Anomaly detection

Stacy

2022-04-02

```
# Define the question : You have also been requested to check whether there are any anomalies in the gi
# The objective of this task being fraud detection.
```

```
# The metric for success:Anomaly detection in the dataset

# The context:

   #Experimental design taken:
##Problem Definition
## Data Sourcing
## Check the Data
## Perform Data Cleaning
## Perform Exploratory Data Analysis
## Implement the Solution
## Challenge the Solution
## Follow up Questions

# The appropriateness of the available data to answer the given question:The data provided information
```

```
# install.packages("anomalize")
# install.packages("devtools")
#devtools::install_github("twitter/AnomalyDetection")
```

```
library(tidyverse)#tidyverse packages like dplyr, ggplot, tidyr
```

```
## -- 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)#tidy anomaly detection
```

```
## == 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 </>
```

```r
library(anomaly)
```

```
##
## Attaching package: 'anomaly'

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

```r
library(ggplot2)
library(AnomalyDetection)
library(tibble)
library(dbplyr)
```

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

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

```r
library(tibbletime)
```

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

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

```r
library(timetk)
library(outliers)
library(mvtnorm)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

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

```r
library(psych)
```

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

```
## The following object is masked from 'package:outliers':
##
##     outlier

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

```
library(lubridate)
```

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

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

```
data<- read.csv('http://bit.ly/CarreFourSalesDataset')
head(data)
```

```
##        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
```

```
colnames(data)
```

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

```
dim(data)
```

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

```
str(data)
```

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

```
#  sum of null values per column
colSums(is.na(data))
```

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

```r
data$Date =as.Date(data$Date)
str(data)
```

```
## 'data.frame':    1000 obs. of  2 variables:
##  $ Date : Date, format: "0001-05-20" "0003-08-20" ...
##  $ Sales: num  549 80.2 340.5 489 634.4 ...
```

```r
data <- as_tibble(data)
class(data)
```

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

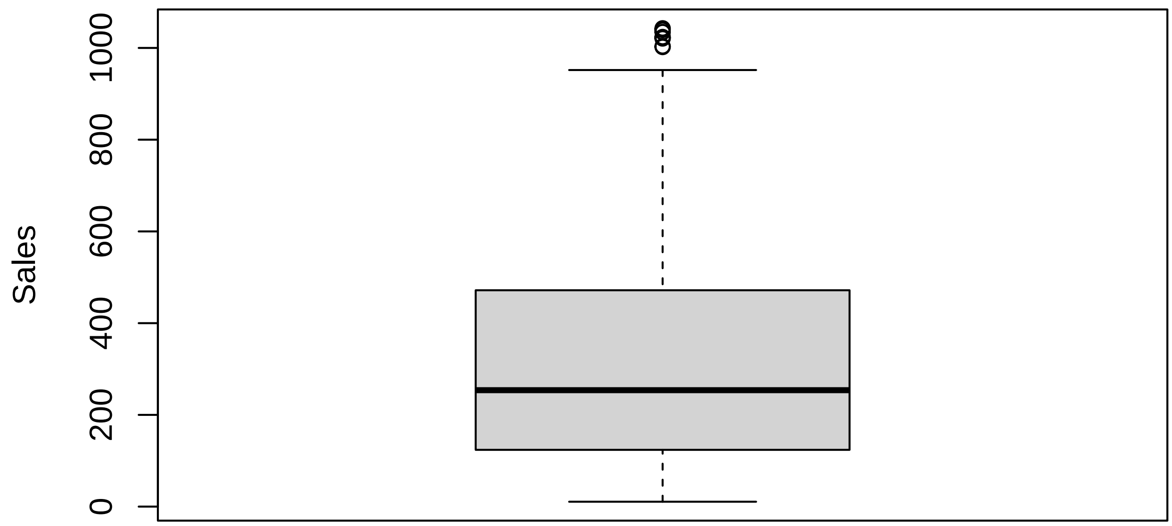
```r
summary(data)
```

```
##       Date                 Sales
##  Min.   :0001-01-20   Min.   :  10.68
##  1st Qu.:0001-11-20   1st Qu.: 124.42
##  Median :0002-07-20   Median : 253.85
##  Mean   :0002-08-02   Mean   : 322.97
##  3rd Qu.:0003-04-20   3rd Qu.: 471.35
##  Max.   :0003-12-20   Max.   :1042.65
##  NA's   :587
```

```r
dup<- data[duplicated(data),]
head(dup)
```

```
## # A tibble: 4 x 2
##   Date   Sales
##   <date> <dbl>
## 1 NA      87.2
## 2 NA     176.
## 3 NA     217.
## 4 NA     471.
```

```r
# outliers of numerical columns
boxplot(data$Sales, ylab= "Sales")
```




```r
grubbs.test(data$Sales)
```

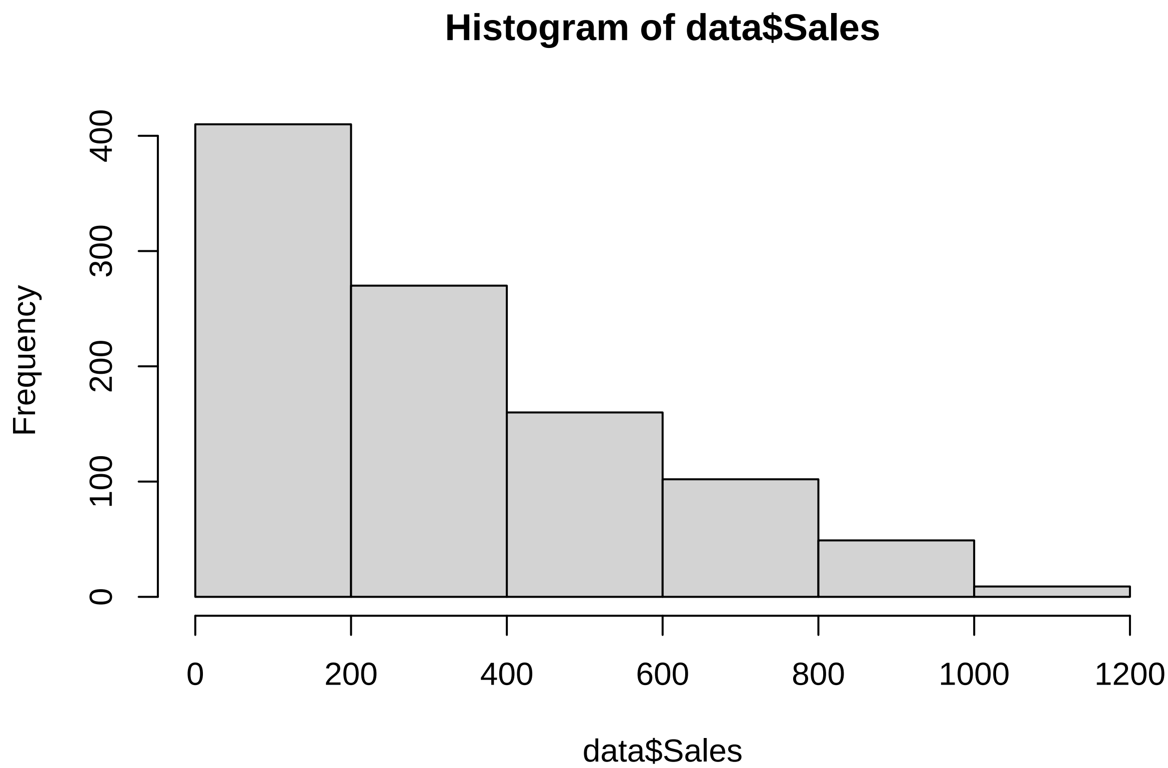
```
##
##  Grubbs test for one outlier
##
## data:  data$Sales
## G = 2.92691, U = 0.99142, p-value = 1
## alternative hypothesis: highest value 1042.65 is an outlier
```

```r
which.max(data$Sales)
```

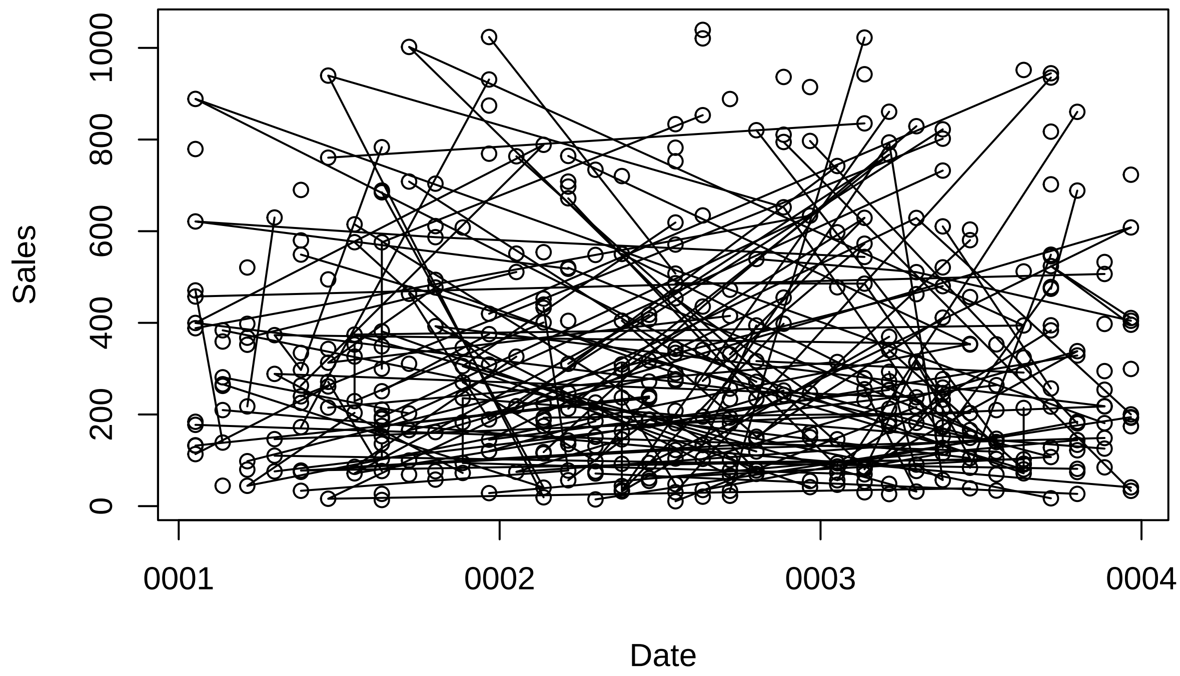
```
## [1] 351
```

Conclusion: The highest value 1042.65 is an outlier. The P value is a number between 0 and 1 that measures how much evidence that the testing point is an outlier. values near to 1 indicate weaker evidence that the test is an outlier. It is not appropirate to use Grubbs test to check for anormalies in the time series data because: Data may contain repeating seasonal patterns It test for one anormaly at a time

```r
hist(data$Sales, breaks = 6)
```



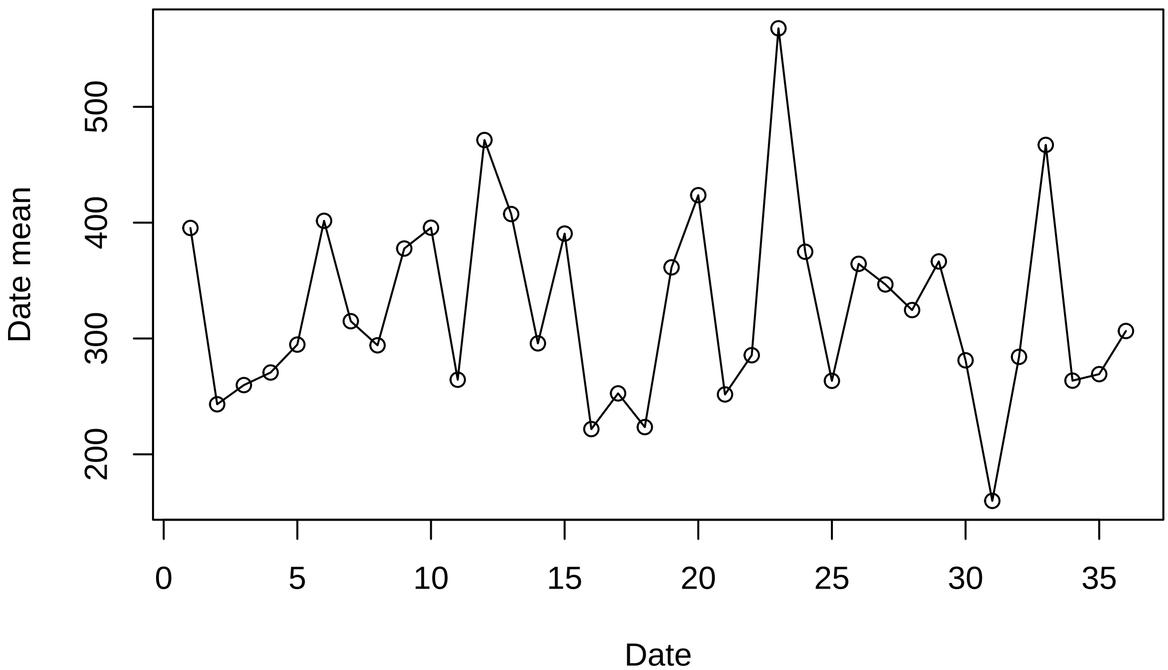

```
plot(Sales ~ Date, data = data, type = "o")
```



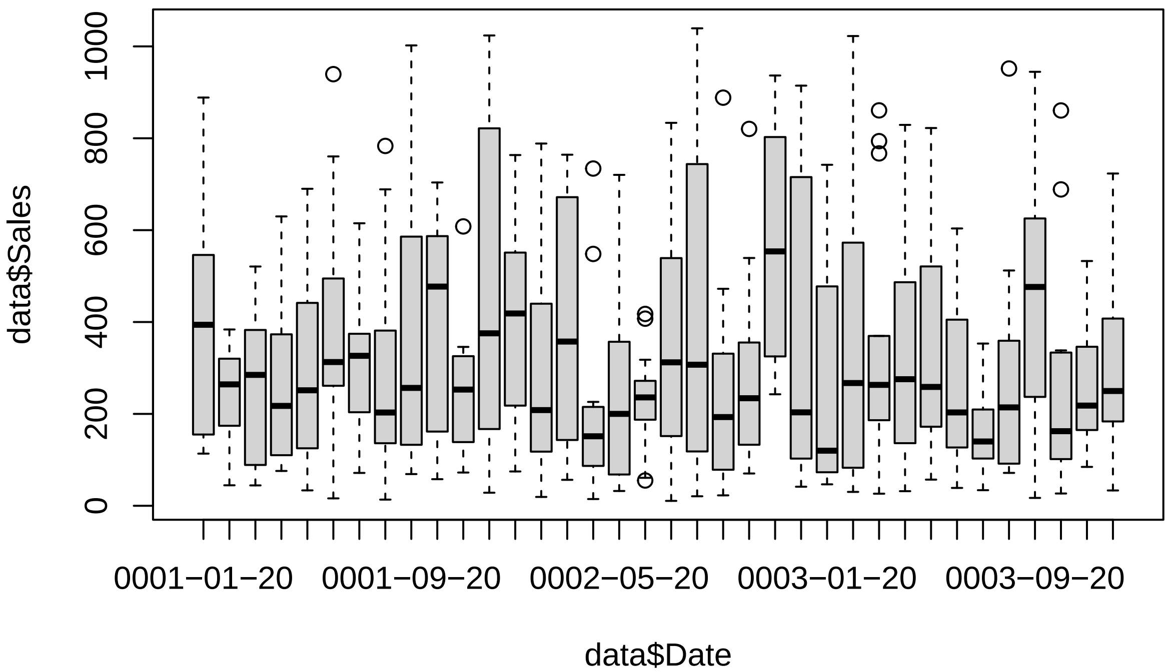

```r
date_mean <- tapply(data$Sales, data$Date, FUN = mean)
date_mean
```

```
## 0001-01-20 0001-02-20 0001-03-20 0001-04-20 0001-05-20 0001-06-20 0001-07-20
##   395.4318   243.1879   259.7661   270.6148   294.7236   401.5783   314.9160
## 0001-08-20 0001-09-20 0001-10-20 0001-11-20 0001-12-20 0002-01-20 0002-02-20
##   294.0962   377.6679   395.6610   264.3703   471.3421   407.4228   295.7820
## 0002-03-20 0002-04-20 0002-05-20 0002-06-20 0002-07-20 0002-08-20 0002-09-20
##   390.5663   221.7724   252.5941   223.4941   361.4105   423.7214   251.6842
## 0002-10-20 0002-11-20 0002-12-20 0003-01-20 0003-02-20 0003-03-20 0003-04-20
##   285.5475   567.7691   374.8736   263.4366   364.4614   346.6553   324.5366
## 0003-05-20 0003-06-20 0003-07-20 0003-08-20 0003-09-20 0003-10-20 0003-11-20
##   366.5223   281.1451   159.8065   284.1262   467.1279   263.6025   269.2047
## 0003-12-20
##   306.4626
```

```r
# Plot the monthly means
plot(date_mean, type = "o", xlab = "Date", ylab = "Date mean")
```




```r
# Create a boxplot of date against sales
boxplot(data$Sales ~ data$Date, data = data)
```



data$Sales
0
200
400
600
800
1000
0001−01−20
0001−09−20
0002−05−20
0003−01−20
0003−09−20
data$Date