Part4_anomaly detection

Martha Irungu

9/18/2020

Implementing Anomaly detection

Part 4: Anomaly Detection Anomaly detection, also known as outlier detection is the process of identifying extreme points or observations that are significantly deviating from the remaining data.Whereas in unsupervised learning, no labels are presented for data to train upon.

The objective of this task is fraud detection

#Loading and previewing the dataset

```
library("data.table")
forecasting<-fread("/Users/marthairungu/desktop/sales_forecasting.csv")
head(forecasting)
```

```
##         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
```

```
#Load the dplyr library
library(dplyr)
```

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

```
## 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
```

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

```
## ── Attaching packages ─────────────────────────────────────────────────
──────── tidyverse 1.3.0 ──

## ✓ ggplot2 3.3.2     ✓ purrr   0.3.4
## ✓ tibble  3.0.3     ✓ stringr 1.4.0
## ✓ tidyr   1.1.2     ✓ forcats 0.5.0
## ✓ readr   1.3.1

## ── 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::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 Det
ection!
## </> Learn more at: https://university.business-science.io/p/learning-labs-
pro </>
```

#Checking the structure of the data

```
str(forecasting)
```

```
## 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>
```

#Change date to as factor

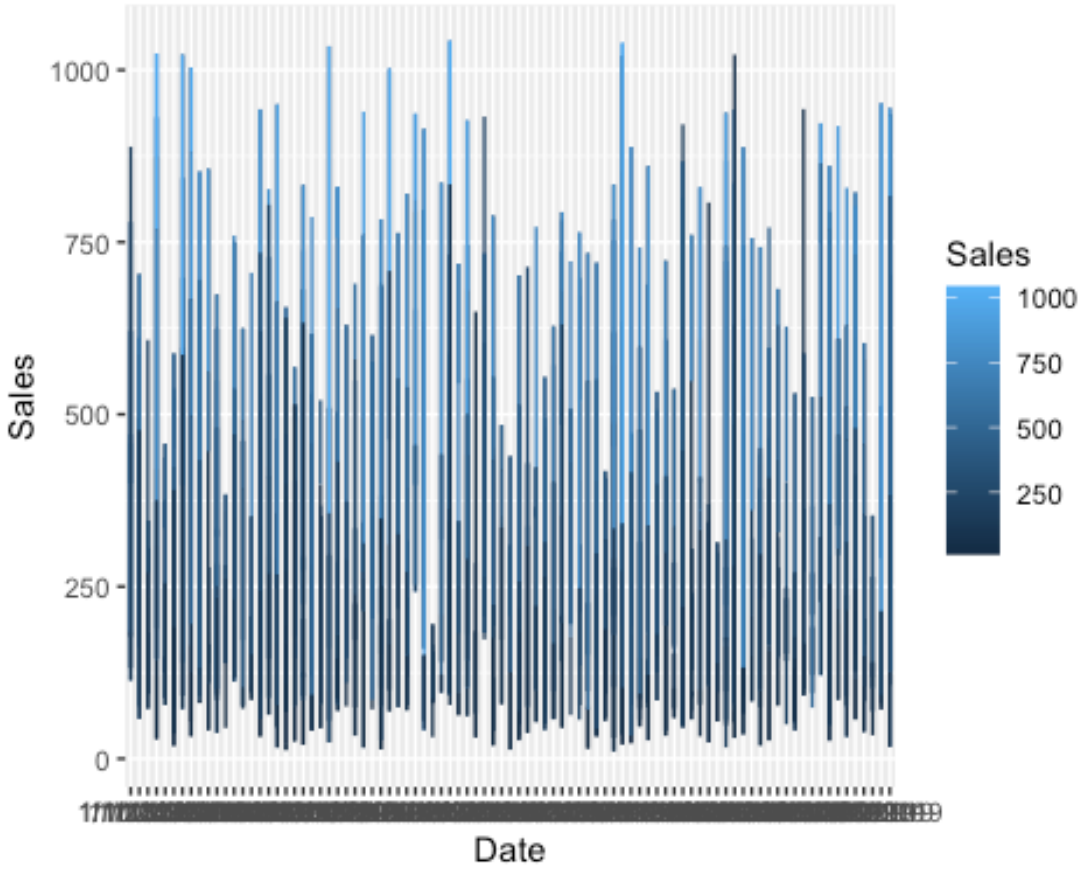
```
forecasting$Date<-as.factor(forecasting$Date)
str(forecasting)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  2 variables:
##  $ Date : Factor w/ 89 levels "1/1/2019","1/10/2019",..: 27 88 82 20 58 77
49 48 2 44 ...
##  $ Sales: num  549 80.2 340.5 489 634.4 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

#Date changed to as factor

#Plotting the sales data using ggplot2. Set the x-axis to the Date and the y-axis to the number of Sales.

```
library(ggplot2)
ggplot(forecasting, aes(x=Date, y=Sales, color=Sales)) +
  geom_line()
```



#Prepare the data for anomaly detection. The date should be in POSIX format

```
forecasting=forecasting[,c("Date","Sales")]
str(forecasting)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  2 variables:
##  $ Date : Factor w/ 89 levels "1/1/2019","1/10/2019",..: 27 88 82 20 58 77
49 48 2 44 ...
##  $ Sales: num  549 80.2 340.5 489 634.4 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

#Change the date format to POSIXct format

```
forecasting$Date<-as.POSIXct(paste(forecasting$Date),format = "%m/%d/%Y")
str(forecasting)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  2 variables:
##  $ Date : POSIXct, format: "2019-01-05" "2019-03-08" ...
##  $ Sales: num  549 80.2 340.5 489 634.4 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

#Date changed to the POSIXct format

#Apply anomaly detection to plot the results

```
#AnomalyDetectionSales = AnomalyDetectionTs(forecasting, direction="pos", plo
t=TRUE, title = "Anomaly Detection")
#AnomalyDetectionSales$plot
```

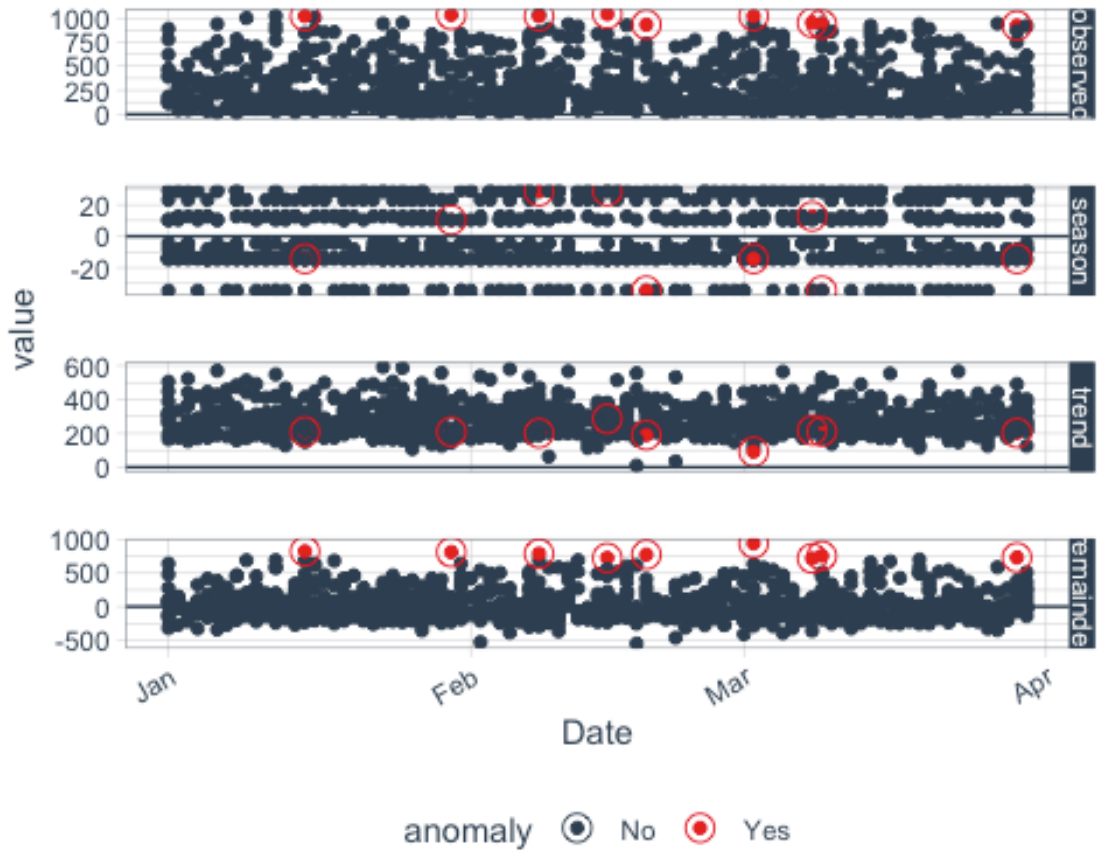
#Loading the libraries

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

#Decompose data using time_decompose() function in anomalize package. We will use stl method which extracts seasonality.

```
df = forecasting %>%
                          as.tibble()

## Warning: `as.tibble()` is deprecated as of tibble 2.0.0.
## Please use `as_tibble()` instead.
## The signature and semantics have changed, see `?as_tibble`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.

df %>%
  time_decompose(Sales, method = "stl",  frequency = "auto", trend = "auto")
%>%
  anomalize(remainder, method = "gesd", alpha = 0.05, max_anoms = 0.1) %>%
  plot_anomaly_decomposition()

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

## Note: Index not ordered. tibbletime assumes index is in ascending order. R
esults may not be as desired.

## frequency = 12 seconds

## Note: Index not ordered. tibbletime assumes index is in ascending order. R
esults may not be as desired.

## trend = 12 seconds
```

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



#We observe observed, season,trend and remainder sales values and extent to which they oscilate accross the dates.

#Plot the data again by recomposing data

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

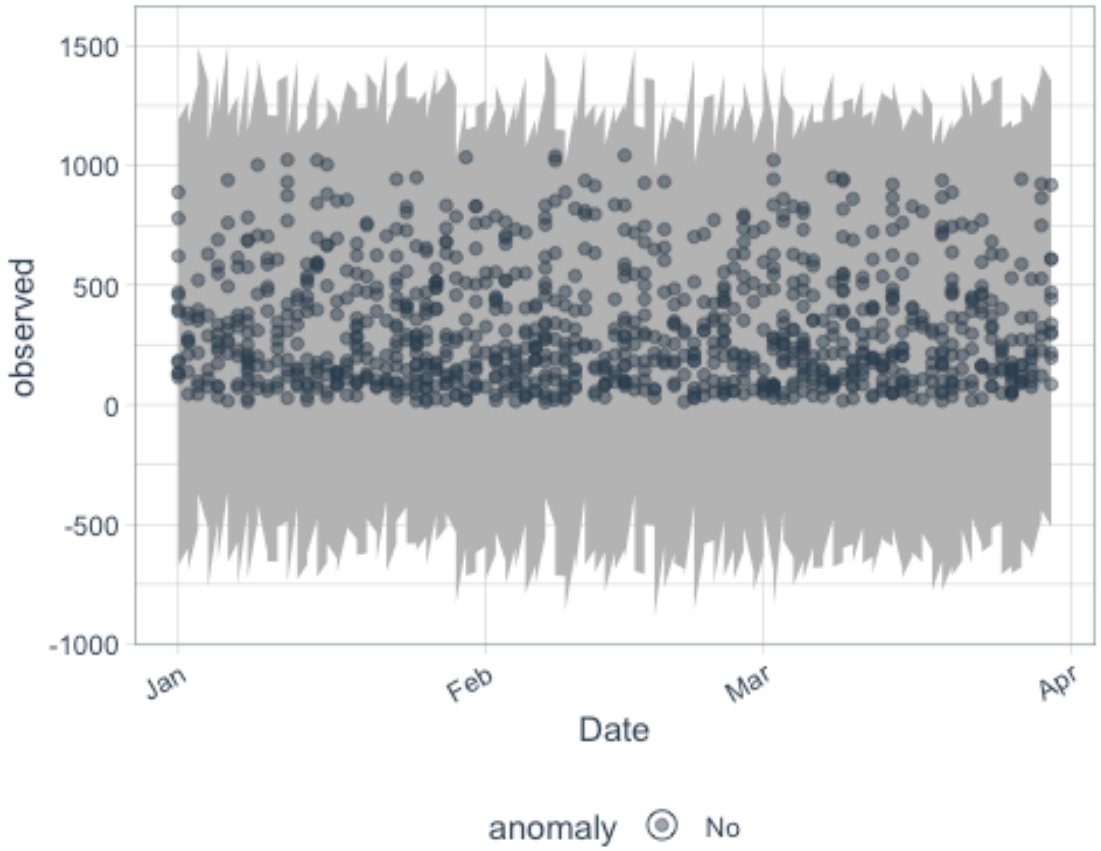
```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = Date
```

```
## Note: Index not ordered. tibbletime assumes index is in ascending order. R
esults may not be as desired.
```

```
## frequency = 12 seconds
```

```
## Note: Index not ordered. tibbletime assumes index is in ascending order. R
esults may not be as desired.
```

```
## trend = 12 seconds
```



#Conclusion

We observe that the trend is to have products sales range between 200-600. We have sales values of above 600 to 1,00 reflected as anomalies in the month of Fenruary and March.