Class 7: Anomaly Detection

Andrew Parnell
andrew.parnell@mu.ie

PRESS RECORD

https://andrewcparnell.github.io/intermediate_ML

Introduction to Anomaly Detection (AD)

- Anomaly detection refers to the process of identifying data points, observations, or patterns that deviate significantly from the norm or standard in a dataset. These might be critical incidents, such as errors, fraud, or system failures
- What you do with an anomaly will depend on the application; you might remove it, stop the whole experiment, or just ignore and note it down for later evaluation
- We will explore lots of different methods in R for performing anomaly detection with examples for all of them

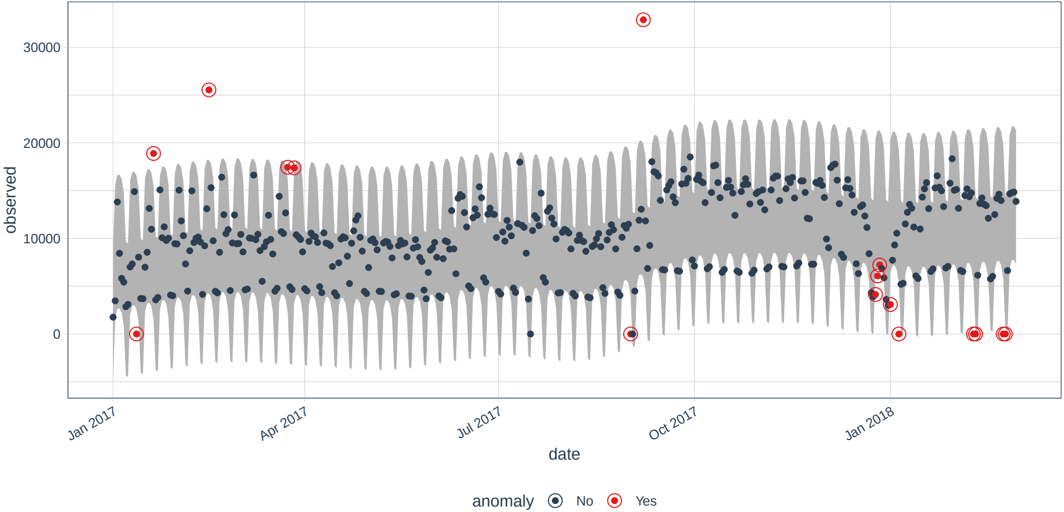
Anomaly Detection in R

- We will use 3 main packages for performing AD, and discuss some other popular methods
    1. `anomalize`: Implements a tidy AD algorithm that works well with dplyr and tidyr pipelines. Pretty user friendly (https://www.youtube.com/watch?v=Gk_HwjhIQJs)
    2. `tsoutliers`: Use for detecting outliers in time-series data. Integrates well with ARIMA modeling and other time-series forecasting methods, though pretty slow
    3. `stray`: Designed for AD in high-dimensional data. More aligned with machine learning techniques as it uses projection and clustering
- We mostly focus on finding anomalies in time series but AD can be used on any type of data

A simple example: CRAN downloads from anomalize

```
library(anomalize)
tidyverse_cran_downloads %>%
    filter(package == "dplyr") %>%
    ungroup() %>%
    time_decompose(count, method = "stl") %>%
    anomalize(remainder, method = "iqr") %>%
    time_recompose() %>%
    plot_anomalies(time_recomposed = TRUE)
```

A simple example: CRAN downloads from `anomalize` - plot



Types of Anomalies

- Point Anomalies. A single ‘anomalous’ data point. But beware, a sudden spike in e.g. energy usage on a hot day might be normal, but the same spike on a mild day could be anomalous.
- Collective Anomalies. A collection of ‘anomalous’ data points occurring together.
- Seasonal Anomalies. Anomalies that occur in a seasonal pattern within time-series data.
- Network Anomalies. Anomalies that occur in multiple time series simultaneously that might not be visible in a single series.

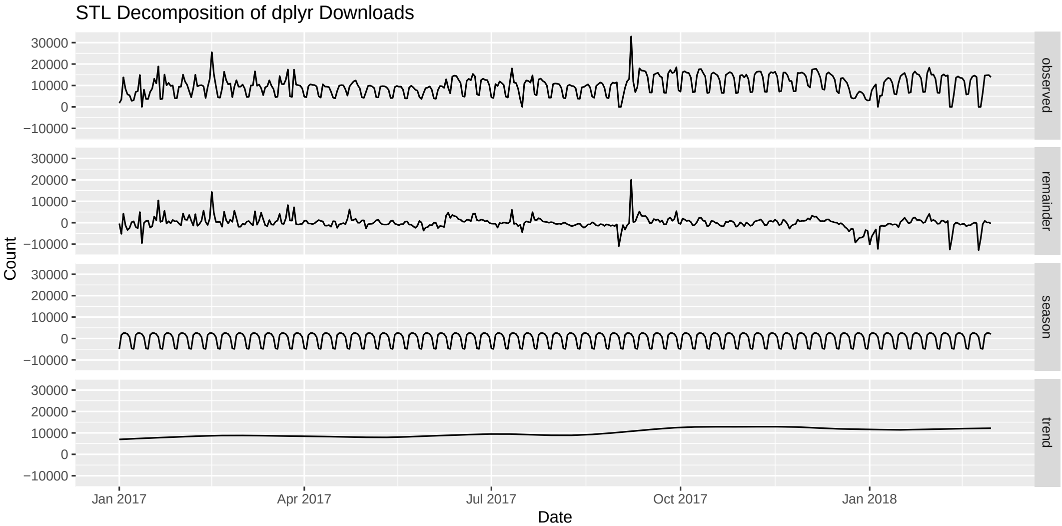
Anomaly Detection Methods

Often a time series model is fitted first and the anomaly detection routine is run on the leftover data (residuals).

- Basic methods: Z-scores, Inter-Quartile Range, Grubbs test, Control charts
- Statistical methods: Generalised-Extreme Studentised Deviate Test (GESD), Seasonal Hybrid ESD approach (Twitter / SH-ESD), Extreme value approaches
- Machine learning approaches. Dimension reduction approaches (stray), RNNs (followed by statistical methods)
- Hybrid methods. Using combinations of the above

Decomposition and Anomaly Detection

Common to perform Seasonal Trend and irregular decomposition using Loess (STL) and running AD on the components:



Basic Methods: G-ESD

- Iteratively tests for and removes the most extreme value as an outlier, allowing detection of multiple outliers in a dataset, as opposed to Grubbs test
- Uses a studentised range statistic, supposedly a robust method for identifying outliers in normally distributed data
- Effective for both small and large data sets, with an upper limit on the number of outliers found

Can be applied to a data set directly, but more commonly applied to the residuals. Provides a list of potential values up to the maximum number of anomalies allowed

Seasonal Hybrid ESD approach

- SH-ESD is an extension of the GESD test specifically for seasonal data by performing a seasonal decomposition and windowing of the data before running the GESD test
- The decomposition allows for the detection of seasonal anomalies, which are extreme relative to a particular season or time frame but might not be extreme in the overall dataset
- Not treated as a particular anomaly detection method but rather a decomposition method in `anomalize`

(Recently extended to work on streaming data rather than just windows)

Time Series Anomaly Detection with `tsoutliers`

Perhaps the most basic time series AD package is `tsoutliers`

- Package works by fitting ARIMA models to the data and then looking at different types of outliers
- Produces predictions and also potential adjustments to the data that would remove the outliers and make the data set 'cleaner'
- Allows for ARIMAX type data (e.g. time series data with extra regressors)

Types of outlier identified by `tsoutliers`

- Additive Outliers (AO), Innovational Outliers (IO), Level Shifts (LS), Temporary Changes (TC), and Seasonal Level Shifts (SLS).
- AO are sudden, abnormal spikes or drops in the time series that are not part of the usual pattern or trend.
- IO are irregularities that introduce a shock to the system, affecting the time series values both at the occurrence and subsequent periods.
- LS are sudden, lasting change in the level of the time series, reflecting a structural change in the process.
- TC are short-term anomalies where the time series deviates from its usual pattern for a brief period before returning to normal.
- SLS are similar to LS but occur in a seasonal pattern, indicating a permanent change in the seasonal component of the time series.

AO, IO and LS are the defaults looked for

Revision: ARIMA models

- Combination of AR and MA: ARIMA models blend AutoRegressive (AR) and Moving Average (MA) approaches, where AR models leverage past values and MA models use past forecast errors for prediction
- Integration for Non-Stationarity: The 'I' in ARIMA stands for 'Integrated' and involves differencing the data to achieve stationarity, essential for time series forecasting
- Parameter Specification: Characterized by three parameters (p, d, q) - 'p' for the order of the AR part, 'd' for the degree of differencing, and 'q' for the order of the MA part
- Flexibility and Adaptability: Suitable for a wide range of time series data, capable of modeling various patterns and structures in both stationary and non-stationary data

Can also have seasonal ARIMA models

Example of using tsoutliers

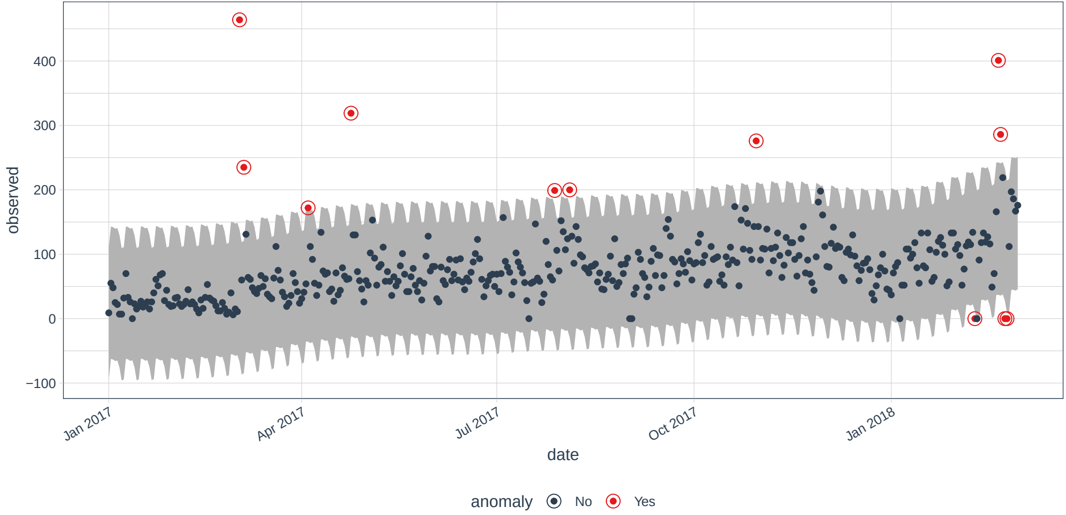
```r
library(tsoutliers)
data(hicp) # Mulivariate time series
tso(y = log(hicp$`011300`))
```

```
## Series: log(hicp$`011300`)
## Regression with ARIMA(0,1,0)(1,0,0)[12] errors
##
## Coefficients:
##         sar1     AO15     TC38    AO120   TC162    LS171
##       0.8268  -0.0109  -0.0142  -0.0092  0.0136  -0.0137
## s.e.  0.0365   0.0026   0.0033   0.0026  0.0033   0.0036
##
## sigma^2 = 2.257e-05:  log likelihood = 888.79
## AIC=-1763.58   AICc=-1763.07   BIC=-1739.6
##
## Outliers:
##   type ind    time  coefhat  tstat
## 1   AO  15 1996:03 -0.01093 -4.286
## 2   TC  38 1998:02 -0.01423 -4.283
## 3   AO 120 2004:12 -0.00921 -3.611
## 4   TC 162 2008:06  0.01365  4.109
## 5   LS 171 2009:03 -0.01373 -3.808
```

Example of anomalize package

```r
library(anomalize)
tidyverse_cran_downloads %>%
    filter(package == "dplyr") %>%
    ungroup() %>%
    time_decompose(count, method = "stl") %>%
    anomalize(remainder, method = "iqr") %>%
    time_recompose() %>%
    plot_anomalies(time_recomposed = TRUE)
```

Output of anomalize



Machine Learning Methods: `stray` approach

- stray = STReam AnomalY. Ideal for high dimensional data
- Uses k-nearest neighbours to find the distances between the observations in a high dimensional space
- Uses ideas from Extreme value theory (EVT) to define a threshold and identify the gaps between the observations
- Enables the method to capture both ‘in-liers’ and outliers
- Produces both a list of outliers and an outlier score for further analysis

Example of using the stray package

Very fast and pretty good

```
library(stray)
set.seed(123)
multivariate_data <- cbind(rnorm(100), rnorm(100))
results <- find_HDoutliers(multivariate_data,
                           alpha = 0.2, # Tail value for outliers
                           k = 10) # Number of neighbours for knn
most_outlying <- which.max(results$out_scores) # Useful to see
multivariate_data[(most_outlying-2):(most_outlying+2),]
```

Other machine learning approaches

Lots of other unsupervised approaches are used before running an AD algorithm

- K-means and mixture models (covered yesterday) commonly used
- DBSCAN is a spatial clustering algorithm that differentiates between core points (inside a cluster), border points (on the edge of a cluster), and noise points (isolated, outlier points)
- Isolation Forests is a version of random forests that spots data points that are commonly split at the top of a tree and are therefore likely to be outliers (R package `isotree`)

Summary

- Lots of different AD techniques; most based on quite traditional statistical methods
- Usually perfrom a time series analysis, which can be quite simple or very complex, before running the AD algorithm
- Lots of different types of anomaly which may or may not be appropriate to different data problems
- See the examples in the script folder for more details on what these packages can do