R-Anomaly Detection

Vivian Bwana

2022-06-13

FRAUD (ANOMALY) DETECTION

Defining the Question

a) Specifying the question

To identify anomalies in the dataset

b) Metric for success

To be able to identify anomalies in the dataset

c) Understanding the Context

Carrefour is one of the leading retail shops, (supermarkets) in the world. It was founded in France, in 1959. It has over the years expanded it's operations internationally with the Kenyan branch opening in 1995.It has several branches in different parts of major cities countrywide.

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.

d) Experimental Design

1. Problem Definition
2. Data Sourcing
3. Check the Data
4. Perform Data Cleaning
5. Perform Anomaly Detection
6. Conclusion
7. Recommendation

e) Data Relevance /Sourcing

The dataset is relevant and reliable since it was provided by the client. We were able to draw relevant insights from it.

Data Understanding

Loading Libraries

```
# loading the necessary libraries
library(data.table)
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(purrr)
```

```
##
## Attaching package: 'purrr'

## The following object is masked from 'package:caret':
##
##     lift

## The following object is masked from 'package:data.table':
##
##     transpose
```

```
library(dbplyr)
library(dplyr)
```

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

## The following objects are masked from 'package:dbplyr':
##
##     ident, sql

## The following objects are masked from 'package:data.table':
##
##     between, first, last

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

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

```
library(tidyverse)
```

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

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

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::ident()     masks dbplyr::ident()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::lift()      masks caret::lift()
## x dplyr::sql()       masks dbplyr::sql()
## x purrr::transpose() masks data.table::transpose()
```

```
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(tibbletime)
```

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

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

```
library(stats)
#library(timetk)
```

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.

Dataset for Part 4: Anomaly Detection

```
# loading dataset
sale_forecast<- fread("~/Downloads/Supermarket_Sales_Forecasting - Sales.csv")
#preview
head(sale_forecast)
```

```
##          Date    Sales
##        <char>    <num>
## 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
```

Exploring the Dataset

Dimensions

```
# checking the dimensions of the datasets
# to see how many rows and colums there are
dim(sale_forecast)
```

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

There are 1000 and 2 columns in the dataset

Data Types

```
#Checking the datatypes of the dataset
str(sale_forecast)
```

```
## Classes 'data.table' and '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 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Descriptive Statistics Summary

```
# checking summary of the dataframe
library(Hmisc)
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```r
#library(describe)
summary(sale_forecast)
```

```
##      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
```

The function summary gives the statistical summary of mean, median, minimum, maximum and quantile ranges as shown above

Column Names

```r
# checking the column names
colnames(sale_forecast)
```

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

Missing Values

```r
#Checking for the sum of Missing values
colSums(is.na(sale_forecast))
```

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

There are no missing values

Duplicates

```r
# checking for duplicates
sale_forecast.duplicates <- sale_forecast[duplicated(sale_forecast),]

#printing duplicated rows
sale_forecast.duplicates
```

```
## Empty data.table (0 rows and 2 cols): Date,Sales
```

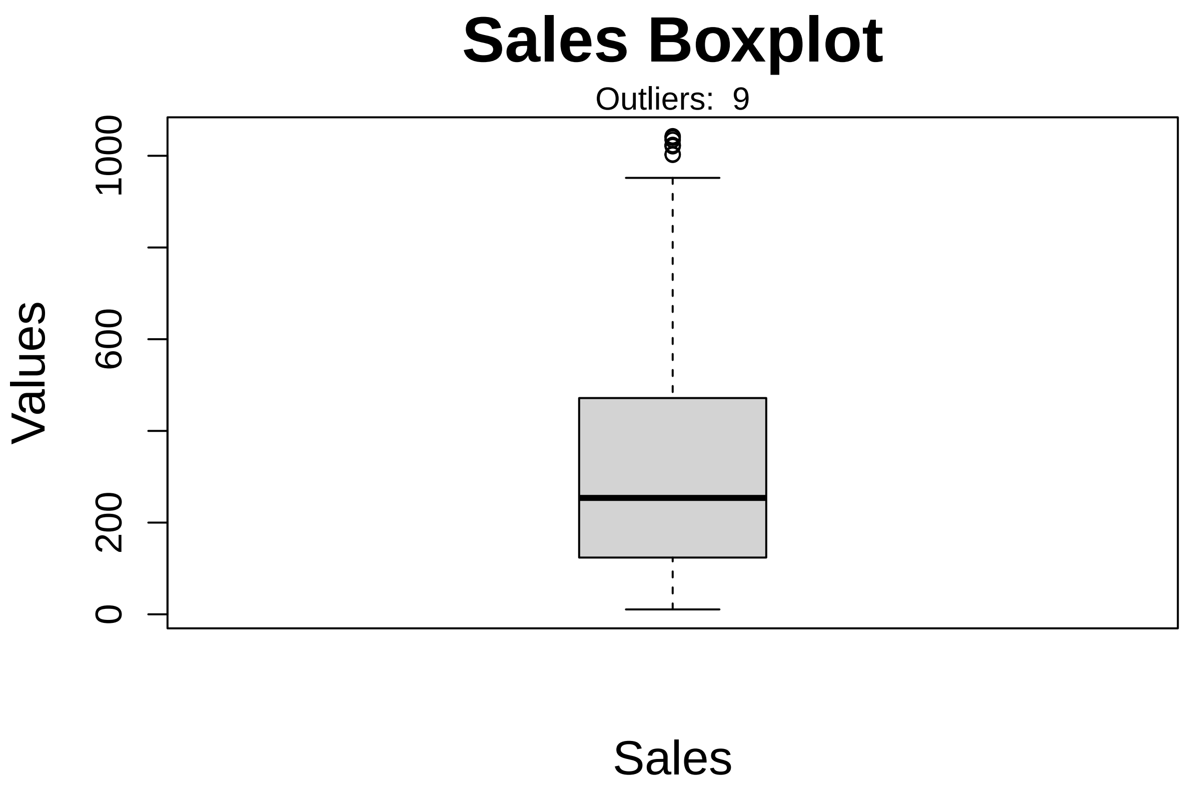
There are no duplicates

Outliers

```r
# Boxplot to check for outliers in 'Sales' column
options(repr.plot.width = 10, repr.plot.height = 7)

boxplot(sale_forecast$Sales, main="Sales Boxplot", xlab = "Sales",
        ylab = "Values", boxwex=0.4, cex.main=2, cex.lab=1.5, cex.axis=1.2)

# Display the number of outliers values in the column
outliers <- boxplot.stats(sale_forecast$Sales)$out
mtext(paste("Outliers: ", paste(length(outliers), collapse=", ")), cex=1)
```



There are outliers in sales. These will not be removed as they will be crucial for this analysis.

Converting Date to date object

```r
# Convert 'Date' to date object
sale_forecast$Date <- as.Date(sale_forecast$Date, format = "%m/%d/%Y")
```

Extracting the day and month from date and converting day to numeric

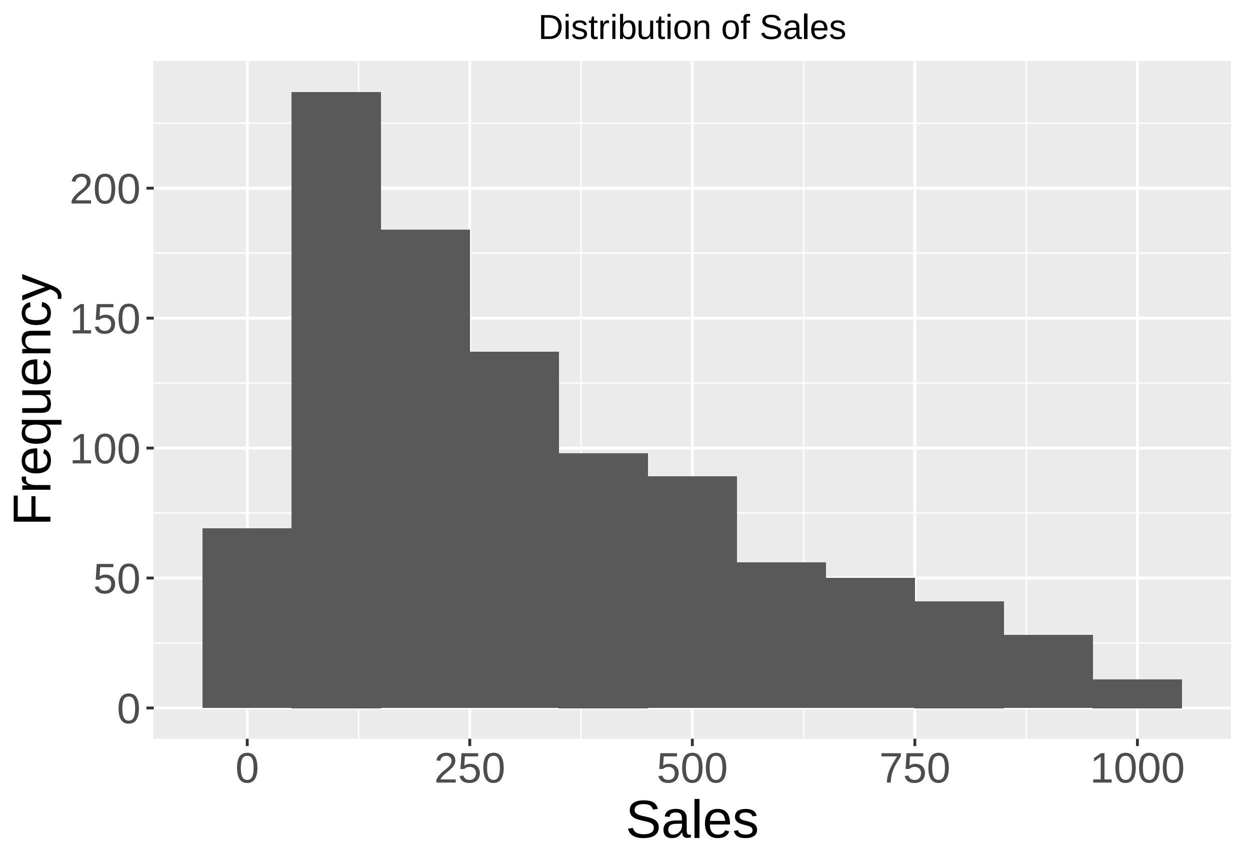
```r
sale_forecast$day <- format(sale_forecast$Date, "%d")
sale_forecast$month <- format(sale_forecast$Date, "%m")
sale_forecast$day <- as.numeric(sale_forecast$day)
```

Plotting a histogram to show distribution of sales

```r
# plotting a histogram to show distribution
# of sales over the given period

options(repr.plot.width = 8, repr.plot.height = 6)
p = sale_forecast %>% ggplot(aes(x = Sales ))

p + geom_histogram(binwidth = 100) +
    labs(title = "Distribution of Sales", x = "Sales", y = "Frequency") +
    theme(axis.title = element_text(size = 20),
          axis.text = element_text(size=16),
          plot.title = element_text(hjust = 0.5))
```



Converting the dataset into a tibble Loading libraries

```r
library(tsibble)
```

```
##
## Attaching package: 'tsibble'

## The following object is masked from 'package:data.table':
##
##     key

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

```r
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:tsibble':
##
##     interval

## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

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

```r
library(base)
library(arules)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

Converting to a tibble

```r
# converting to a tibble for easier manipulation

# Convert 'Date' to date object
sale_forecast$Date <- as.Date(sale_forecast$Date, format = "%m/%d/%Y")

# Sorting the date column
sale_forecast$Date <- sort(sale_forecast$Date )
```

```r
#converting to a tibble
sale_forecast <- as_tbl_time(sale_forecast, index = Date)

# preview
head(sale_forecast)
```

```
## # A time tibble: 6 x 4
## # Index: Date
##   Date       Sales   day month
##   <date>     <dbl> <dbl> <chr>
## 1 2019-01-01 549.      5 01
## 2 2019-01-01  80.2     8 03
## 3 2019-01-01 341.      3 03
## 4 2019-01-01 489.     27 01
## 5 2019-01-01 634.      8 02
## 6 2019-01-01 628.     25 03
```

We have a tibble with 6 rows and 4 columns, sorted from the earliest date to the latest.

PERFORMING ANOMALY DETECTION

Decomposing and plotting

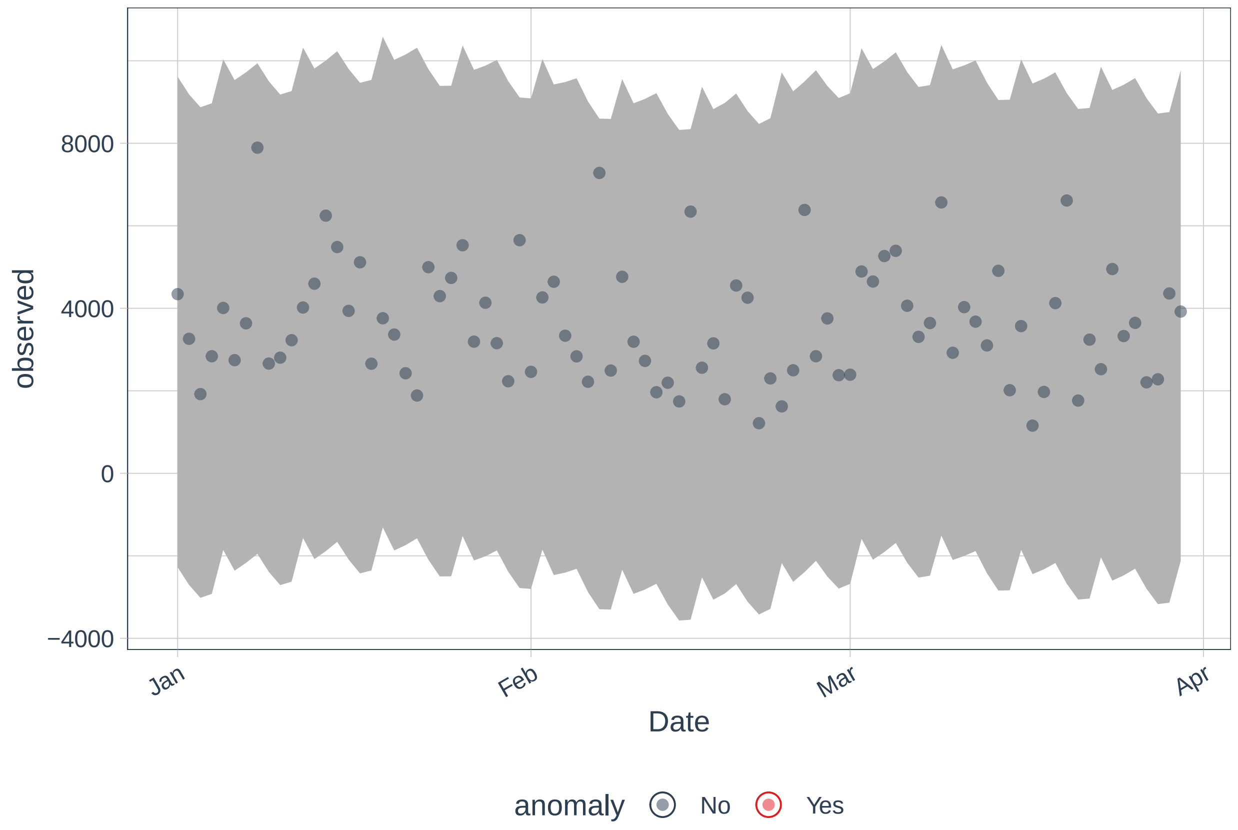
```r
# decomposing time using the decompose() function
library(tibbletime)
sales_anom<- sale_forecast %>%
  group_by(Date) %>%
  summarise(Sales = sum(Sales)) %>%
  time_decompose(Sales) %>%
  anomalize(remainder) %>%
  time_recompose()
```

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

```
## trend = 30 days
```

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

```r
sales_anom %>% plot_anomalies(time_recomposed = TRUE, ncol = 3, alpha_dots = 0.5)
```


8000
4000
0
−4000
observed
Jan
Feb
Mar
Apr
Date
anomaly
No
Yes