Kiriakos\_MSDS 664\_X70\_Week 7: Anomaly Detection

Cathy Kiriakos

April 22, 2020

## Week 7: Anomaly Detection using Earthquake data

The data set that we will investigate is a listing of earthquakes from 1900-1998 whose magnitudes were greater than 7.0. The purpose of this investigation is to gain both an understanding of the data; and to identify anomalies in the data by leveraging anomaly detection packages in R. We will perform an exploratory data analysis, identify two or more outlier detection techniques, and follow it up by discussing our findings and providing a suggestion on best application.

We will first get started by loading our data:

eq <- read.csv("C:/Users/Cathy/Downloads/wk7\_eq.csv",header = TRUE, colClasses=c("character","numeric"))

Now to get a quick view of our data:

summary(eq)

## year earthquakes   
## Length:99 Min. : 6.00   
## Class :character 1st Qu.:15.00   
## Mode :character Median :20.00   
## Mean :20.02   
## 3rd Qu.:24.00   
## Max. :41.00

head(eq)

## year earthquakes  
## 1 1900 13  
## 2 1901 14  
## 3 1902 8  
## 4 1903 10  
## 5 1904 16  
## 6 1905 26

str(eq)

## 'data.frame': 99 obs. of 2 variables:  
## $ year : chr "1900" "1901" "1902" "1903" ...  
## $ earthquakes: num 13 14 8 10 16 26 32 27 18 32 ...

We will convert year to date time format:

library(lubridate)

##   
## Attaching package: 'lubridate'

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

eq$year <- round\_date(as.Date(eq$year, format='%Y'), unit='year')

Test to make sure that the date time format took:

str(eq)

## 'data.frame': 99 obs. of 2 variables:  
## $ year : Date, format: "1900-01-01" "1901-01-01" ...  
## $ earthquakes: num 13 14 8 10 16 26 32 27 18 32 ...

head(eq)

## year earthquakes  
## 1 1900-01-01 13  
## 2 1901-01-01 14  
## 3 1902-01-01 8  
## 4 1903-01-01 10  
## 5 1904-01-01 16  
## 6 1905-01-01 26

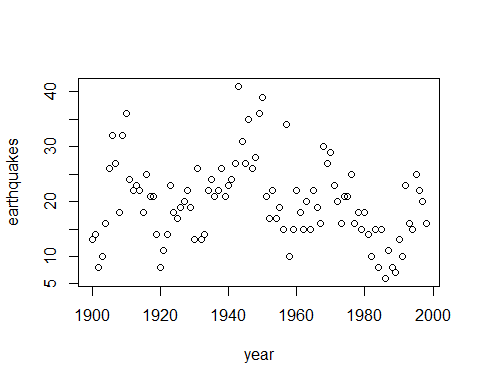
tail(eq)

## year earthquakes  
## 94 1993-01-01 16  
## 95 1994-01-01 15  
## 96 1995-01-01 25  
## 97 1996-01-01 22  
## 98 1997-01-01 20  
## 99 1998-01-01 16

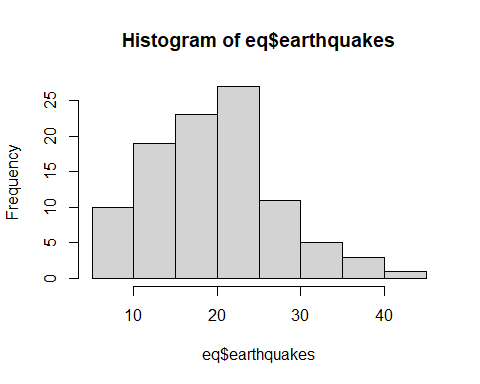
We can see that our data has two variables year & earthquakes, the data starts in 1900 and ends in 1998. Our maximum number of earthquakes is 41 and min is 6.

Now to get a quick visual on the data:

plot(eq)

 For good measure let’s look at a histogram of our distribution, we can see where our outliers are in the upper bounds of the distribution.

hist(eq$earthquakes)



Now that we have a good visual on this data, we can move on to outlier detection techniques. For this investigation we will leverage the R grDevices package, so we will get started by calling that library:

library(grDevices)

We will use the boxplot.stats, and get a view of its output, details on this function are below for reference:

boxplot.stats(x, coef = 1.5, do.conf=TRUE, do.out=TRUE)

x a numeric vector for which the boxplot will be constructed (NAs and NaNs are allowed and omitted). coef this determines how far the plot ``whiskers’’ extend out from the box. If coef is positive, the whiskers extend to the most extreme data point which is no more than coef times the interquartile coef from the box. A value of zero causes the whiskers to extend to the data extremes. do.conf,do.out logicals; if FALSE, the conf or out component respectively will be empty in the result.

bp <- boxplot.stats(eq$earthquakes,coef = 1.5 ,do.conf = TRUE, do.out = TRUE)  
bp

## $stats  
## [1] 6 15 20 24 36  
##   
## $n  
## [1] 99  
##   
## $conf  
## [1] 18.57084 21.42916  
##   
## $out  
## [1] 41 39

Our stats output gave us a vector of length 5, containing the extreme of the lower whisker(6), the lower ‘hinge’ (15), the median (20), the upper ‘hinge’ (24), and the extreme of the upper whisker(36). N tells us that we have 99 observations, and the conf provides us the the upper and lower extremes of the “notch.” Last but not least out, provides us with the values that fall outside of the extremes of our wiskers where’re we are only getting outliers on the high range of our distribution.

Moving on with our investigation of outliers in the earthquakes data set, we will leverage the envStats package in R and use ronserTest(). First for a dive into the functionality of the rest, it is designed to test for outliers that are either much smaller or much larger than the rest of the data. Rosner’s approach is designed to avoid the problem of masking, where an outlier that is close in value to another outlier can go undetected. Rosner’s test is appropriate only when the data, excluding the suspected outliers, are approximately normally distributed, and when the sample size is greater than or equal to 25.

From our box plot investigation we have two outliers, for this test we will assign a couple more to K which is the number of outliers that we believe we have so I will use a K of 4

library(EnvStats)

##   
## Attaching package: 'EnvStats'

## The following objects are masked from 'package:stats':  
##   
## predict, predict.lm

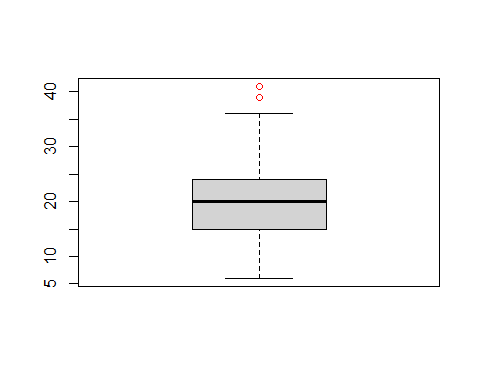
## The following object is masked from 'package:base':  
##   
## print.default

rosnerTest(eq$earthquakes, k= 4, warn = FALSE)

## $distribution  
## [1] "Normal"  
##   
## $statistic  
## R.1 R.2 R.3 R.4   
## 2.888489 2.750003 2.434227 2.526802   
##   
## $sample.size  
## [1] 99  
##   
## $parameters  
## k   
## 4   
##   
## $alpha  
## [1] 0.05  
##   
## $crit.value  
## lambda.1 lambda.2 lambda.3 lambda.4   
## 3.380651 3.377176 3.373658 3.370097   
##   
## $n.outliers  
## [1] 0  
##   
## $alternative  
## [1] "Up to 4 observations are not\n from the same Distribution."  
##   
## $method  
## [1] "Rosner's Test for Outliers"  
##   
## $data  
## [1] 13 14 8 10 16 26 32 27 18 32 36 24 22 23 22 18 25 21 21 14 8 11 14 23 18  
## [26] 17 19 20 22 19 13 26 13 14 22 24 21 22 26 21 23 24 27 41 31 27 35 26 28 36  
## [51] 39 21 17 22 17 19 15 34 10 15 22 18 15 20 15 22 19 16 30 27 29 23 20 16 21  
## [76] 21 25 16 18 15 18 14 10 15 8 15 6 11 8 7 13 10 23 16 15 25 22 20 16  
##   
## $data.name  
## [1] "eq$earthquakes"  
##   
## $bad.obs  
## [1] 0  
##   
## $all.stats  
## i Mean.i SD.i Value Obs.Num R.i+1 lambda.i+1 Outlier  
## 1 0 20.02020 7.263242 41 44 2.888489 3.380651 FALSE  
## 2 1 19.80612 6.979585 39 51 2.750003 3.377176 FALSE  
## 3 2 19.60825 6.733865 36 11 2.434227 3.373658 FALSE  
## 4 3 19.43750 6.554729 36 50 2.526802 3.370097 FALSE  
##   
## attr(,"class")  
## [1] "gofOutlier"

What is interesting is that our Rosner Test is telling us that we do not have any outliers. But out boxplot is clearly stating that we have two at the top of our range.

boxplot(eq$earthquakes, outcol = "red")

 We will move onto additional investigations of the data set to see if any detection agrees with our boxplot results. I had some issues installing the package the standard install.packages approach, so the script below installed the latest development version, let’s see how it goes.

We will get a look at anomalies using the anomalize package. This package is designed for time series analysis, which is perfect for our times series of earthquakes throughout 1900-2000.

Providing some background on the the methods we will utilize with the anomalize package in R:

The STL method uses the stl() function from the stats package. STL works very well in circumstances where a long term trend is present. The Loess algorithm typically does a very good job at detecting the trend. However, it circumstances when the seasonal component is more dominant than the trend, Twitter tends to perform better[1].

The GESD method is used in Twitter’s AnomalyDetection package. It involves an iterative evaluation of the Generalized Extreme Studentized Deviate test, which progressively evaluates anomalies, removing the worst offenders and recalculating the test statistic and critical value. The critical values progressively contract as more high leverage points are removed[1].

1. Anomalize Methods.<https://cran.r-project.org/web/packages/anomalize/vignettes/anomalize_methods.html#b.-gesd>

library(dplyr)

##   
## Attaching package: 'dplyr'

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

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

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

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 </>

##   
## Attaching package: 'anomalize'

## The following object is masked from 'package:EnvStats':  
##   
## iqr

library(tseries)

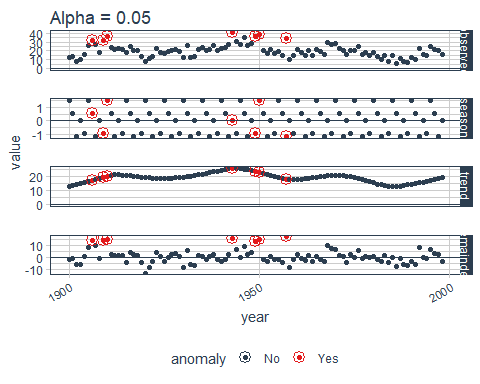
## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

library(ggplot2)  
eq.ts <- eq %>% as\_tibble()  
eq.ts %>%   
 time\_decompose(earthquakes, method="stl", frequency="auto", trend="auto") %>%  
 anomalize(remainder, method = "gesd", alpha = 0.05, max\_anoms = 0.2) %>%  
 plot\_anomaly\_decomposition()+  
 ggtitle("Alpha = 0.05")

## Converting from tbl\_df to tbl\_time.  
## Auto-index message: index = year

## frequency = 5 years

## trend = 25 years

 Our output tells us that there are 7 outliers in the upper bands of our distribution. Seven is significantly more than the zero that we got with our rosner test, which makes me think that we may want to adjust our alpha increasing the bands to see how that compares with our rosner test results.

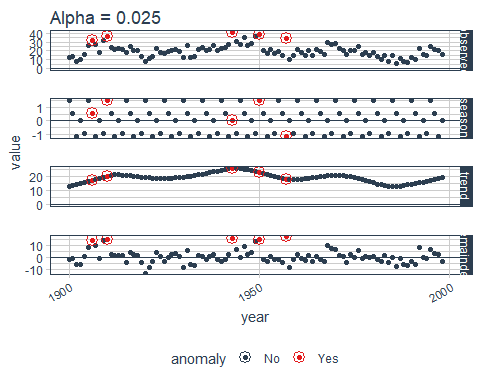
By adjusting our alpha we’re telling the algorithm to be less sensitive.

eq.ts %>%   
 time\_decompose(earthquakes, method="stl", frequency="auto", trend="auto") %>%  
 anomalize(remainder, method = "gesd", alpha = 0.025, max\_anoms = 0.2) %>%  
 plot\_anomaly\_decomposition()+  
 ggtitle("Alpha = 0.025")

## Converting from tbl\_df to tbl\_time.  
## Auto-index message: index = year

## frequency = 5 years

## trend = 25 years

 By changing our alpha to 0.025 we essentially lowered the sensitivity to the outliers, now having 5 outliers, which would have fallen into my anticipated number of anomalies when I conducted the rosner test.

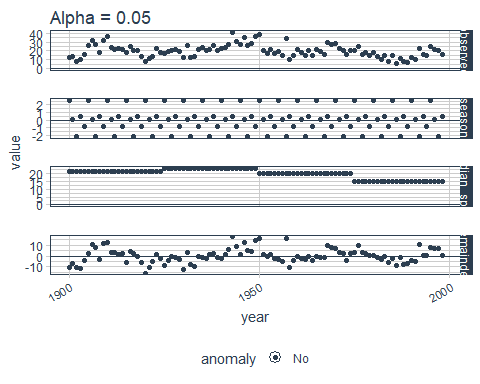
So far we’re taken a dive into the STL Decomposition methods; we can take a quick view to see how this method stack up against twitter decomposition. The Twitter method is a similar decomposition method to that used in Twitter’s AnomalyDetection package. The Twitter method works identically to STL for removing the seasonal component. The main difference is in removing the trend, which is performed by removing the median of the data rather than fitting a smoother. The median works well when a long-term trend is less dominant that the short-term seasonal component. This is because the smoother tends to overfit the anomalies.[2][2]Anomalize Methods <https://cran.r-project.org/web/packages/anomalize/vignettes/anomalize_methods.html#b.-gesd>

eq.ts %>%   
 time\_decompose(earthquakes, method="twitter", frequency="auto", trend="auto") %>%  
 anomalize(remainder, method = "gesd", alpha = 0.05, max\_anoms = 0.2) %>%  
 plot\_anomaly\_decomposition()+  
 ggtitle("Alpha = 0.05")

## Converting from tbl\_df to tbl\_time.  
## Auto-index message: index = year

## frequency = 5 years

## median\_span = 25 years

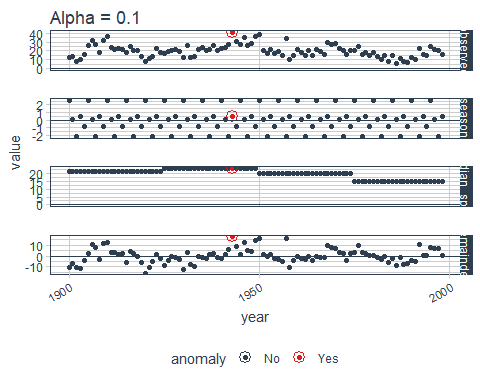
 Similar to our first test, the twitter decomposition method returned back no anomalies within the data. We will change the alpha to ten to make it more sensitive - and see how that changes our results using twitter twitter decomposition:

eq.ts %>%   
 time\_decompose(earthquakes, method="twitter", frequency="auto", trend="auto") %>%  
 anomalize(remainder, method = "gesd", alpha = 0.1, max\_anoms = 0.2) %>%  
 plot\_anomaly\_decomposition()+  
 ggtitle("Alpha = 0.1")

## Converting from tbl\_df to tbl\_time.  
## Auto-index message: index = year

## frequency = 5 years

## median\_span = 25 years

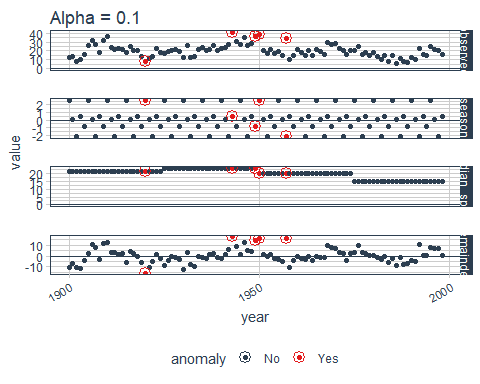
 For good measure lets see how this model changes when we leverage the IQR method for our remainders

eq.ts %>%   
 time\_decompose(earthquakes, method="twitter", frequency="auto", trend="auto") %>%  
 anomalize(remainder, method = "iqr", alpha = 0.1, max\_anoms = 0.2) %>%  
 plot\_anomaly\_decomposition()+  
 ggtitle("Alpha = 0.1")

## Converting from tbl\_df to tbl\_time.  
## Auto-index message: index = year

## frequency = 5 years

## median\_span = 25 years

 This gets us back to our 5 outliers, what interesting is that it’s looking at ouliers on our lower range - which makes more sense than having none especially considering its clear to see that we’ve got some very low observations that do seem to fall outside of the norm.

Going with my initial intuition on this data frame after getting our initial view using a box plot; I feel that the most reasonable approach to identifying outliers in this particular time series, is to use the anomalize package, leveraging the twitter method, and applying the IQR method for our remainder, and a slightly more sensitive alpha of 0.1. This approach yields results that make sense; obtaining outliers on the top and bottom our range.