Anomaly Detection

Geoffrey Chege

2022-06-10

1. Introduction

Defining the question

- I am 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).
- I am expected to find out if there are any anomalies in the data.

Metric for success

- Be able to effectively detect anomalies in the products.

Understanding the context

- Carrefour operates different store formats, as well as multiple online offerings to meet the growing needs of its diversified customer base.
- In line with the brand's commitment to provide the widest range of quality products and value for money, Carrefour offers an unrivalled choice of more than 500,000 food and non-food products, and a locally inspired exemplary customer experience to create great moments for everyone every day.

Recording the experimental design

- Problem Definition
- Anomaly Detection
- Provide insights based on my analysis
- Provide recommendations

Data Relevance

- Link to the dataset: http://bit.ly/CarreFourSalesDataset

2. Loading libraries and dataset

```
# Load tidyverse and anomalize
library(tidyverse)
```

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

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## 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,warn.conflicts = FALSE)
```

```
## == 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(tibbletime)
```

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

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

- Data:

```
# read data
forecast <- read.csv("C:/Users/user/Downloads/Supermarket_Sales_Forecasting - Sales.csv")
head(forecast)
```

```
##        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
```

3. Anomaly Detection

Overview

- We are to check whether there are any anomalies in the given sales dataset. The objective of this task being fraud detection.

```r
# checking the structure of our data
str(forecast)
```

```
## '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 ...
```

```r
# checking the shape
dim(forecast)
```

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

- We have 1000 observations and 2 variables.

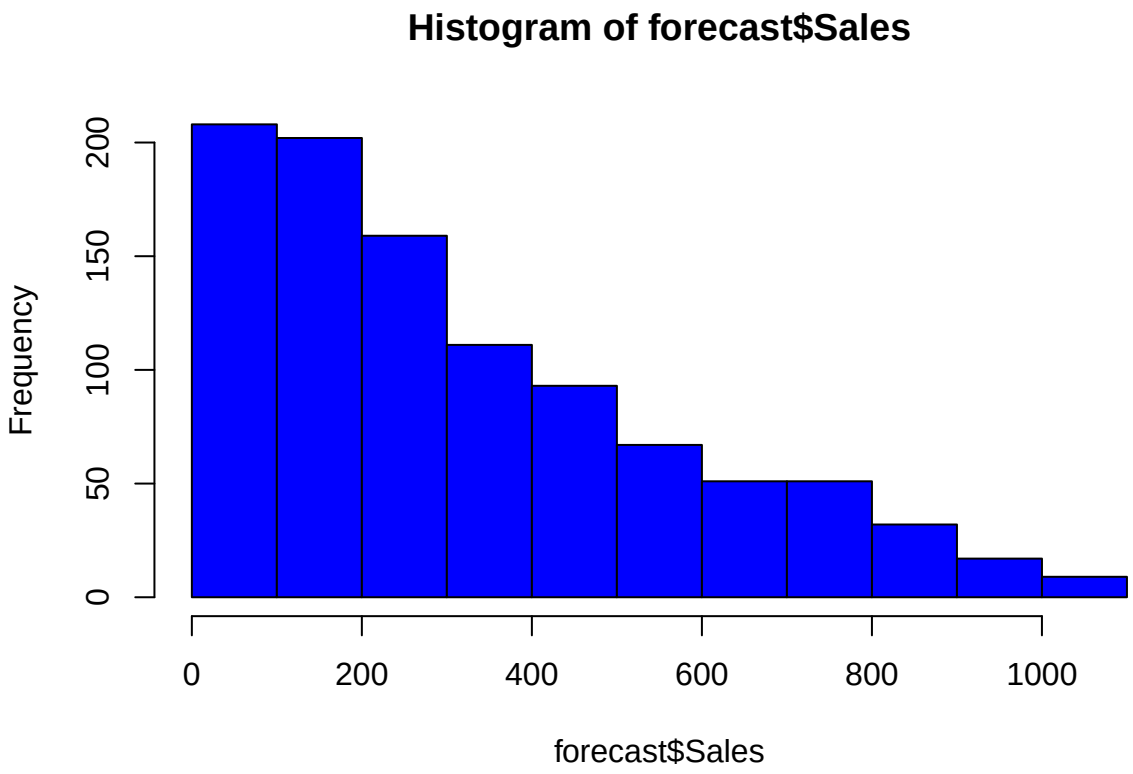
```r
# converting variables to our preferred format
forecast$Date <- as.Date(forecast$Date, "%m/%d/%Y")
```

```r
str(forecast)
```

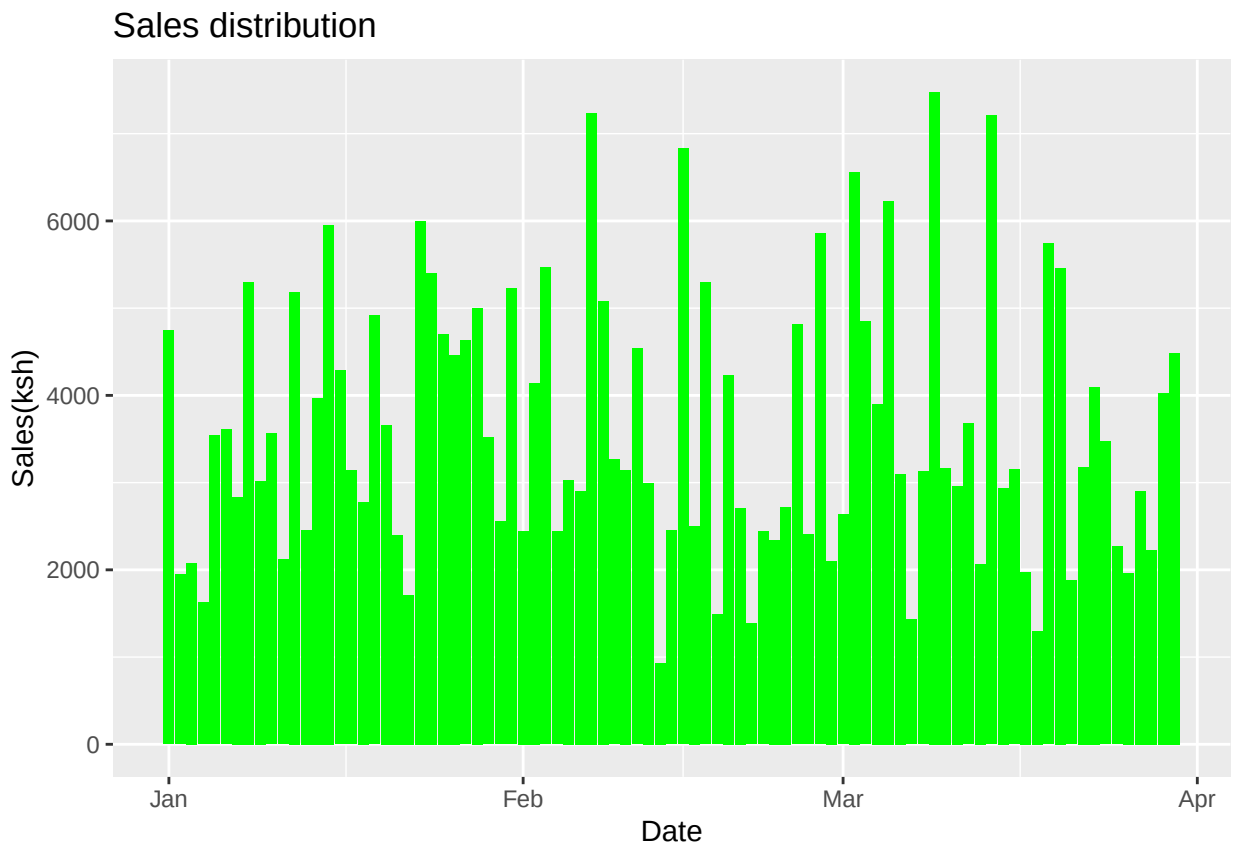
```
## 'data.frame':    1000 obs. of  2 variables:
##  $ Date : Date, format: "2019-01-05" "2019-03-08" ...
##  $ Sales: num  549 80.2 340.5 489 634.4 ...
```

Visualization

```r
# visualizing our sales
hist(forecast$Sales,col="blue")
```




```
# Sales distribution over time
library(ggplot2)
ggplot(data = forecast, aes(x = Date, y = Sales)) +
      geom_bar(stat = "identity", fill = "green") +
      labs(title = "Sales distribution",
           x = "Date", y = "Sales(ksh)")
```




```
# Ordering the data by Date
forecast = forecast %>% arrange(Date)
head(forecast)
```

```
##         Date   Sales
## 1 2019-01-01 457.443
## 2 2019-01-01 399.756
## 3 2019-01-01 470.673
## 4 2019-01-01 388.290
## 5 2019-01-01 132.762
## 6 2019-01-01 132.027
```

```
# Since our data has many records per day,
# We get the average per day, so that the data
forecast = aggregate(Sales ~ Date , forecast , mean)
head(forecast)
```

```
##         Date    Sales
## 1 2019-01-01 395.4318
## 2 2019-01-02 243.1879
## 3 2019-01-03 259.7661
## 4 2019-01-04 270.6148
## 5 2019-01-05 294.7236
## 6 2019-01-06 401.5783
```

```
# tbl_time have a time index that contains information about which column

# should be used for time-based subsetting and other time-based manipulation,

forecast= tbl_time(forecast, Date) # Converting data frame to a tibble time (tbl_time)
class(forecast)
```

```
## [1] "tbl_time"   "tbl_df"     "tbl"        "data.frame"
```

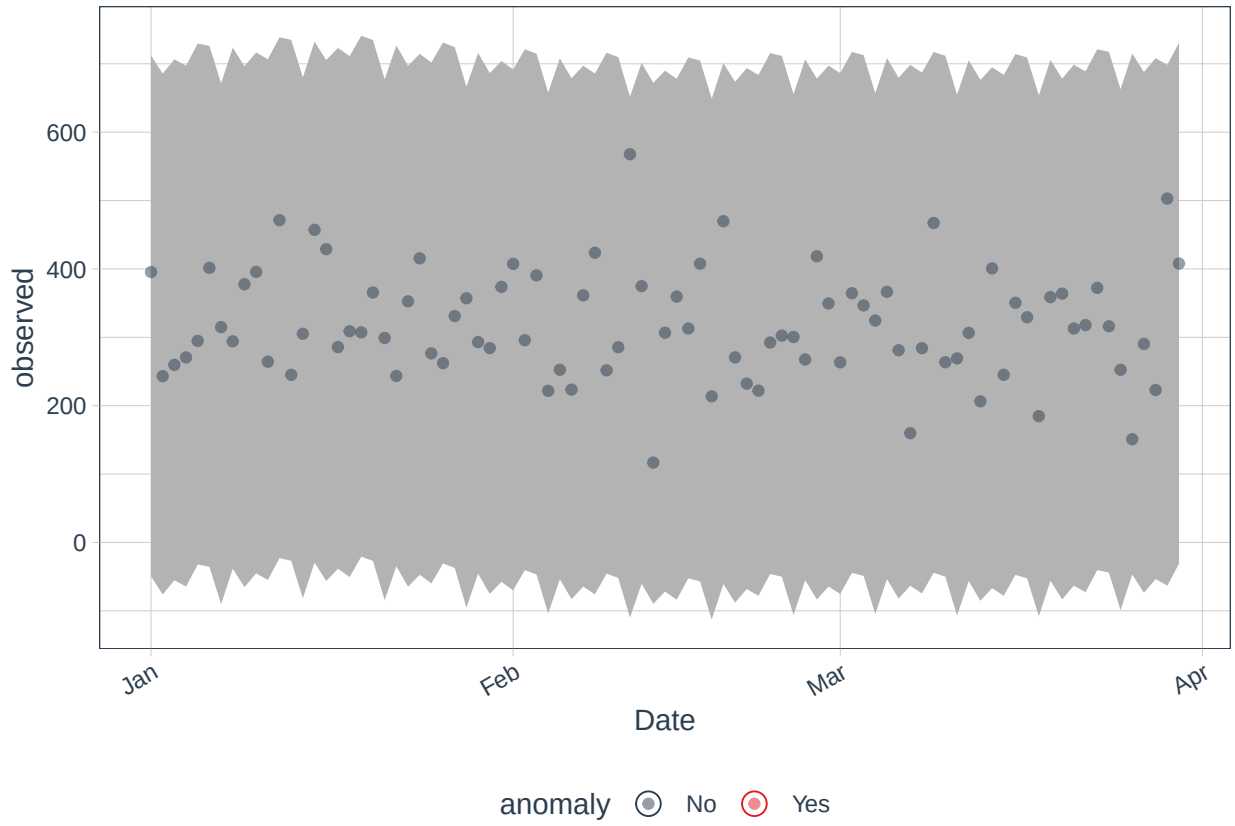
- I will use the following functions to detect and visualize anomalies:

```
forecast %>%
    time_decompose(Sales) %>%
    anomalize(remainder) %>%
    time_recompose() %>%
    plot_anomalies(time_recomposed = TRUE, ncol = 3, alpha_dots = 0.5)
```

```
## frequency = 7 days

## trend = 30 days

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



4. Conclusion

- There were no anomalies detected in the data.