ANOMALY DETECTION

CHEROTICH FAITH

2022-06-12

Anomaly Detection

You have also been requested to check whether there are any anomalies in the given sales dataset. The objective of this task being fraud detection.

Anomaly Detection is used for different applications. It is a commonly used technique for fraud detection. It is also used in manufacturing to detect anomalous systems such as aircraft engines. It can also be used to identify anomalous medical devices and machines in a data center.

```
#we start by loading the required packages
#install.packages("anomalize")
library(anomalize)
```

```
## == Use anomalize to improve your Forecasts by 50%! =============================
## Business Science offers a 1-hour course - Lab #18: Time Series Anomaly Detection!
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```
library(factoextra)
```

```
## Loading required package: ggplot2

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(tibble)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tibbletime)
```

```
##
## Attaching package: 'tibbletime'

## The following object is masked from 'package:stats':
##
##     filter
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tibbletime::filter() masks dplyr::filter(), stats::filter()
## x dplyr::lag()         masks stats::lag()
```

```
#loading and reading the dataset

dir <- "C:/Users/user/Downloads/"
anomaly_data <- file.path(dir, "Supermarket_Sales_Forecasting - Sales.csv")
data <- read.csv(anomaly_data)
head(data)
```

```
##        Date    Sales
## 1  1/5/2019 548.9715
## 2  3/8/2019  80.2200
## 3  3/3/2019 340.5255
## 4 1/27/2019 489.0480
## 5  2/8/2019 634.3785
## 6 3/25/2019 627.6165
```

```
#let's preview the dimension of our dataset
dim(data)
```

```
## [1] 1000    2
```

Our dataset has 1000entries and 2 columns

```
#we also preview data types of our variables
sapply(data,class)
```

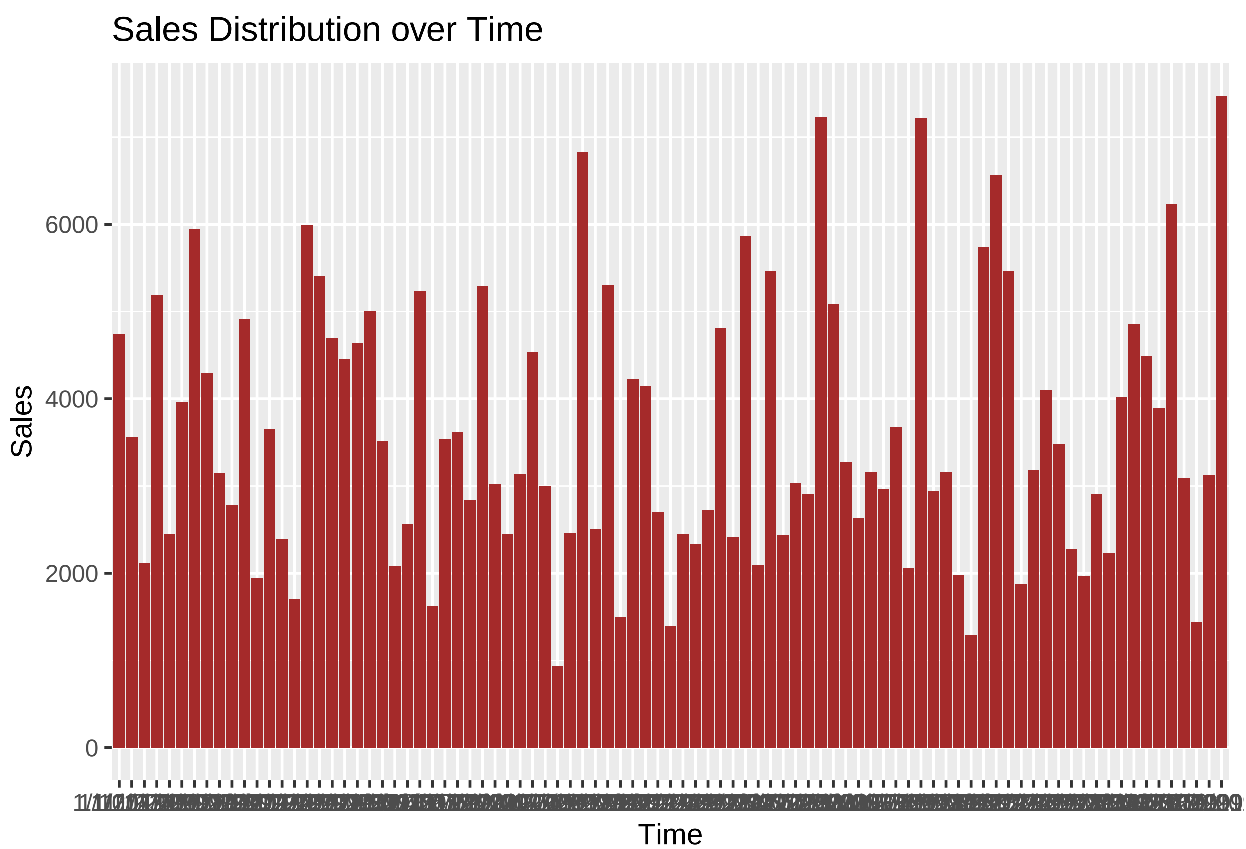
```
##        Date       Sales
## "character"   "numeric"
```

Our date is a factor datatype which we have to change to datetime datatype Our sale column has the right datatype

```r
# let's compute the total sales based on their common shared dates
total_sales<- aggregate(data$Sales, by = list(Date = data$Date), FUN = sum)

head(total_sales)
```

```
##        Date        x
## 1  1/1/2019 4745.181
## 2 1/10/2019 3560.949
## 3 1/11/2019 2114.963
## 4 1/12/2019 5184.764
## 5 1/13/2019 2451.204
## 6 1/14/2019 3966.617
```

```r
#we can then plot this distribution of sales over time using the ggplot2 package
# Sales distribution over time
library(ggplot2)
ggplot(data = data, aes(x = Date, y = Sales)) +
      geom_bar(stat = "identity", fill= "brown")+
      labs(title = "Sales Distribution over Time",
           x = "Time", y = "Sales")
```



We can see that the distribution of sales is widespread and not steady over different dates.

```r
#we can construct a frequency table for the sale distribution over different dates
```

```r
sale <- data.frame(table(data$Date))
head(sale)
```

```
##        Var1 Freq
## 1  1/1/2019   12
## 2 1/10/2019    9
## 3 1/11/2019    8
## 4 1/12/2019   11
## 5 1/13/2019   10
## 6 1/14/2019   13
```

```r
#we then combine the dataframes

new_data<- merge(total_sales,sale,by.x="Date",by.y="Var1")
head(new_data)
```

```
##        Date        x Freq
## 1  1/1/2019 4745.181   12
## 2 1/10/2019 3560.949    9
## 3 1/11/2019 2114.963    8
## 4 1/12/2019 5184.764   11
## 5 1/13/2019 2451.204   10
## 6 1/14/2019 3966.617   13
```

```r
#we rename the columns
names(new_data)<- c("Date","Sales","Frequency")
head(new_data)
```

```
##        Date    Sales Frequency
## 1  1/1/2019 4745.181        12
## 2 1/10/2019 3560.949         9
## 3 1/11/2019 2114.963         8
## 4 1/12/2019 5184.764        11
## 5 1/13/2019 2451.204        10
## 6 1/14/2019 3966.617        13
```

```r
#we can now change our date format to the right datetime datatype
new_data$Date<-as.Date(new_data$Date,"%m%d%Y")
str(new_data)
```

```
## 'data.frame':    89 obs. of  3 variables:
##  $ Date     : Date, format: NA NA ...
##  $ Sales    : num  4745 3561 2115 5185 2451 ...
##  $ Frequency: int  12 9 8 11 10 13 13 10 11 9 ...
```

```r
head(new_data)
```

```
##   Date    Sales Frequency
## 1 <NA> 4745.181        12
## 2 <NA> 3560.949         9
```

```
## 3 <NA> 2114.963         8
## 4 <NA> 5184.764        11
## 5 <NA> 2451.204        10
## 6 <NA> 3966.617        13
```

```r
new_data$Date<- as_tbl_time(new_data,index="Date")
str(new_data$Date)
```

```
## tbl_time [89 x 3] (S3: tbl_time/tbl_df/tbl/data.frame)
##  $ Date     : Date[1:89], format: NA NA ...
##  $ Sales    : num [1:89] 4745 3561 2115 5185 2451 ...
##  $ Frequency: int [1:89] 12 9 8 11 10 13 13 10 11 9 ...
##  - attr(*, "index_quo")= language ~"Date"
##   ..- attr(*, ".Environment")=<environment: R_EmptyEnv>
##  - attr(*, "index_time_zone")= chr "UTC"
```

```r
#let's preview the structure of our dataset
str(new_data)
```

```
## 'data.frame':    89 obs. of  3 variables:
##  $ Date     : tbl_time [89 x 3] (S3: tbl_time/tbl_df/tbl/data.frame)
##   ..$ Date     : Date, format: NA NA ...
##   ..$ Sales    : num  4745 3561 2115 5185 2451 ...
##   ..$ Frequency: int  12 9 8 11 10 13 13 10 11 9 ...
##   ..- attr(*, "index_quo")= language ~"Date"
##   .. ..- attr(*, ".Environment")=<environment: R_EmptyEnv>
##   ..- attr(*, "index_time_zone")= chr "UTC"
##  $ Sales    : num  4745 3561 2115 5185 2451 ...
##  $ Frequency: int  12 9 8 11 10 13 13 10 11 9 ...
```

```r
dim(new_data)
```

```
## [1] 89  3
```

Our dataset now has 89 entries and 3 columns

```r
#let's convert our dataframe to a tibble time
head(new_data)
```

```
## Warning in format.data.frame(if (omit) x[seq_len(n0), , drop = FALSE] else x, :
## corrupt data frame: columns will be truncated or padded with NAs
```

```
##                          Date    Sales Frequency
## 1        # A tibble: 6 x 3 4745.181        12
## 2   Date    Sales Frequency 3560.949         9
## 3   <date> <dbl>     <int> 2114.963         8
## 4 1 NA     4745.        12 5184.764        11
## 5 2 NA     3561.         9 2451.204        10
## 6 3 NA     2115.         8 3966.617        13
```

```r
new_data$Date <- as.Date(Sys.Date() + 1:nrow(new_data))
```

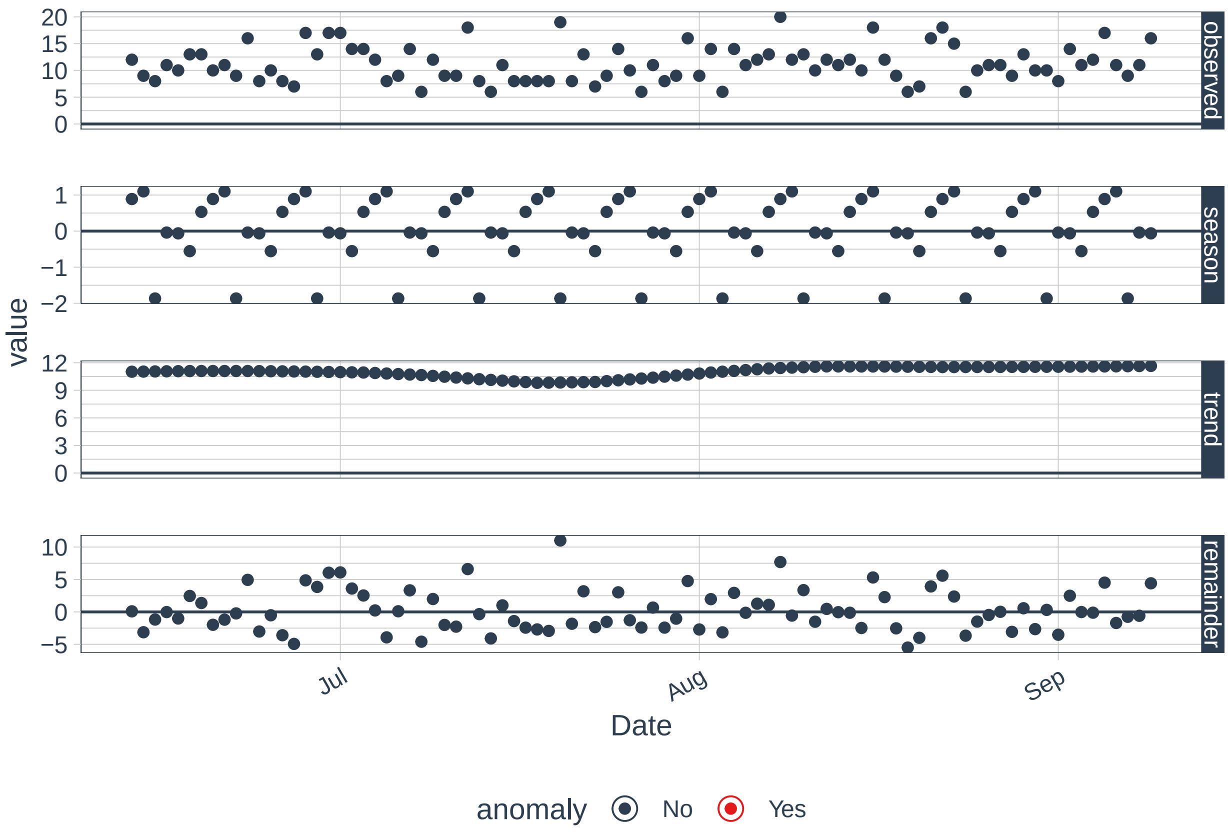
```r
#having loaded the required packages and their libraries we can now check for the anomaly

new_data%>%
  as_tibble()%>%
  time_decompose(Frequency)%>%
  anomalize(remainder)%>%
  time_recompose()%>%
  plot_anomaly_decomposition()
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = Date
```

```
## frequency = 7 days
```

```
## trend = 44.5 days
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```



From the output above, there is no anomaly detected. This means there was no fraud or unusual behaviour with sales at Carrefour Market in Kenya.