Anomaly Detection

2022-06-10

1. Defining the question

a) Specifying the question

form the marketing department on the most relevant marketing strategies that will result in the highest no. of sales (total price including tax).

b) Defining the metrics of success

explore a recent marketing dataset by performing various unsupervised learning techniques and later providing recommendations based on your insights. Check if there are any anomalies in the dataset

c) Understanding the context

You are a Data analyst at Carrefour Kenya and are currently undertaking a project that will inform the marketing department on the most relevant marketing strategies that will result in the highest no. of sales (total price including tax). Your project has been divided into four parts where you'll explore a recent marketing dataset by performing various unsupervised learning techniques and later providing recommendations based on your insights.

Part 4: 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.

d) Recording the experimental design

1. Problem Definition

2. Data Sourcing
3. Check the Data
4. Implement the Solution
5. Challenge the Solution
6. Follow up Questions

e) Data relevance

The data is relevant since it was provided by the company itself and can be used to answer the question

```
# Loading the dataset
# url <- http://bit.ly/CarreFourDataset
df <- read.csv('http://bit.ly/CarreFourSalesDataset')
head(df)
```

```
##        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
```

```
# Preview the last 6 items in the dataset
tail(df)
```

```
##           Date     Sales
## 995  2/18/2019   63.9975
## 996  1/29/2019   42.3675
## 997   3/2/2019 1022.4900
## 998   2/9/2019   33.4320
## 999  2/22/2019   69.1110
## 1000 2/18/2019  649.2990
```

```
# Checking the shape/ dimension of the dataframe
dim(df)
```

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

The dataset has 1000 rows and 2 columns

```
# Checking the summury of the dataframe
summary(df)
```

```
##      Date               Sales
##  Length:1000        Min.   :  10.68
##  Class :character   1st Qu.: 124.42
##  Mode  :character   Median : 253.85
##                     Mean   : 322.97
##                     3rd Qu.: 471.35
##                     Max.   :1042.65
```

```
# Checking the structure of the dataset
str(df)
```

```
## 'data.frame':    1000 obs. of  2 variables:
##  $ Date : chr  "1/5/2019" "3/8/2019" "3/3/2019" "1/27/2019" ...
##  $ Sales: num  549 80.2 340.5 489 634.4 ...
```

The date column is of type character while the sales column is of number

Data Cleaning

```
# Checking for uniformity in the column names of the dataset
colnames(df)
```

```
## [1] "Date"  "Sales"
```

```
# Checking for missing values
colSums(is.na(df))
```

```
##  Date Sales
##     0     0
```

There are no missing values in the dataset

```
# Checking for duplicates in the dataset
sum(duplicated(df))
```

```
## [1] 0
```

There are no duplicates in the data set

Univariate Analysis

```
# Analysis of the sales column
library(ggplot2)
library(dplyr)
```

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

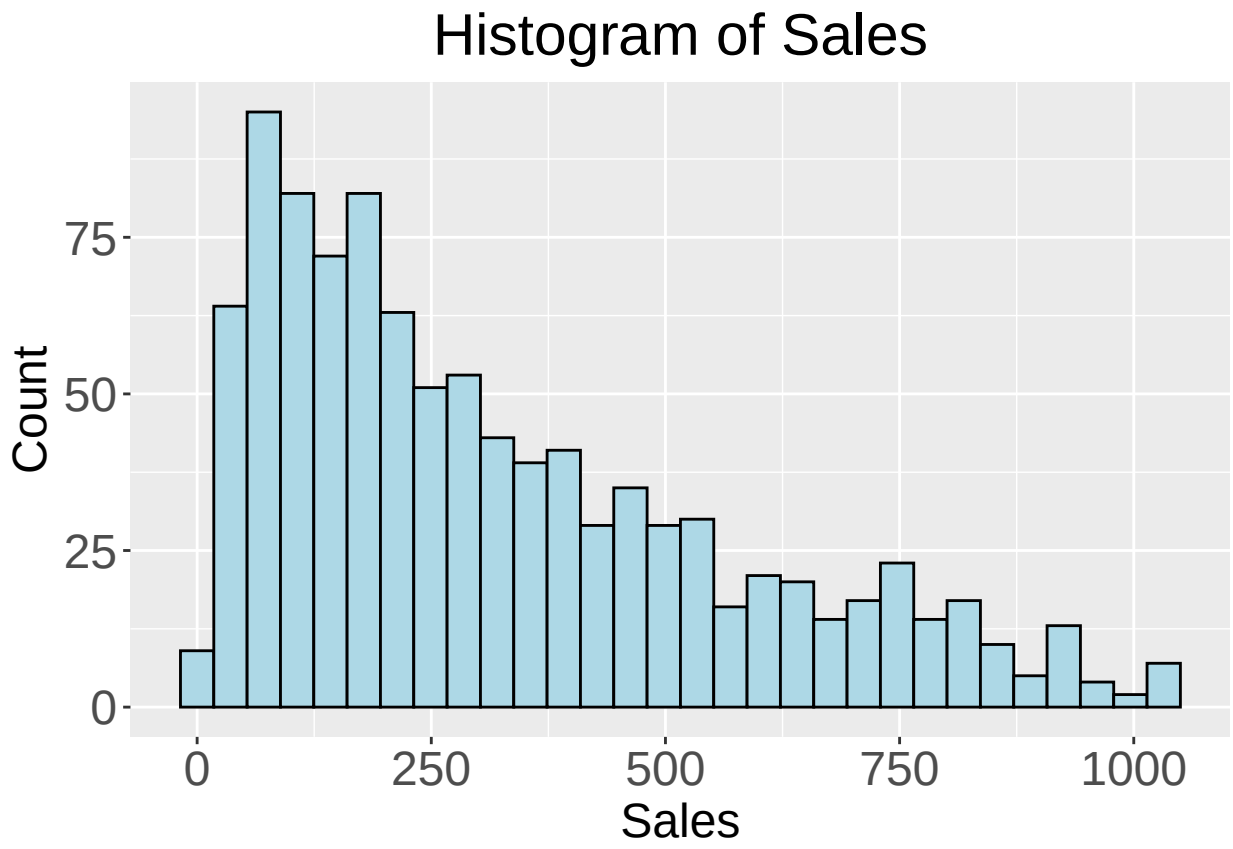
```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
options(repr.plot.width = 8, repr.plot.height = 8)
p = df %>% ggplot(aes(x = Sales))

p + geom_histogram(color="black", fill="lightblue" ) +
    labs(title = "Histogram of Sales", x = "Sales", y = "Count") +
    theme(axis.text = element_text(size=18),
          axis.title = element_text(size = 18),
          plot.title = element_text(hjust = 0.5, size = 22))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Most of the sales were less than 250

Anomaly Detection

```
# Loading the necessary libraries
library(tidyverse)
```

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

```
## v tibble  3.1.7     v purrr   0.3.4
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

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

```
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 </>
```

```
# Converting the date column to the right format
df$Date <- as.Date(df$Date,format = "%m/%d/%y")
head(df)
```

```
##         Date    Sales
## 1 2020-01-05 548.9715
```

```
## 2 2020-03-08  80.2200
## 3 2020-03-03 340.5255
## 4 2020-01-27 489.0480
## 5 2020-02-08 634.3785
## 6 2020-03-25 627.6165
```

```
# Ordering the dates
df1  = df[order(df$Date), ]
head(df1)
```

```
##           Date   Sales
## 18  2020-01-01 457.443
## 246 2020-01-01 399.756
## 451 2020-01-01 470.673
## 485 2020-01-01 388.290
## 497 2020-01-01 132.762
## 524 2020-01-01 132.027
```

```
str(df1)
```

```
## 'data.frame':    1000 obs. of  2 variables:
##  $ Date : Date, format: "2020-01-01" "2020-01-01" ...
##  $ Sales: num  457 400 471 388 133 ...
```

```
df1$Date <- as.POSIXct(df1$Date)
```

```
# Data Pre-processing using tibble
df2 <- as_tibble(df1)
head(df2)
```

```
## # A tibble: 6 x 2
##   Date                Sales
##   <dttm>              <dbl>
## 1 2020-01-01 03:00:00  457.
## 2 2020-01-01 03:00:00  400.
## 3 2020-01-01 03:00:00  471.
## 4 2020-01-01 03:00:00  388.
## 5 2020-01-01 03:00:00  133.
## 6 2020-01-01 03:00:00  132.
```

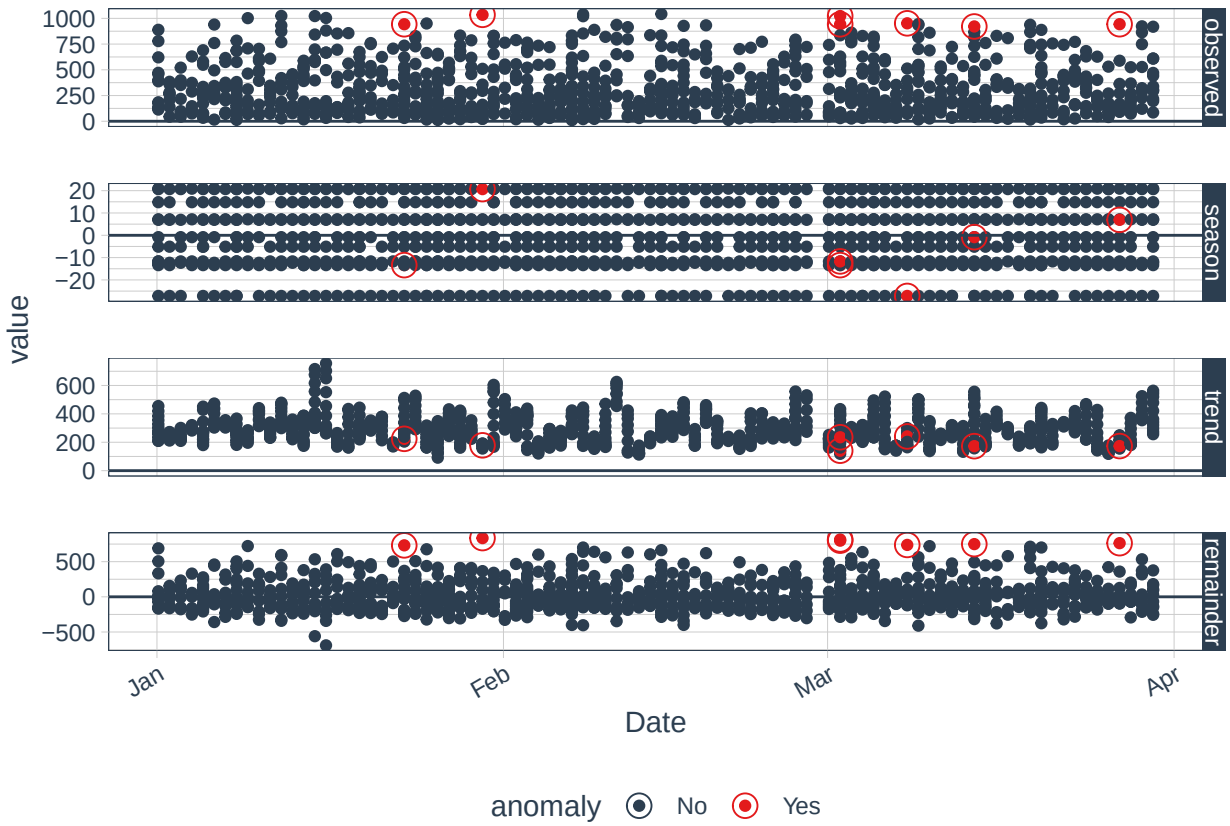
```
# Time series decomposition with anomalies
df2 %>% time_decompose(Sales, method = "stl", frequency = "auto", trend = "auto", message = TRUE)  %>%
  anomalize(remainder, method = "gesd", alpha = 0.05, max_anoms = 0.2) %>%
  plot_anomaly_decomposition()
```

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

```
## frequency = 11 seconds
```

```
## trend = 11 seconds
```

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

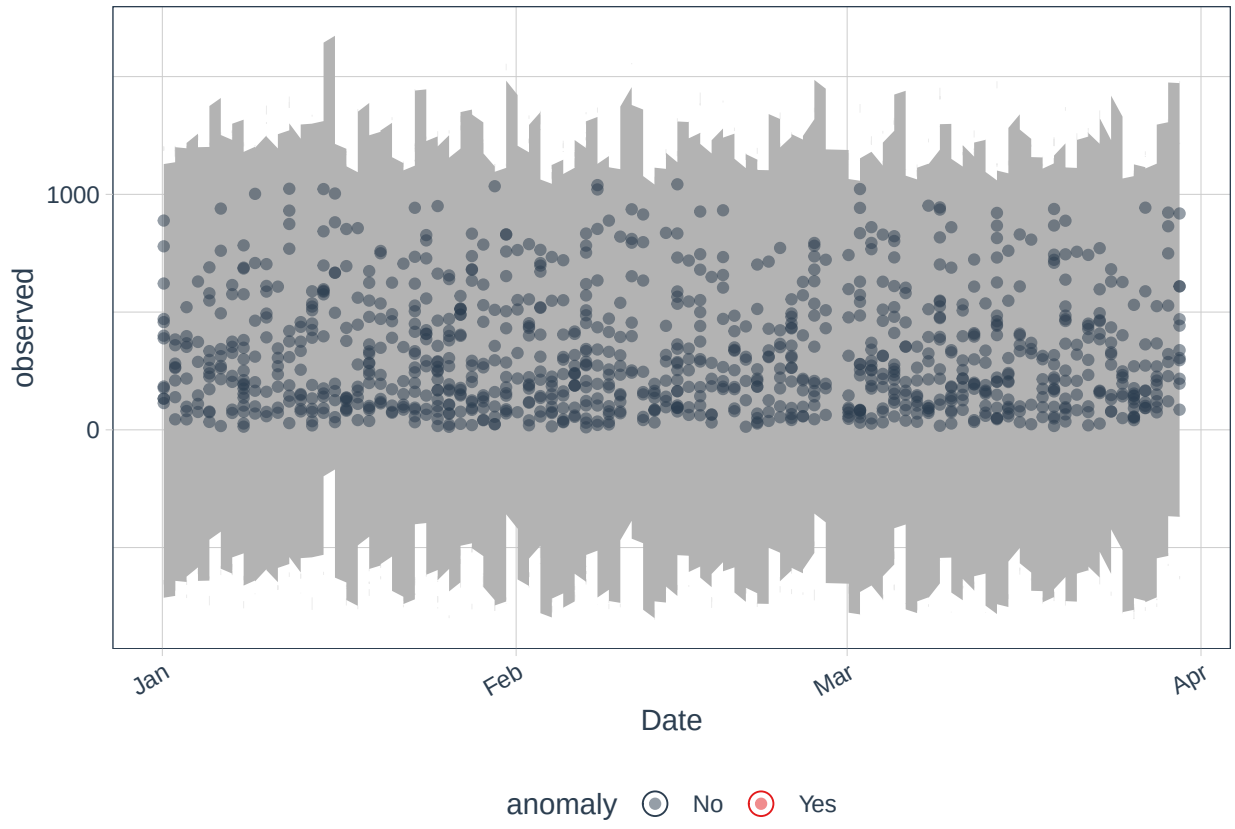



```
# Anomaly Detection
df2 %>%
  time_decompose(Sales) %>%
  anomalize(remainder) %>%
  time_recompose() %>%
  plot_anomalies(time_recomposed = TRUE, ncol = 3, alpha_dots = 0.5)
```

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

## frequency = 11 seconds

## trend = 11 seconds
```




```
# Extracting the points that are anomalies
df2 %>%
  time_decompose(Sales) %>%
  anomalize(remainder) %>%
  time_recompose() %>%
  filter(anomaly == 'Yes')
```

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

## frequency = 11 seconds

## trend = 11 seconds

## # A time tibble: 0 x 10
## # Index: Date
## # ... with 10 variables: Date <dttm>, observed <dbl>, season <dbl>,
## #   trend <dbl>, remainder <dbl>, remainder_l1 <dbl>, remainder_l2 <dbl>,
## #   anomaly <chr>, recomposed_l1 <dbl>, recomposed_l2 <dbl>
```