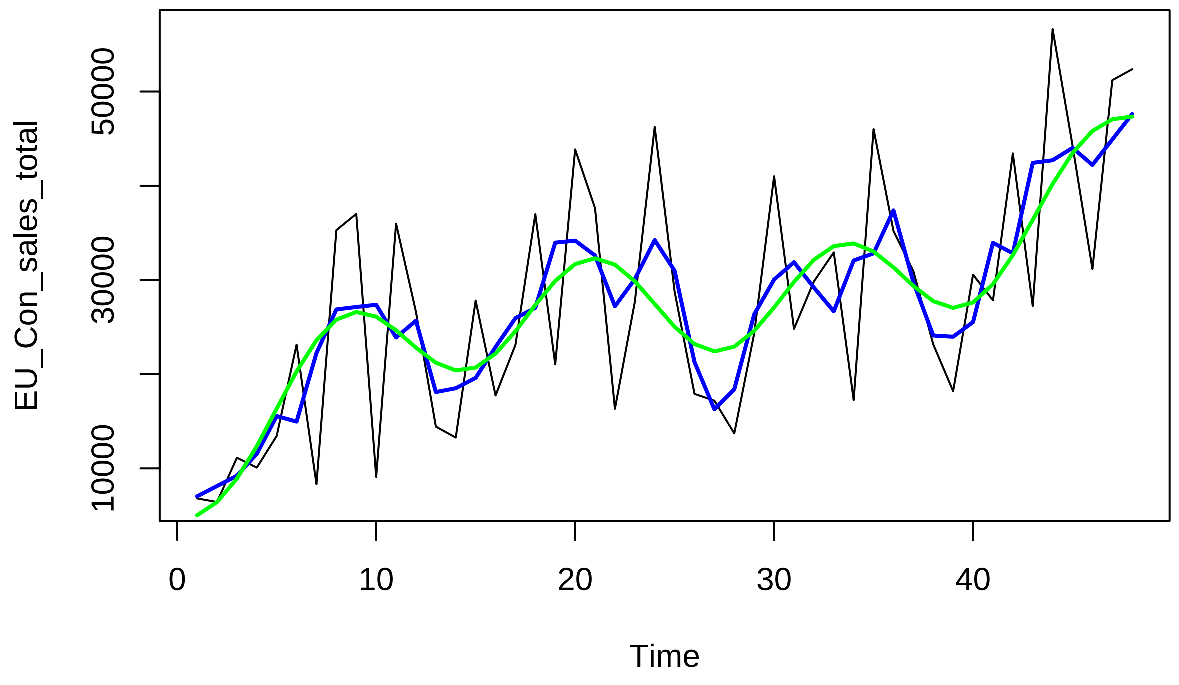
5.1.3 Prediction of sales for the next 3 Months

Now that we have determined which is the better model (GAM Model), we can now predict the estimated sales of the store in the next 3 months.

```
EU_Con_sales_total <- ts(EU_Consumer$monthly_sales)
plot(EU_Con_sales_total)

#Smoothing the total EU sales time series
smoothed_EU_sales_total <- smoothing_fun(EU_Con_sales_total)

# Plotting the smoothed time series of EU_Consumer_sales
EU_Con_sales_timevals_total <- EU_Consumer$month_year
lines(smoothed_EU_sales_total,col="blue",lwd=2)


EU_sales_smootheddf_total <- as.data.frame(cbind(EU_Con_sales_timevals_total,
                                                 as.vector(smoothed_EU_sales_total)))
colnames(EU_sales_smootheddf_total) <- c('Month', 'Sales')


#Now, let's fit a additive model with trend and seasonality to the data
#Seasonality will be modeled using a sinusoid function

lmfit_EU_sales_total <- lm(Sales ~ sin(0.5*Month) + cos(0.5*Month) + poly(Month,3),
                           data=EU_sales_smootheddf_total)
global_pred_EU_sales_total <- predict(lmfit_EU_sales_total, Month=EU_Con_sales_timevals_total)
summary(global_pred_EU_sales_total)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5026   22894   27207   27321   31801   47347
```

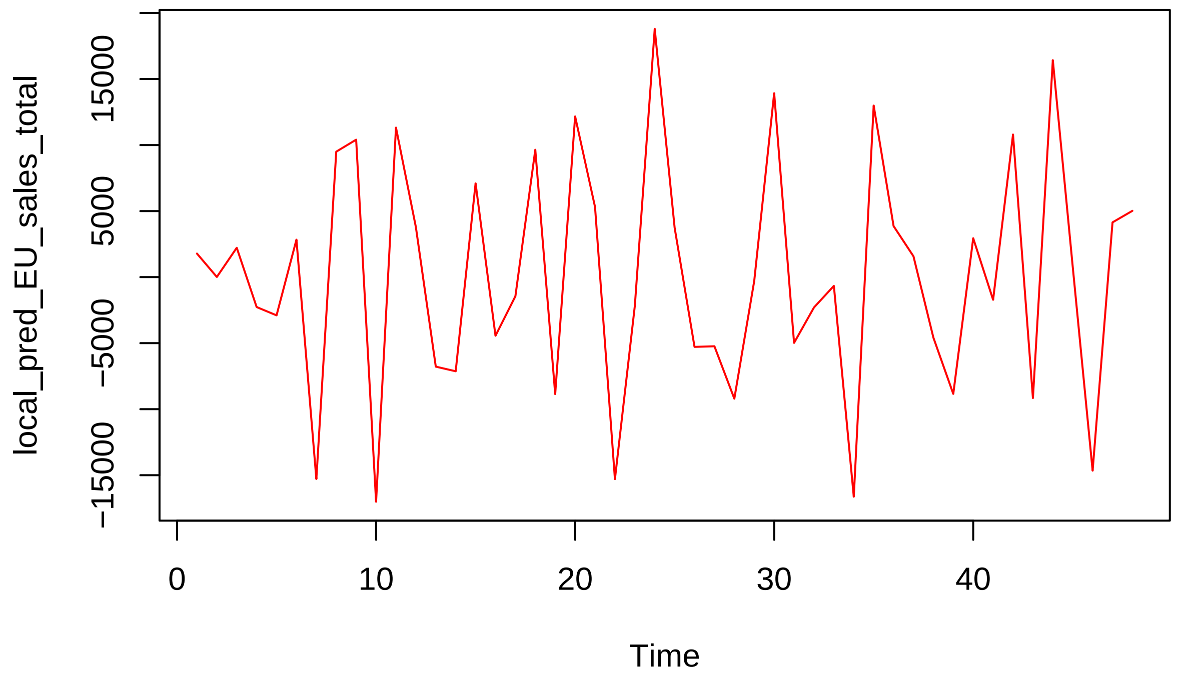
```
lines(EU_Con_sales_timevals_total, global_pred_EU_sales_total, col='green', lwd=2)
```



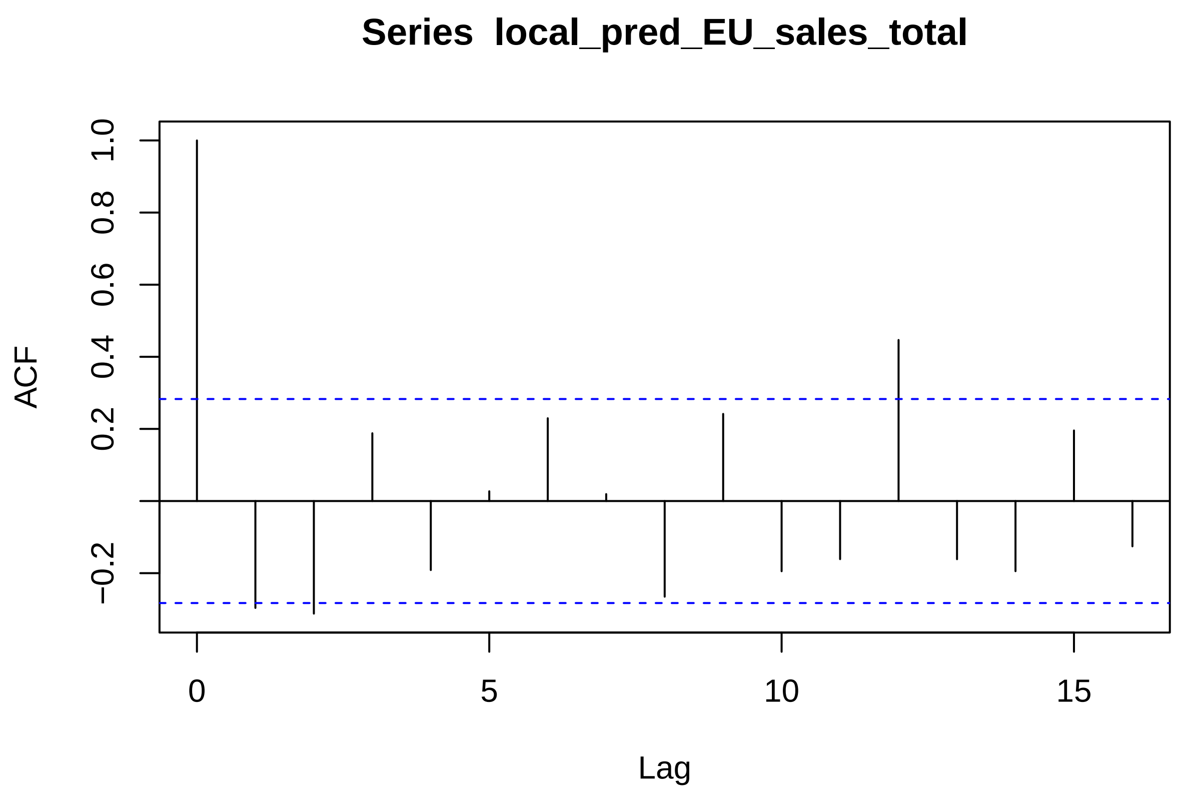
```
future_months <- as.data.frame(c(49:51))
colnames(future_months)<- c("Month")

future_EU_sales <- predict(lmfit_EU_sales_total,newdata=future_months)
future_EU_sales
```

```
##        1        2        3
## 47121.43 46975.66 47532.69
```

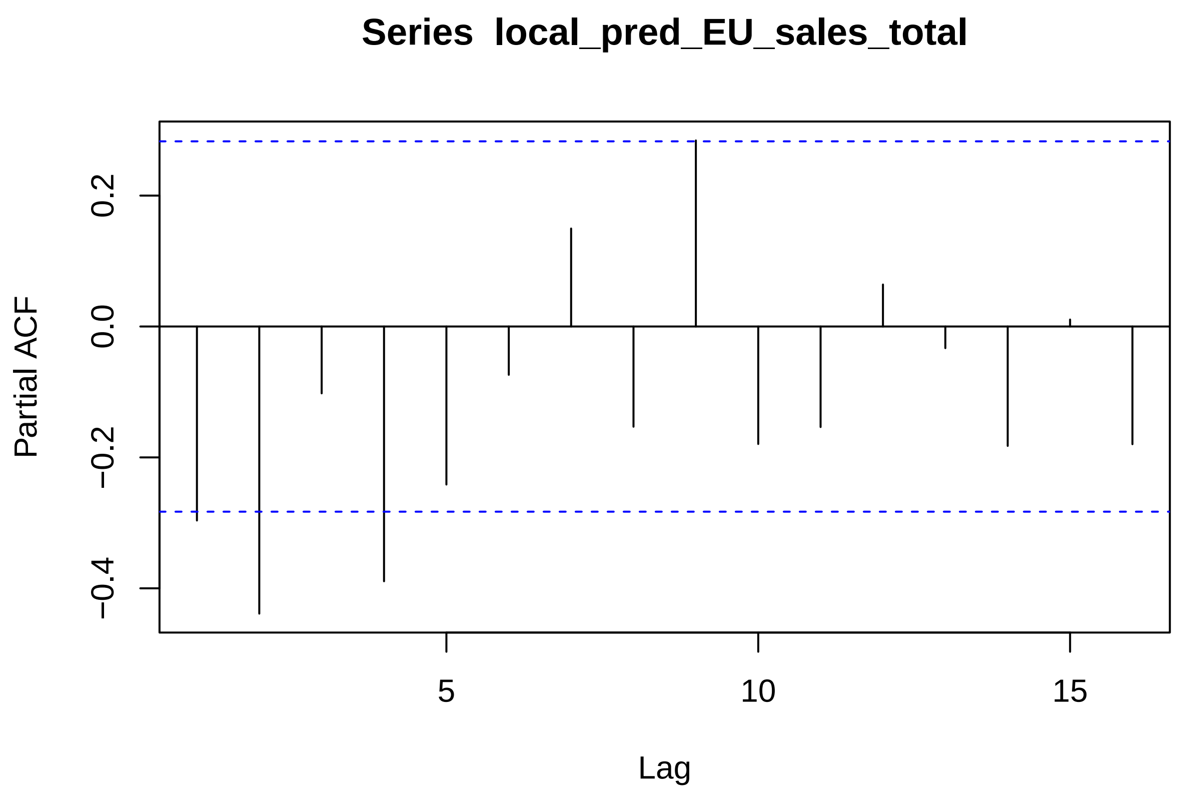
```
#Now, let's look at the locally predictable series
#We will model it as an ARMA series

local_pred_EU_sales_total <- EU_Con_sales_total-global_pred_EU_sales_total
plot(local_pred_EU_sales_total, col='red', type = "l")
```



```
acf(local_pred_EU_sales_total)
```



```
acf(local_pred_EU_sales_total, type="partial")
```



```
armafit <- auto.arima(local_pred_EU_sales_total)
armafit
```

```
## Series: local_pred_EU_sales_total
## ARIMA(0,0,0) with zero mean
##
## sigma^2 estimated as 76758046:  log likelihood=-503.86
## AIC=1009.71   AICc=1009.8   BIC=1011.59
```

```
## as it is arima(0,0,0)
## we can say that its not havinfg any autoregressive behaviour left in local part.

#As it is white noise and stationary ,so the local time series model has zero correlation.

resi <- local_pred_EU_sales_total-fitted(armafit)

adf.test(resi,alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  resi
## Dickey-Fuller = -6.1557, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

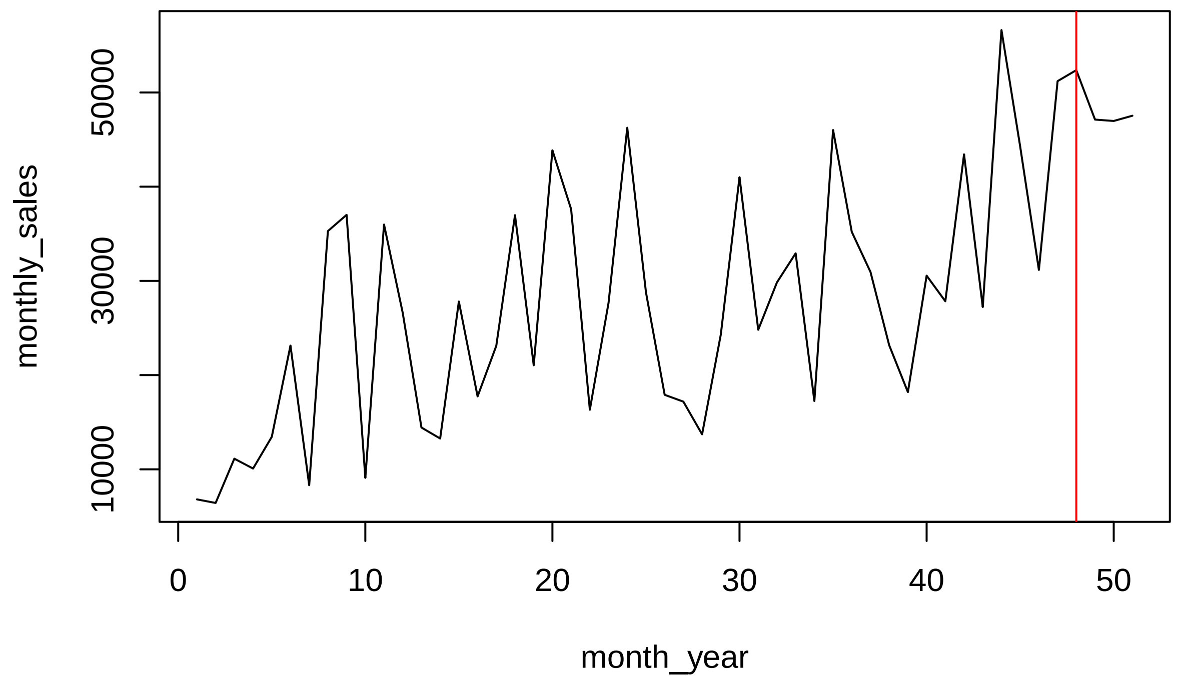
```
kpss.test(resi)
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  resi
## KPSS Level = 0.044692, Truncation lag parameter = 3, p-value = 0.1
```

```
# Hence the Forcasted 6 months values are
future_EU_sales
```

```
##        1        2        3
## 47121.43 46975.66 47532.69
```

```
future_EU_sales_df <- cbind(future_months,future_EU_sales)
EU_Consumer_past <- EU_Consumer[, c(7,4)]
colnames(future_EU_sales_df) <- c("month_year", "monthly_sales")
EU_Consumer_tot1 <- rbind(EU_Consumer_past,future_EU_sales_df)
plot(EU_Consumer_tot1, col = "black", type = "l")
abline(v=48, col = "red")
```



We can see that for the next 3 months, the sales of the store are going to go down.

5.2 Modelling on EU Consumer Quanitity Data

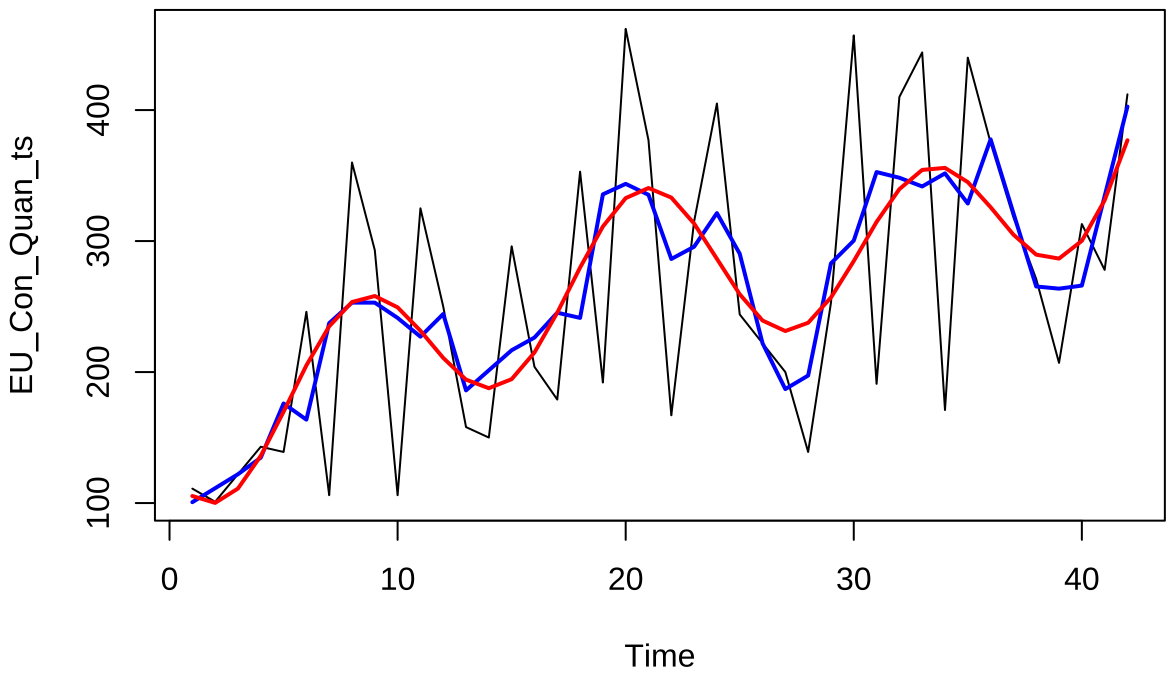
This process is going to be similar to the sales data but instead of sales we'll be determining the stock of quantity that the store should have for the next 3 months.

5.2.1 General Additive Model

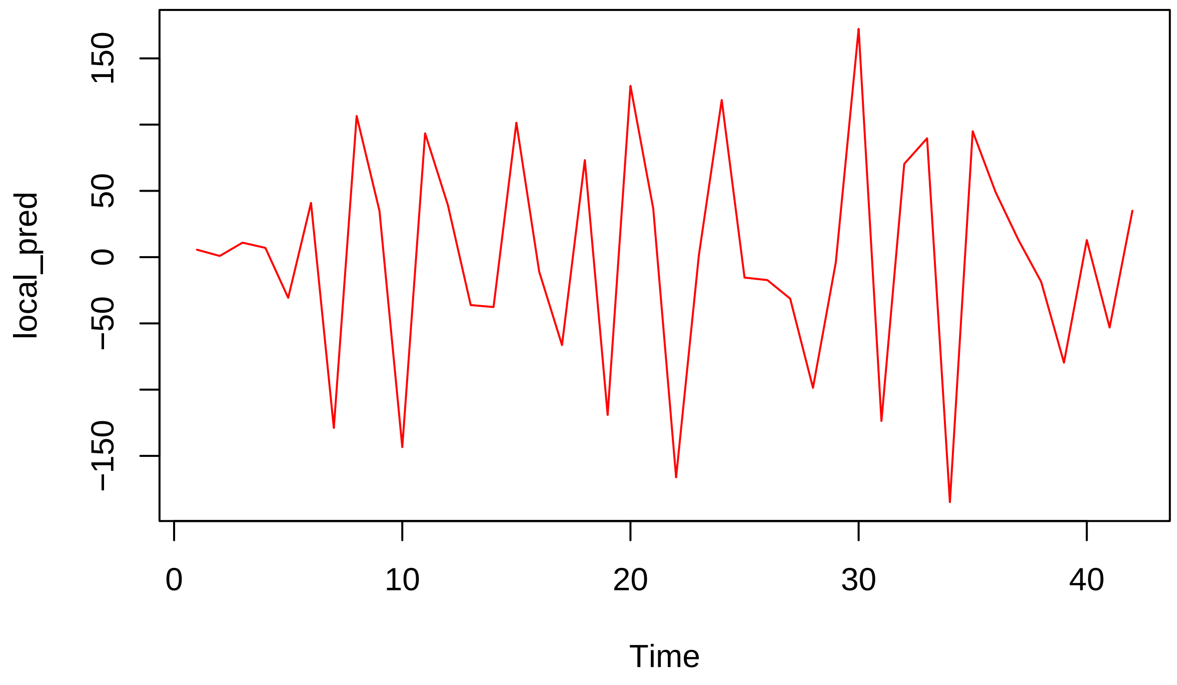
```
#Let's create the model using the first 42 rows.
#Then we can test the model on the remaining 6 rows later

EU_Con_Quan_ts_tot <- ts(EU_Consumer$monthly_qty)
# Dividing the train and test data
EU_Con_Quan_ts_indata <- EU_Consumer[1:42,]
EU_Con_Quan_ts_outdata <- EU_Consumer[43:48,]
#Plotting the TS of indata
EU_Con_Quan_ts <- ts(EU_Con_Quan_ts_indata$monthly_qty)
plot(EU_Con_Quan_ts)


#Smoothing the EU_Consumer_quantity

smoothed_EU_quantity <- smoothing_fun(EU_Con_Quan_ts)

#Plotting the smoothed series
EU_quantity_timevals_in <- EU_Con_Quan_ts_indata$month_year
lines(smoothed_EU_quantity, col="blue", lwd=2)

#Building a model on the smoothed time series using classical decomposition
#First, let's convert the time series to a data frame
#smoothed series
EU_quantity_smootheddf <- as.data.frame(cbind(EU_quantity_timevals_in,
                                              as.vector(smoothed_EU_quantity)))
colnames(EU_quantity_smootheddf) <- c('Month', 'Quantity')


#modelling the global part of the time series using the additive model for both trend and seasonality
lmfit <- lm(Quantity ~ sin(0.5*Month)+poly(Month,4)+cos(0.5*Month),
            data=EU_quantity_smootheddf)
global_pred <- predict(lmfit, Month=timevals_in)
summary(global_pred)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   100.1   219.0   258.7   260.3   314.3   377.0
```

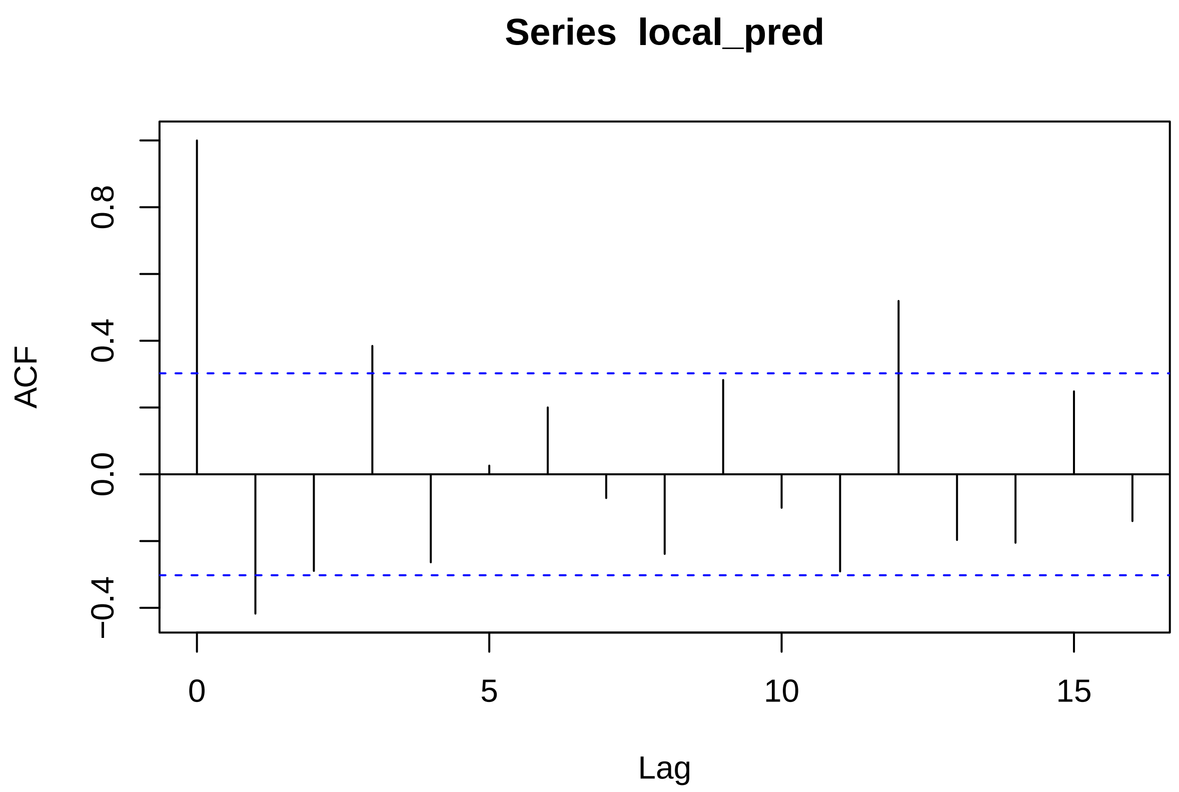
```
lines(EU_quantity_timevals_in, global_pred, col='red', lwd=2)
```



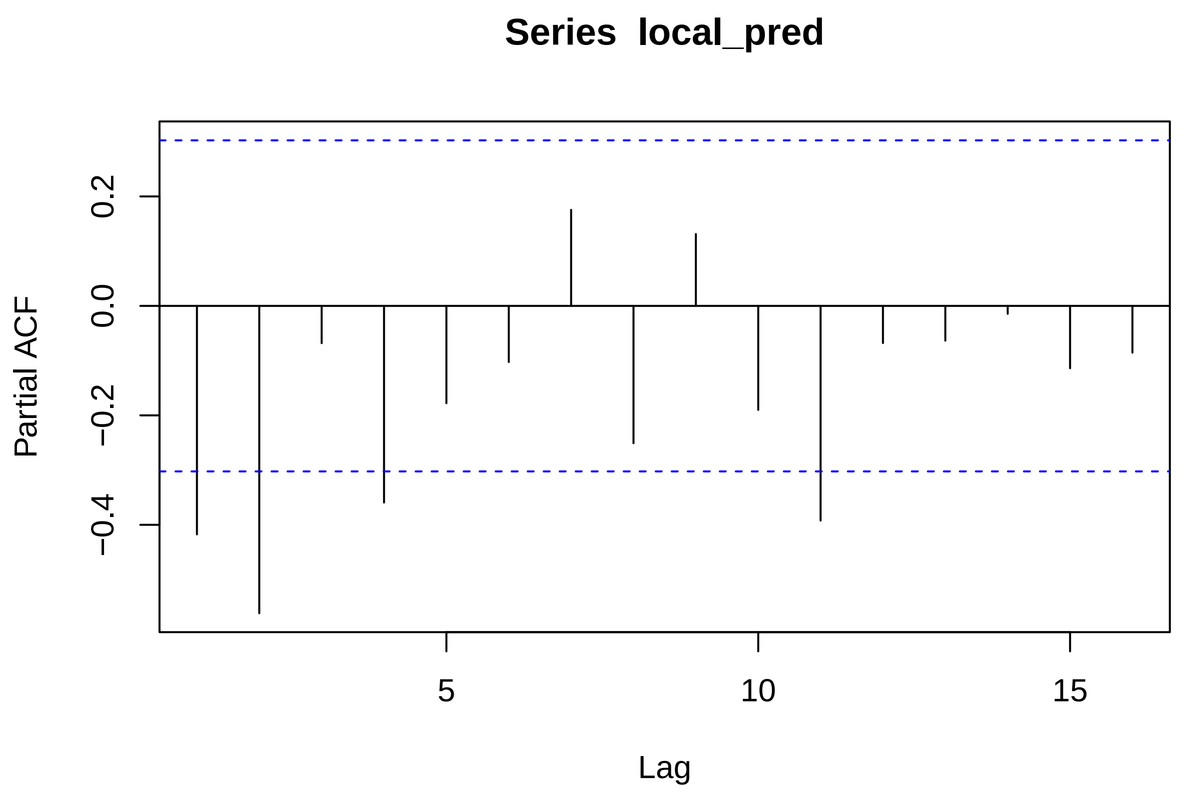
```
#Now, let's look at the locally predictable series
#We will model it as an ARMA series
local_pred <- EU_Con_Quan_ts-global_pred
plot(local_pred, col='red', type = "l")
```



```
# Acf plot of local pred
acf(local_pred)
```



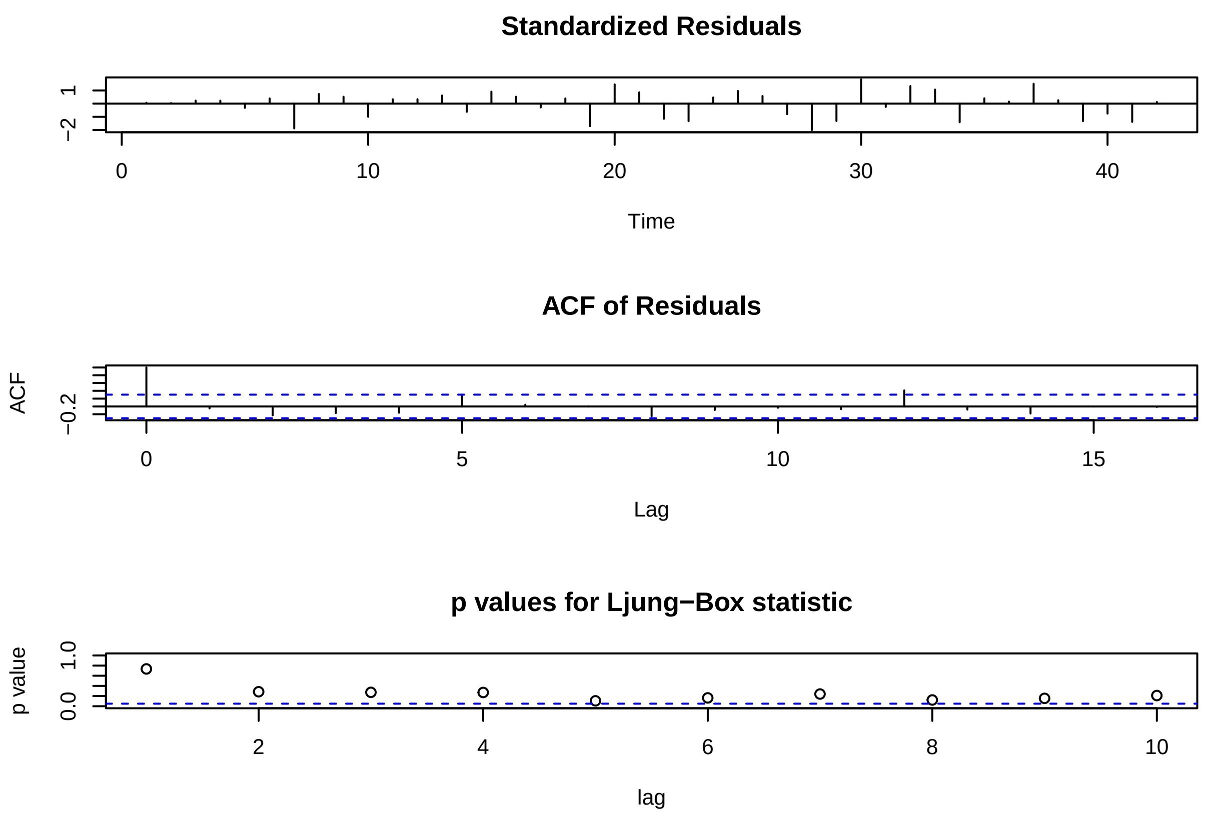
```
acf(local_pred, type="partial")
```



```
armafit <- auto.arima(local_pred)
armafit
```

```
## Series: local_pred
## ARIMA(2,0,0) with zero mean
##
## Coefficients:
##           ar1      ar2
##       -0.6375  -0.5403
## s.e.   0.1266   0.1234
##
## sigma^2 estimated as 4031:  log likelihood=-233.35
## AIC=472.7   AICc=473.33   BIC=477.91
```

```
tsdiag(armafit)
```



We can see that the left part is white noise ACF residual plot.

```
arma_fitted=armafit$fitted

global_pred_combined=global_pred+arma_fitted

#We'll check if the residual series is white noise
resi <- local_pred-fitted(armafit)

adf.test(resi,alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  resi
## Dickey-Fuller = -5.0079, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

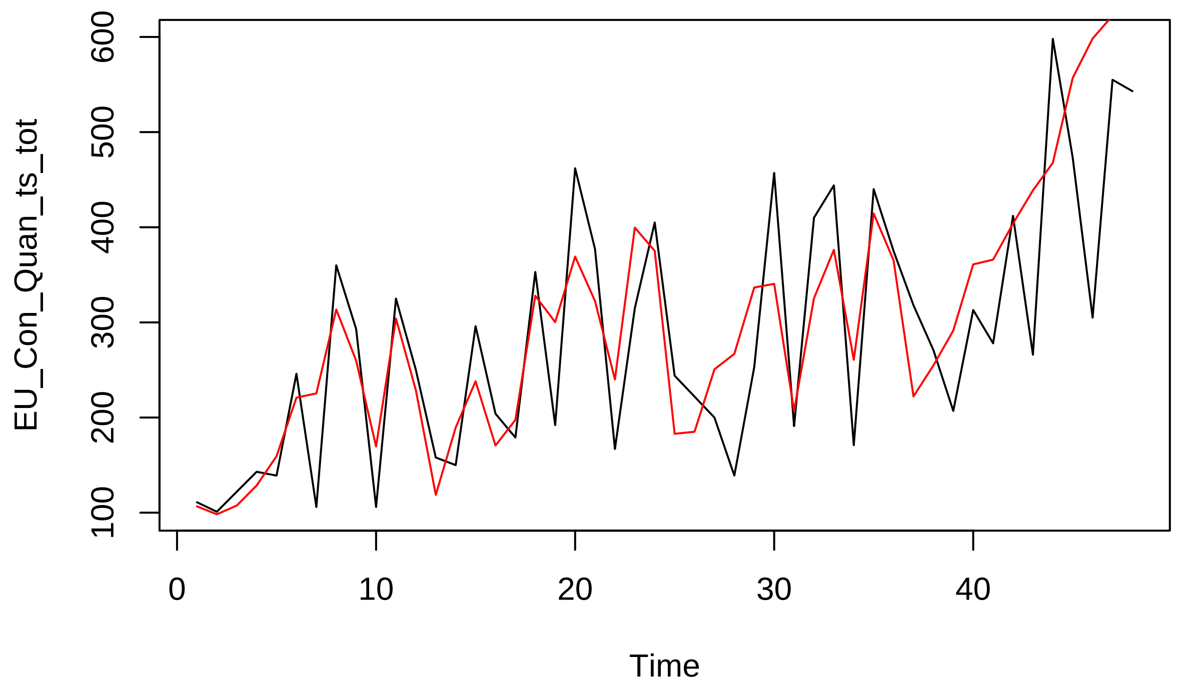
```
kpss.test(resi)
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  resi
## KPSS Level = 0.067656, Truncation lag parameter = 3, p-value = 0.1
```

```r
#So we have concluded that , the residual time series is staionary and WHITE NOISE .

#Now, let's evaluate the model using MAPE
#First, let's make a prediction for the last 6 months

timevals_out <- EU_Con_Quan_ts_outdata$month_year

global_pred_out <- predict(lmfit,data.frame(Month =timevals_out))

fcast <- global_pred_out

#Now, let's compare our prediction with the actual values, using MAPE
#predict values for auto arima and add local pred values

arma_pred=predict(armafit,n.ahead = 6)$pred
fcast_total=fcast+arma_pred


# Calculating the MAPE for classical decompostion
MAPE_class_dec <- accuracy(fcast_total,
                           EU_Con_Quan_ts_outdata$monthly_qty)[5]
MAPE_class_dec
```

```
## [1] 39.27774
```

```r
#Let's also plot the predictions along with original values, to
#get a visual feel of the fit

class_dec_pred <- c(ts(global_pred_combined),ts(fcast_total))
plot(EU_Con_Quan_ts_tot, col = "black")
lines(class_dec_pred, col = "red")
```



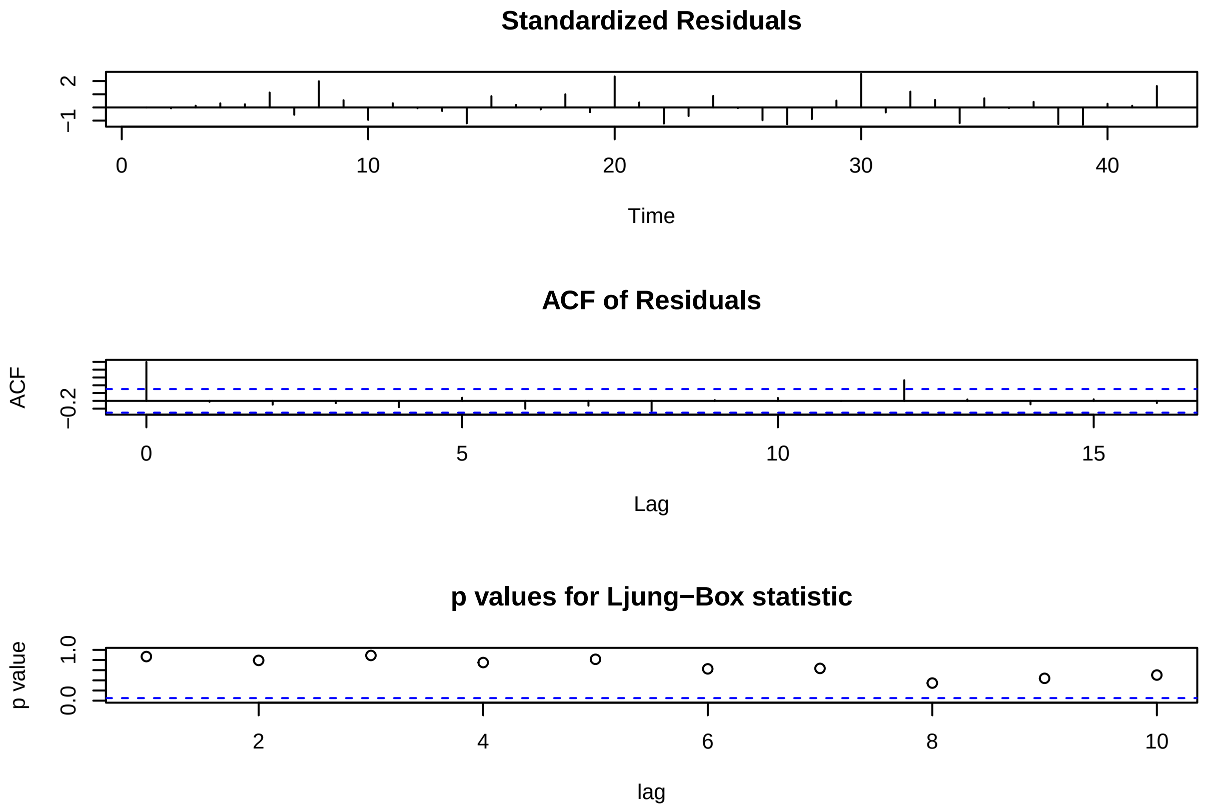
The GAM Model has done a good job in predicting the quantities. It has taken into considerations the peaks and dips of the time series. However, it is predicting a continuous upward trend after the 35th month indicating there would a constant rise in the stock of products. But that is not the case, as there have been some dips after the 35th Month.

5.2.2 ARIMA Model

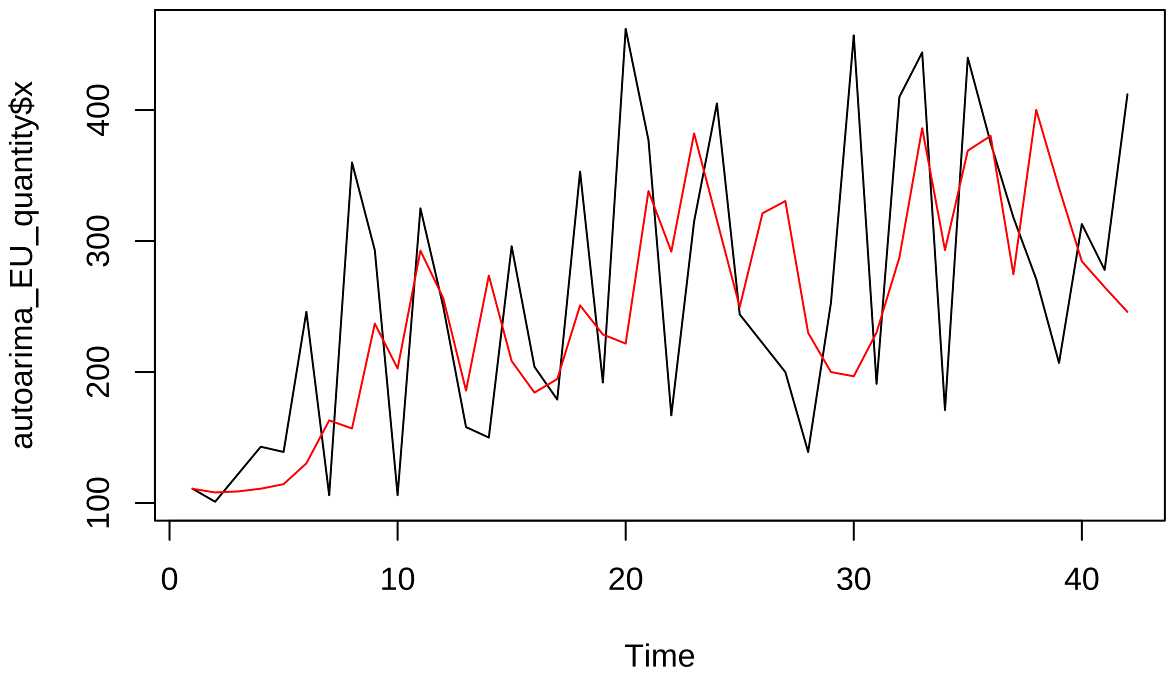
```
#now let's do an ARIMA fit

autoarima_EU_quantity<- auto.arima(EU_Con_Quan_ts)
autoarima_EU_quantity
```

```
## Series: EU_Con_Quan_ts
## ARIMA(2,1,0)
##
## Coefficients:
##           ar1      ar2
##       -0.7958  -0.5655
## s.e.   0.1264   0.1228
##
## sigma^2 estimated as 10445:  log likelihood=-247.39
## AIC=500.78   AICc=501.43   BIC=505.92
```

```
tsdiag(autoarima_EU_quantity)
```



```
plot(autoarima_EU_quantity$x, col="black")
lines(fitted(autoarima_EU_quantity), col="red")
```



```
#Again, let's check if the residual series is white noise

resi_auto_arima_EU_quantity <- EU_Con_Quan_ts - fitted(autoarima_EU_quantity)

adf.test(resi_auto_arima_EU_quantity,alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  resi_auto_arima_EU_quantity
## Dickey-Fuller = -3.7882, Lag order = 3, p-value = 0.03017
## alternative hypothesis: stationary
```

```
kpss.test(resi_auto_arima_EU_quantity)
```

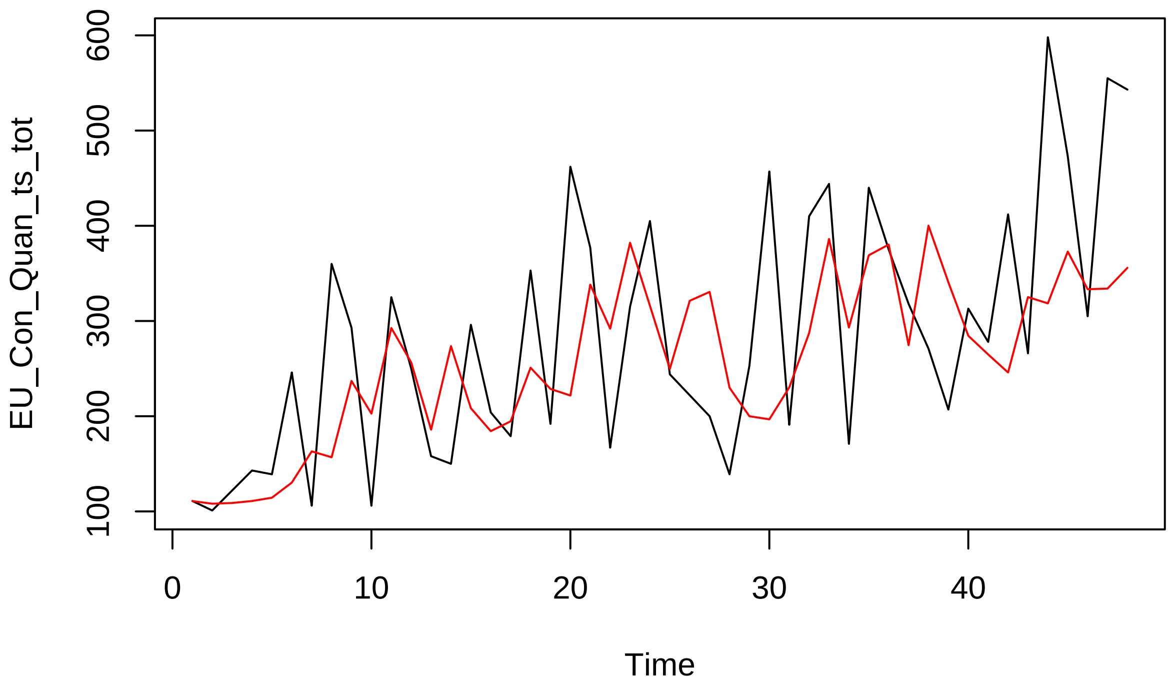
```
##
##  KPSS Test for Level Stationarity
##
## data:  resi_auto_arima_EU_quantity
## KPSS Level = 0.05576, Truncation lag parameter = 3, p-value = 0.1
```

```
#Also, let's evaluate the model using MAPE
fcast_auto_arima_EU_quantity <- predict(autoarima_EU_quantity, n.ahead = 6)
```

```r
MAPE_auto_arima_EU_quantity <- accuracy(fcast_auto_arima_EU_quantity$pred,
                                        EU_Con_Quan_ts_outdata$monthly_sales)[5]
MAPE_auto_arima_EU_quantity
```

```
## [1] 99.16651
```

```r
#Lastly, let's plot the predictions along with original values, to
#get a visual feel of the fit

auto_arima_pred_EU_quantity <- c(fitted(autoarima_EU_quantity),
                                 ts(fcast_auto_arima_EU_quantity$pred))
plot(EU_Con_Quan_ts_tot, col = "black")
lines(auto_arima_pred_EU_quantity, col = "red")
```



The ARIMA model is also not that bad as it is taking into consideration some of the peaks and dips. But comparing the GAM and Additive model, i think the GAM Model is doing a better job over here as well.

5.2.3 Prediction of quantity for the next 6 Months

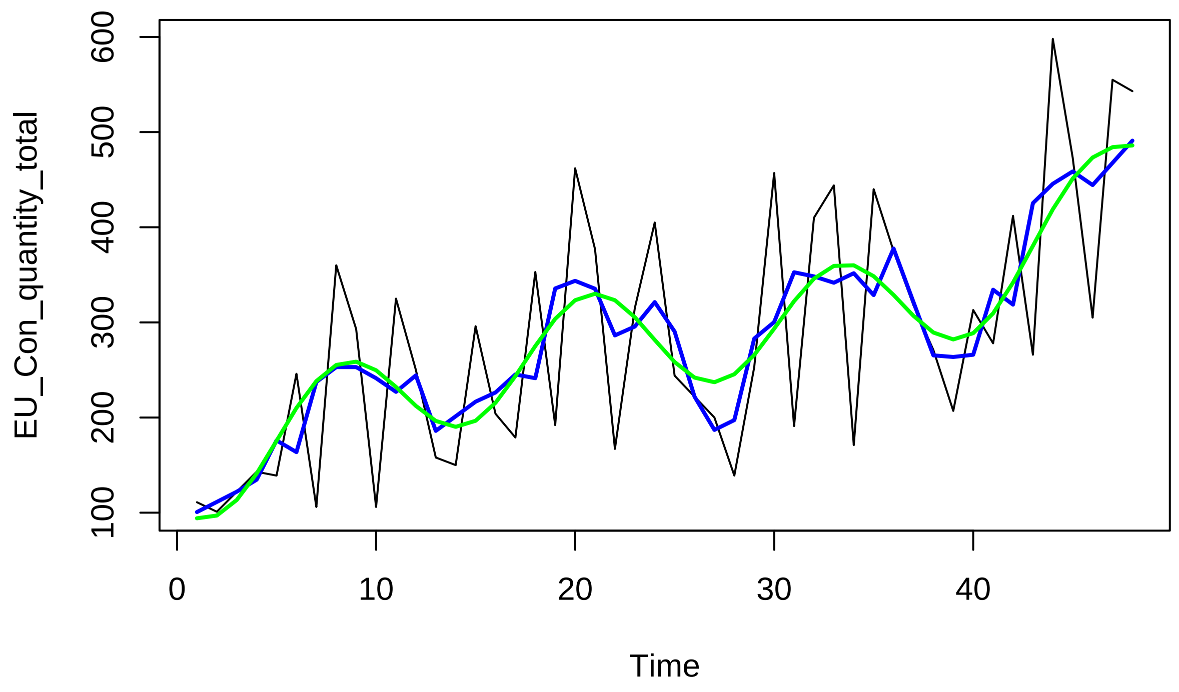
Now that we have determined which is the better model (GAM Model), we can now predict the estimated quantity for the next 3 months.

```r
# Forecasting the values for EU Consumer Quantity
EU_Con_quantity_total <- ts(EU_Consumer$monthly_qty)
```

```
plot(EU_Con_quantity_total)

#Smoothing the total EU Quantity time series
smoothed_EU_quantity_total <- smoothing_fun(EU_Con_quantity_total)

# Plotting the smoothed time series of EU_Consumer_sales
EU_Con_Quantity_timevals_total <- EU_Consumer$month_year
lines(smoothed_EU_quantity_total,col="blue",lwd=2)


EU_quantity_smootheddf_total <- as.data.frame(cbind(EU_Con_Quantity_timevals_total,
                                                    as.vector(smoothed_EU_quantity_total)))
colnames(EU_quantity_smootheddf_total) <- c('Month', 'Quantity')


#Now, let's fit a additive model with trend and seasonality to the data
#Seasonality will be modeled using a sinusoid function

lmfit_EU_quantity_total <- lm(Quantity ~ sin(0.5*Month) + cos(0.5*Month) + poly(Month,4),
                              data=EU_quantity_smootheddf_total)
global_pred_EU_quantity_total <- predict(lmfit_EU_quantity_total,
                                         Month=EU_Con_Quantity_timevals_total)
summary(global_pred_EU_quantity_total)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   94.13  235.88  281.86  282.96  329.16  485.99
```

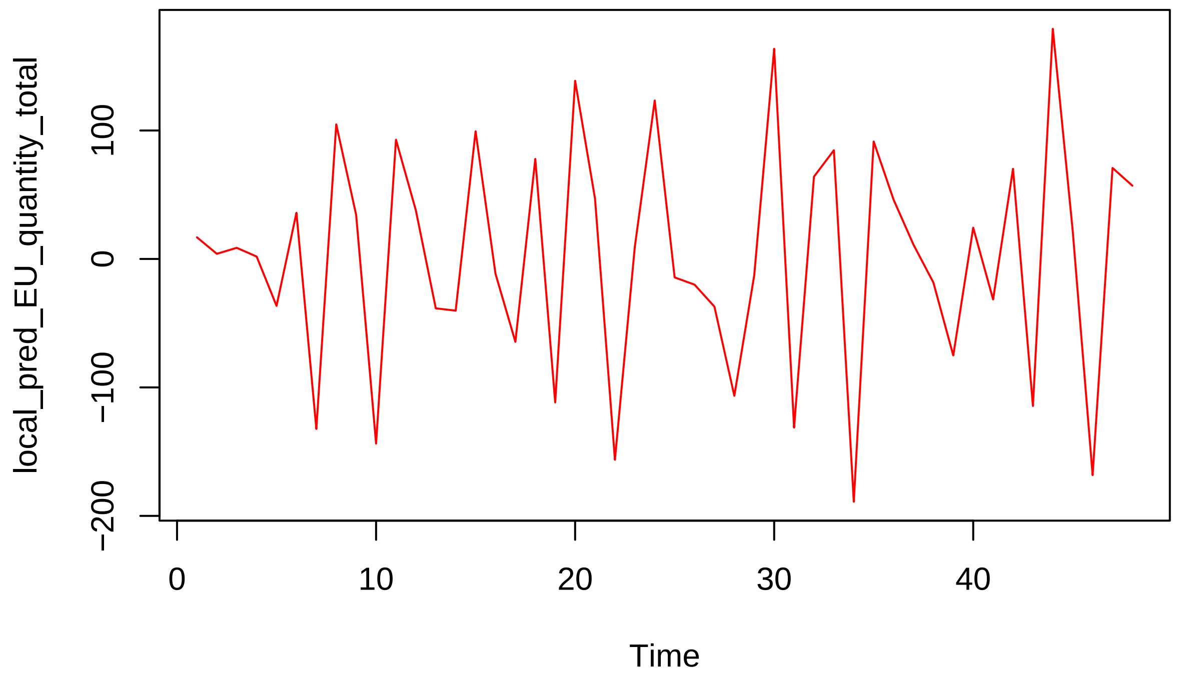
```
lines(EU_Con_Quantity_timevals_total, global_pred_EU_quantity_total, col='green',
      lwd=2)
```



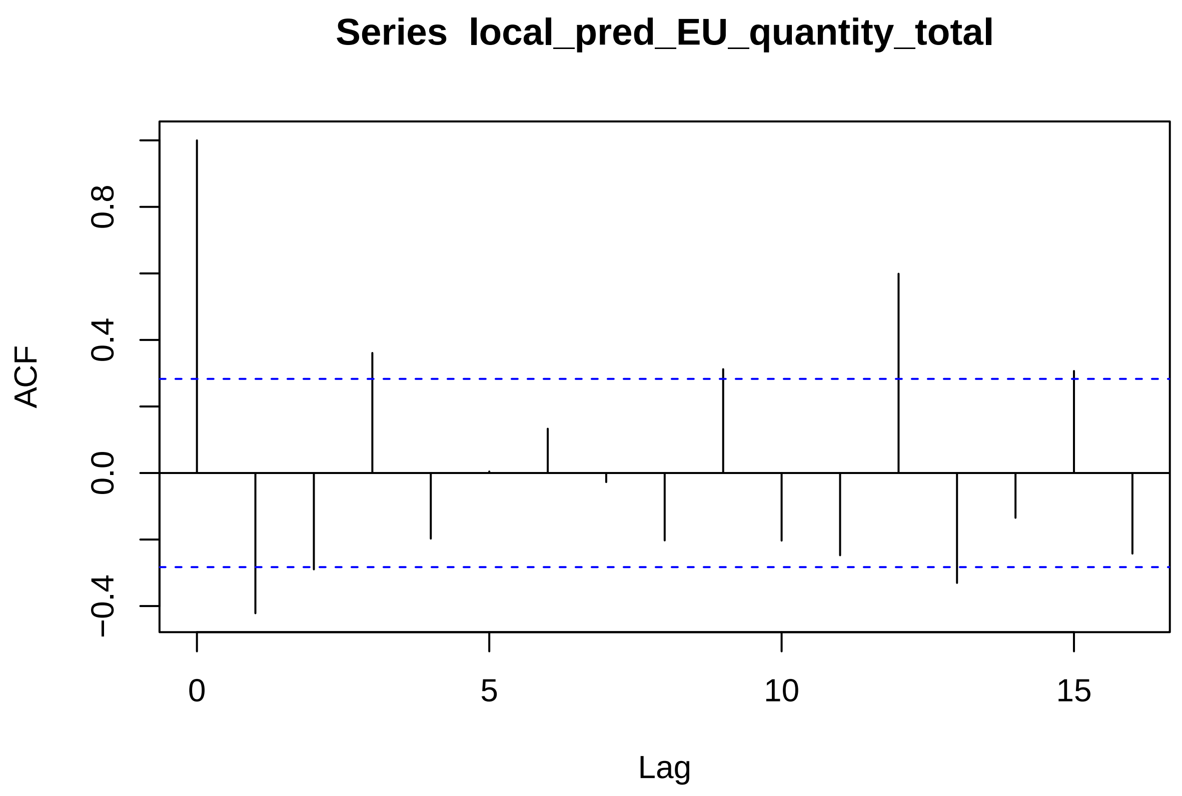
```
future_months <- as.data.frame(c(49:54))
colnames(future_months)<- c("Month")

future_EU_quantity <- predict(lmfit_EU_quantity_total,newdata=future_months)
future_EU_quantity
```

```
##        1        2        3        4        5        6
## 483.7658 484.0383 493.4105 516.9424 556.9636 612.5504
```

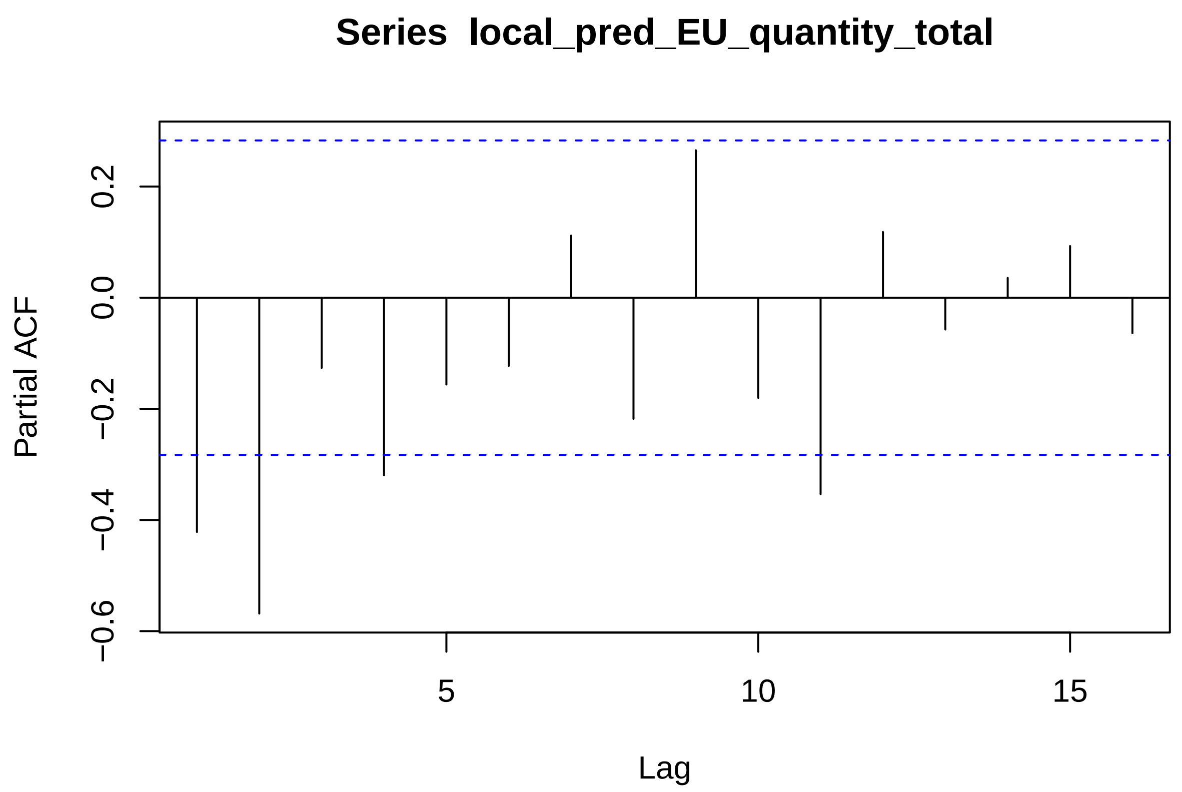
```
#Now, let's look at the locally predictable series
#We will model it as an ARMA series

local_pred_EU_quantity_total <- EU_Con_quantity_total-global_pred_EU_quantity_total
plot(local_pred_EU_quantity_total, col='red', type = "l")
```



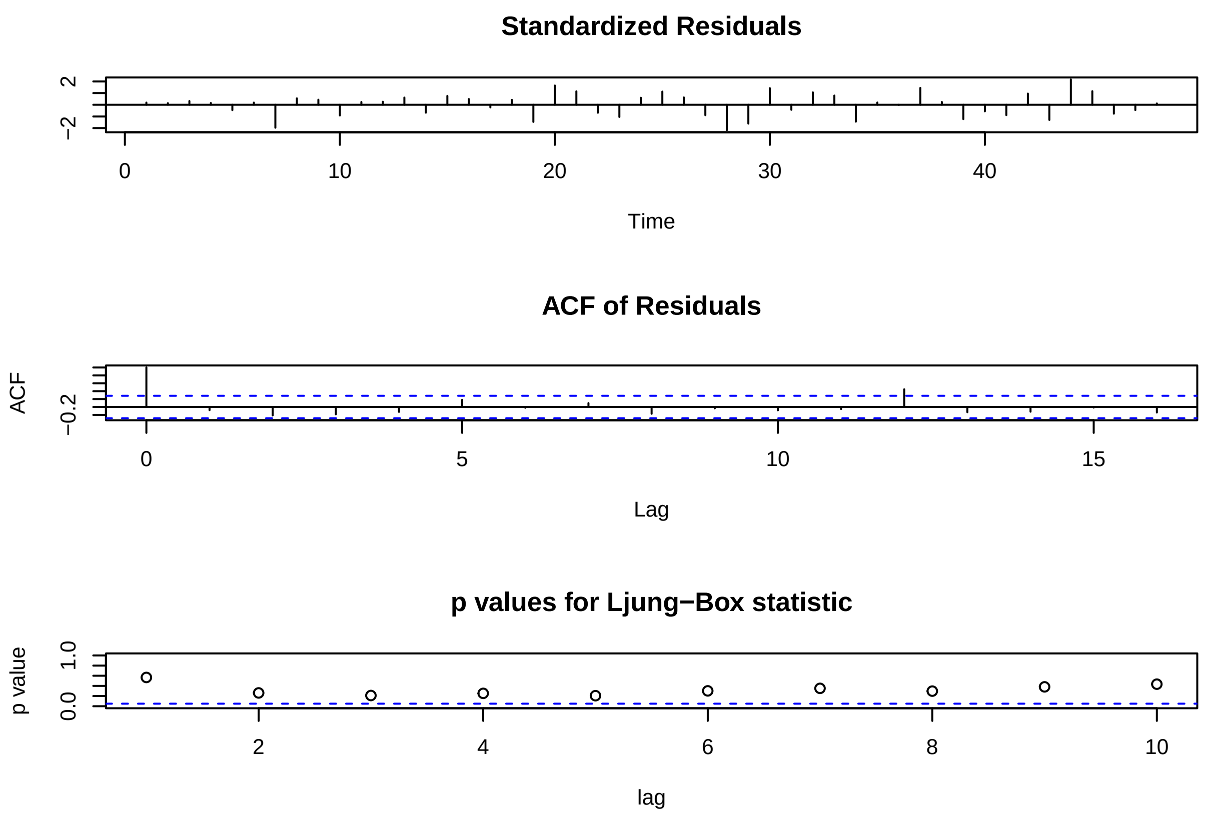
```
acf(local_pred_EU_quantity_total)
```



```
acf(local_pred_EU_quantity_total, type="partial")
```



```
armafit <- auto.arima(local_pred_EU_quantity_total)

tsdiag(armafit)
```



```
##  from acf of residuals after auto arima modelling we can say there is white noise
armafit
```

```
## Series: local_pred_EU_quantity_total
## ARIMA(2,0,0) with zero mean
##
## Coefficients:
##           ar1      ar2
##       -0.6739  -0.5771
## s.e.   0.1171   0.1144
##
## sigma^2 estimated as 4312:  log likelihood=-268.45
## AIC=542.91   AICc=543.45   BIC=548.52
```

```
## ar(2)
# it means there is some correlation among the residuals which can be used for better forecasting..

resi <- local_pred_EU_quantity_total-fitted(armafit)

adf.test(resi,alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  resi
## Dickey-Fuller = -5.295, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

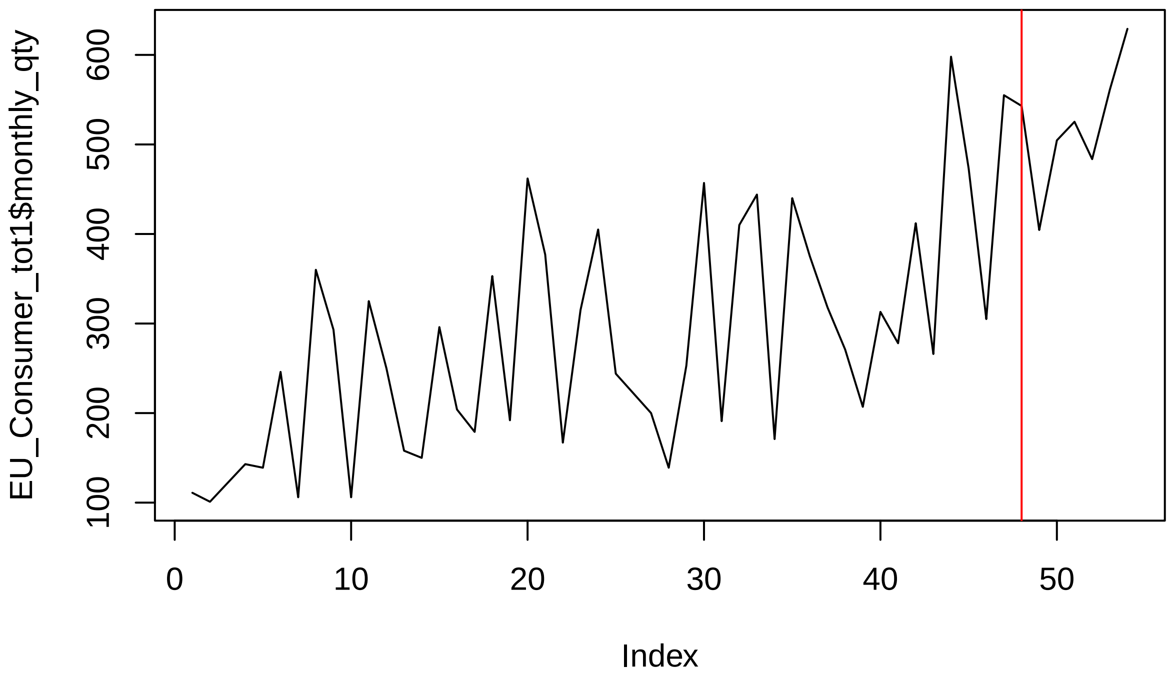
```
kpss.test(resi)
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  resi
## KPSS Level = 0.040318, Truncation lag parameter = 3, p-value = 0.1
```

```
## forecasting for local part from ar(2)
arma_pred = predict(armafit,n.ahead = 6)
# Hence the Forcasted 6 months values are combination of local and global pred
local_qnty_pred=arma_pred$pred


future_EU_quantity=future_EU_quantity+local_qnty_pred
future_EU_quantity
```

```
## Time Series:
## Start = 49
## End = 54
## Frequency = 1
##        1        2        3        4        5        6
## 404.4609 504.5848 525.3331 483.5705 561.0306 629.0697
```

```
future_EU_quantity_df <- cbind(future_months,as.data.frame(future_EU_quantity))
EU_Consumer_past <- EU_Consumer[, c(7,5)]
colnames(future_EU_quantity_df) <- c("month_year", "monthly_qty")
EU_Consumer_tot1 <- rbind(EU_Consumer_past,future_EU_quantity_df)


plot(EU_Consumer_tot1$monthly_qty, col = "black", type = "l")
abline(v=48, col = "red")
```



6. Conclusion

1. Based on the data provided, we identified the two most profitable segments worldwide which are APAC_Consumer and EU_Consumer.
2. We also identified that EU and US are the most profitable population per million wise.
3. Sales and Profits in both EU and US are high during the months of November and Christmas.
4. Appliances, Bookcases and Copiers are the most profitable during August and September in the European Union
5. Accessories, Phones and Copiers are the most profitable during September, October and December in the United States.
6. Tables are facing a loss across the world and the store should reduce the stock and sales of tables.
7. Upon Modelling, the General Additive Model gave the best accuracy as compared to ARIMA.
8. EU Consumer Sales may show the slow rise in coming months.
9. EU Consumer Quantity is likely to drop during initial 1 or 2 months & then rise rapidly in next 3 months, eventually reaching a plateau.

7. References

[1] https://github.com/vaibhavpalkar21/Time-Series-Retail-Giant-Sales-Forecasting