Part 4: Anomaly Detection

Victoria Maina

2022-04-02

Anomaly Detection

Defining the question

a) Specifying the Question

Identify anomalies in the dataset = fraud detection

b) Defining the metrics for success

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

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.

d) Recording the Experimental Design

Define the question, the metric for success, the context, experimental design taken. Read and explore the given dataset. Identify anomalies in the dataset = fraud detection

e) Relevance of the data

The data used for this project will inform the marketing department on the most relevant marketing strategies that will result in the highest no. of sales (total price including tax)

[http://bit.ly/CarreFourSalesDataset].

Loading Libraries

```
# loading libraries
library(data.table)
library(ggplot2)
library(tibble)
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 dplyr   1.0.8
## v readr   2.1.2     v stringr 1.4.0
## v purrr   0.3.4     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks tibbletime::filter(), stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## 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(dbplyr)
```

```
##
## Attaching package: 'dbplyr'
```

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

```
library(timetk)
```

```
##
## Attaching package: 'timetk'
```

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

```
library(tibble)
library(mvtnorm)
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(tibbletime)
library(data.table)
library(dplyr)
```

loading the dataset

```
df<- read.csv("http://bit.ly/CarreFourSalesDataset")
#Lets preview the head
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
```

The dataset contains two columns that is sales and particular dates that those sales were done but dates are in string format.

```
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
```

```
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 for shape of the dataset

```
dim(df)
```

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

The dataset has 1000 records and 2 variables

checking for missing values

```
colSums(is.na(df))
```

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

There are no missing values in the dataset

Checking for data types

```
#data structure
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 ...
```

Date columns has a character data types thus needs to be converted to date using as.Date() while sales has numerical

Changing column date from character to date format

```
df$Date <- as.Date(df$Date, format = "%m/%d/%Y")
df$Date <- sort(df$Date, decreasing = FALSE)
str(df)
```

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

Convert the sales into a tibbletime object.

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

```
df <- as_tibble(df)
```

Anomaly Detection.

Decomposing

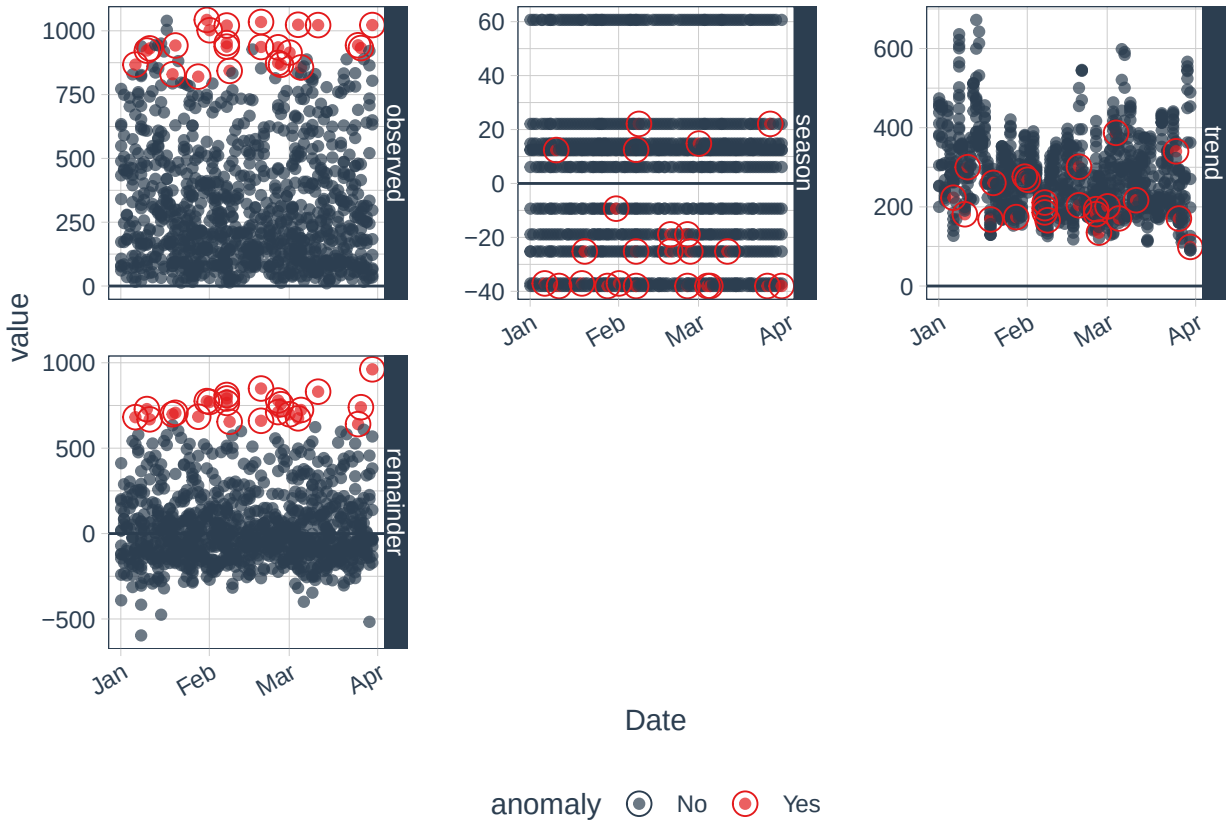
```
df %>%
time_decompose(Sales, method = 'stl', frequency = 'auto', trend = 'auto') %>%
anomalize(remainder, method = 'gesd', alpha = 0.1, max_anoms = 0.5) %>%
plot_anomaly_decomposition(ncol = 3, alpha_dots = 0.7)
```

```
## 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
```



Time Series Decomposition

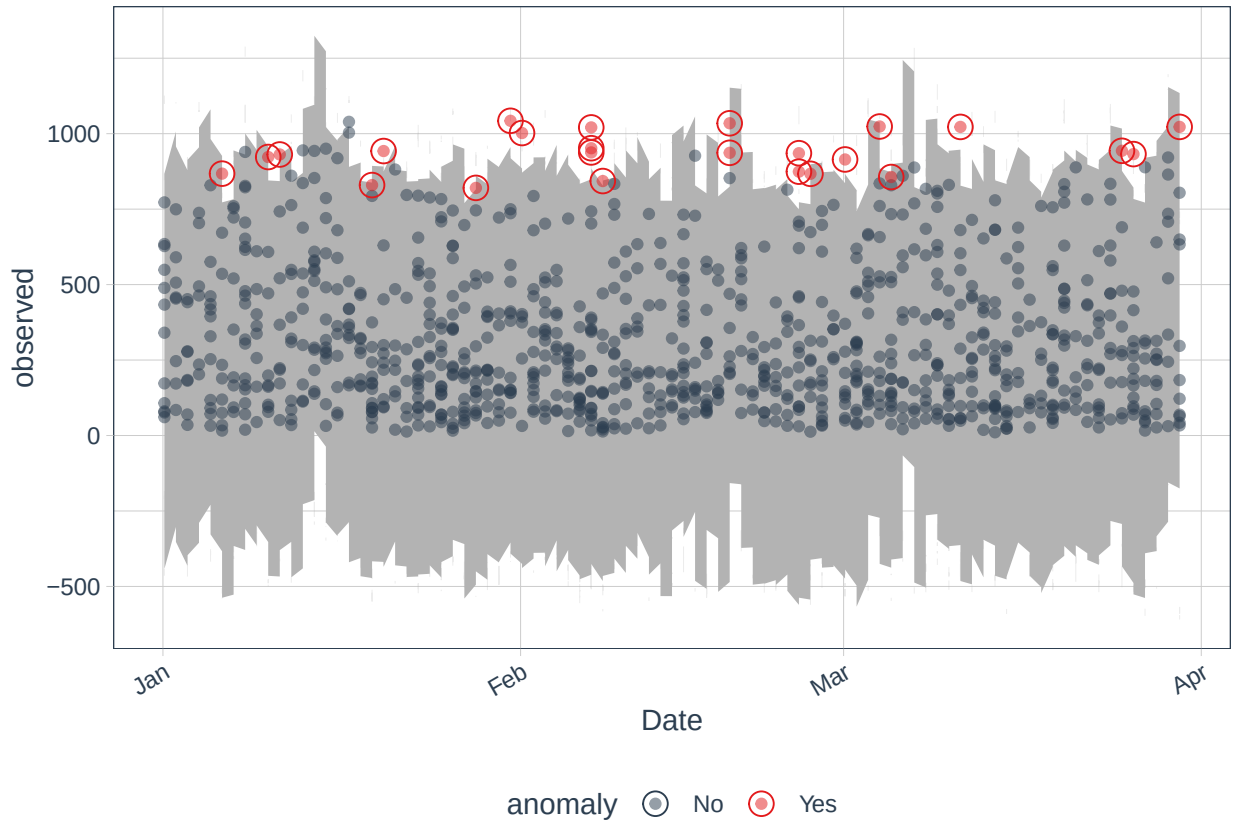
```
# Recomposing
# Using time_recompose() function to generate bands around the normal levels of observed values


df %>%
time_decompose(Sales, method = 'stl', frequency = 'auto', trend = 'auto') %>%
anomalize(remainder, method = 'gesd', alpha = 0.1, max_anoms = 0.1) %>%
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
```




```
anomalies = df %>%
time_decompose(Sales, method = 'stl', frequency = 'auto', trend = 'auto') %>%
anomalize(remainder, method = 'gesd', alpha = 0.05, max_anoms = 0.1) %>%
time_recompose() %>%
filter(anomaly == 'Yes')
```

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

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

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

```
anomalies
```

```
## # A time tibble: 20 x 10
## # Index: Date
##    Date                observed season trend remainder remainder_l1 remainder_l2
##    <dttm>                 <dbl>  <dbl> <dbl>     <dbl>        <dbl>        <dbl>
##  1 2019-01-06 03:00:00     868. -37.1  223.       682.        -699.         670.
##  2 2019-01-10 03:00:00     923.  12.4  181.       729.        -699.         670.
##  3 2019-01-19 03:00:00     830. -37.1  168.       699.        -699.         670.
##  4 2019-01-20 03:00:00     942. -25.2  261.       707.        -699.         670.
##  5 2019-01-28 03:00:00     820. -38.0  174.       684.        -699.         670.
```

```
##  6 2019-01-31 03:00:00    1043.  -9.34 276.       776.        -699.         670.
##  7 2019-02-01 03:00:00    1002. -37.1  269.       771.        -699.         670.
##  8 2019-02-07 03:00:00    1021.  12.5  197.       811.        -699.         670.
##  9 2019-02-07 03:00:00     952. -25.2  186.       791.        -699.         670.
## 10 2019-02-07 03:00:00     938. -38.0  211.       765.        -699.         670.
## 11 2019-02-19 03:00:00    1034. -18.9  204.       849.        -699.         670.
## 12 2019-02-25 03:00:00     935. -38.0  194.       779.        -699.         670.
## 13 2019-02-25 03:00:00     874. -18.9  182.       711.        -699.         670.
## 14 2019-02-26 03:00:00     867. -25.2  135.       757.        -699.         670.
## 15 2019-03-01 03:00:00     915.  14.8  201.       698.        -699.         670.
## 16 2019-03-04 03:00:00    1024. -38.0  387.       675.        -699.         670.
## 17 2019-03-05 03:00:00     856. -38.0  171.       724.        -699.         670.
## 18 2019-03-11 03:00:00    1022. -25.2  217.       831.        -699.         670.
## 19 2019-03-26 03:00:00     932.  22.1  170.       740.        -699.         670.
## 20 2019-03-30 03:00:00    1022. -38.0   98.8      962.        -699.         670.
## # ... with 3 more variables: anomaly <chr>, recomposed_l1 <dbl>,
## #   recomposed_l2 <dbl>
```

There are several anomalies in the carrefour dataset between the 6th February 2019 and 30th march 2019

##Conclusion There were several anomalies in the month of February and March

Recommendations

carrerfour should review sales that happen between February and march to investigate the reason why there were why anomalies between February and March.