Project IV: PageRank and Anomaly Detection

DS 5494 - Statistical Data Mining II

Willliam Ofosu Agyapong* University of Texas at El Paso (UTEP)

October 25, 2022

Contents

*woagyapong@miners.utep.edu

1 PageRank

1.1 Part (a): Obtain the link matrix $\mathbb{L}$

```
include_graphics("webpage-links.png")
```

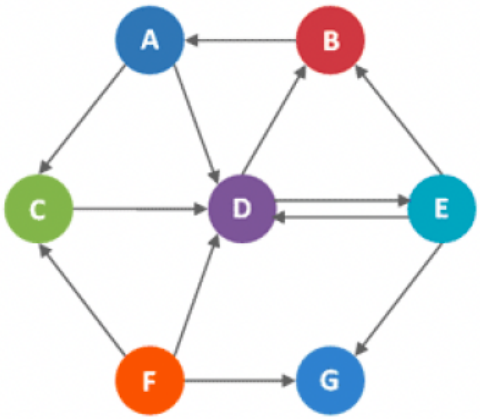

Figure 1: The original graph showing Links among several webpages

1.1.1 Link matrix for network in Figure 1

From Figure 1, we can derive the link matrix, $\mathbb{L}$, as follows where we let $L_{ij} = 1$ if page j points to page i, and 0 otherwise. We utilized the `graph_from_literal()` function from `igraph` package to obtain the link matrix by specifying the vertices and indicating the directed connections among vertices with the "-+" operator.

```
webgraph <- graph_from_literal(A, B, C, D, E, F, G, A-+C, A-+D, B-+A, C-+D, D-+B,
                               D-+E, E-+B, E-+D, E-+G, F-+C, F-+D, F-+G)

# Extract the adjacency matrix of the network
L <- as_adjacency_matrix(webgraph, sparse = F)

data.frame(L) %>%
    kable(caption = "Link matrix for network in Figure 1", booktabs=T,
          linesep="") %>%
    kable_styling(latex_options = c("HOLD_position"))
```

Table 1: Link matrix for network in Figure 1

	A	B	C	D	E	F	G
A	0	0	1	1	0	0	0
B	1	0	0	0	0	0	0
C	0	0	0	1	0	0	0
D	0	1	0	0	1	0	0
E	0	1	0	1	0	0	1
F	0	0	1	1	0	0	1
G	0	0	0	0	0	0	0

1.2 Part (b): Reproduction of the graph in Figure 1 from the link matrix

```
set.seed(110)
webgraph2 <- graph_from_adjacency_matrix(L)
V(webgraph2)$color <- c("dodgerblue", "tomato", "lightgreen", "violet", "cyan",
                        "orange", "dodgerblue")
plot(webgraph2, edge.arrow.size=0.5)
```

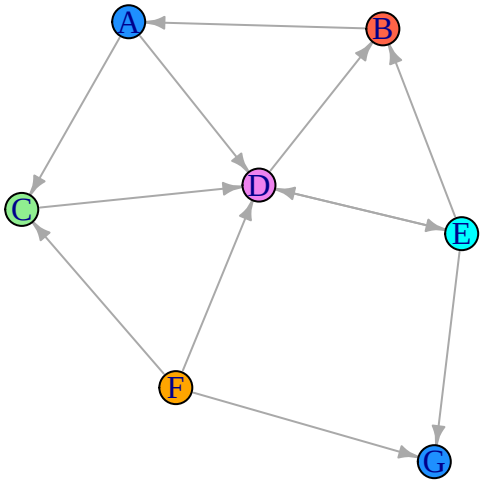

Figure 2: Webpages graph constructed from the link matrix obtained in part (a)

Comparing Figure 1 to Figure 2, we observe the same network structure, that is, we have the same connections among the nodes as before. We can therefore be confident that the link matrix was correctly generated.

1.3 Part (c): Computing PageRank score for each webpage

For this, we use the `page.rank()` function from the `igraph` package by supplying our graph object from part (b) and leaving all other parameters at their default settings.

```
pagerank <- page.rank(webgraph)$vector
pagerank_df <- data.frame(Webpage=names(pagerank), PageRank=pagerank)
```

```
kable(pagerank_df, caption = "PageRank scores for each webpage", booktabs=T,
      col.names = c("Webpage", "PageRank score"), row.names = F, linesep="")%>%
    kable_styling(latex_options = c("HOLD_position")) %>%
    kable_classic()
```

Table 2: PageRank scores for each webpage

Webpage	PageRank score
A	0.1858004
B	0.1819276
C	0.1189563
D	0.2602350
E	0.1417618
F	0.0311619
G	0.0801570

```
ggplot(pagerank_df, aes(x=reorder(Webpage, -PageRank), y=pagerank)) +
    geom_bar(stat = "identity", fill="navy", alpha=0.6) +
    labs(x="Webpage", y="PageRank scores", title = "Webpages ranked by their PageRank scores")
```

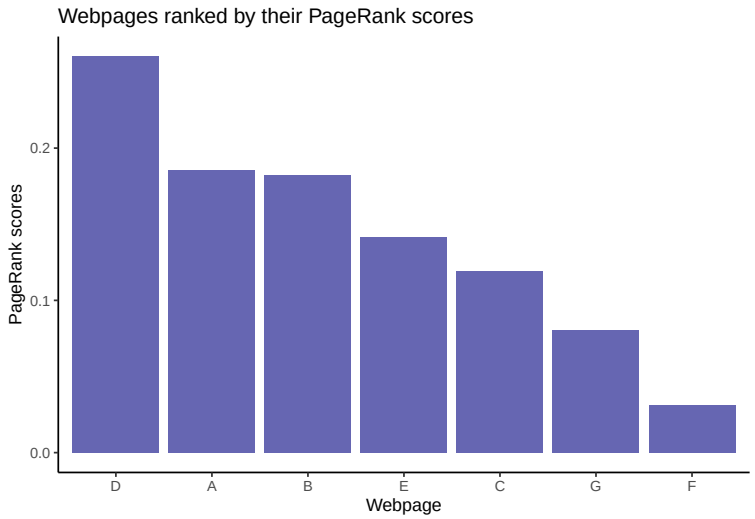

Figure 3: Barplot of PageRank scores for each webpage arranged in decreasing order of magnitude by their scores.

As suggested by Figure 3, the pages, D, A and B, are in the top-3 list, with web page F having the least PageRank score. From the webpage graph (Figure 2), D had the most in-coming links, so based on the mechanism of the PageRank algorithm, it comes as no surprise that it is on top of the list. This makes page D the most important webpage, followed by A, then B, and finally F.

2 Anomaly Detection

Here, we consider the `HTP` (high tech part) data available in the R package `ICSOutlier`. This data set contains the results of $p = 88$ numerical tests for $n = 902$ high-tech parts. Based on these results the producer considered all parts functional and all of them were sold. However two parts, **581** and **619**, showed defects in use and were returned to the manufacturer. These two observations can thus be considered as outliers and the objective is ***to detect them by re-examining the test data***.

2.1 Part (a): Bringing in the data

We use the following codes to retrieve the **HTP** data for our analysis.

```
# install.packages("ICSOutlier")
library(ICSOutlier)
data(HTP)
htp_dat <- HTP
# dim(dat); head(dat)
outliers.true <- c(581, 619)

dim(htp_dat)
```

```
## [1] 902  88
```

The above output confirms theat the HTP data set we obtained indeed contains 902 (high-tech parts designed for consumer products) observations and 88 variables (tests) as we expect.

2.1.1 Initial EDA

```
htp_dat %>%
tidyr::pivot_longer(everything(), names_to="variable", values_to="value") %>%
ggplot(aes(variable, value, fill=variable)) +
geom_boxplot() +
labs(x='Variables', y='Values') +
theme(legend.position = "none",
axis.text.x = element_text(angle = 90, vjust = .5, hjust = 1))
```

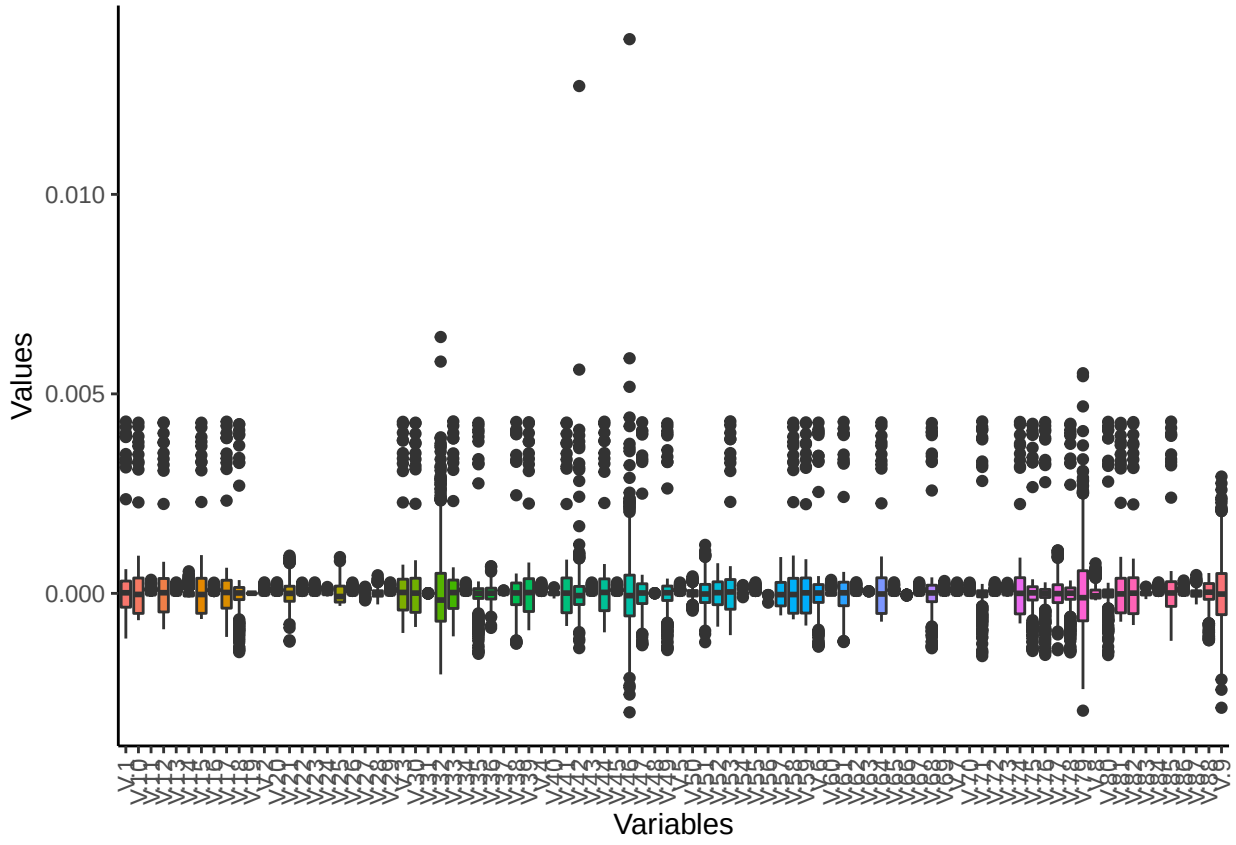

Figure 4: Boxplots of all variables in the HTP data set

Figure 4 reveals the presence of many outlying observations. The goal of the rest of the analysis is to determine whether the two defective parts **581** and **619** are among these potential outliers.

2.2 Part (b)

2.2.1 Estimates of the mean vector $\hat{\mu}$ and the VCOV matrix $\hat{\Sigma}$ with MCD

Obtaining estimates of the mean vector $\hat{\mu}$ and the variance-covariance (VCOV) matrix $\hat{\Sigma}$ of the data with MCD with a 25% breakdown point. Using the `covMcd()` function from the `robustbase` package with `cor` set to false, and `alpha` $= 0.75$ based on our breakdown point, we obtain the required estimates as follows. Given the many outputs, we chose to display the means of only the first 10 variables and their corresponding 10 by 10 submatrix from the estimated VCOV matrix.

```
library(robustbase)
# Obtain MCD estimates with a breakdown point of 25%
fit.robust <- covMcd(htp_dat, cor = F, alpha = 0.75)
mean_vector <- fit.robust$center # the final estimate of location
VCOV <- fit.robust$cov # the final estimate of scatter

# displaying results
data.frame(rbind(mean_vector[1:10])) %>%
    kable(caption = "Estimated mean vector for first 10 variables", booktabs=T)%>%
```

```
    kable_styling(latex_options = c("HOLD_position")) %>%
    kable_classic()
```

Table 3: Estimated mean vector for first 10 variables

V.1	V.2	V.3	V.4	V.5	V.6	V.7	V.8	V.9	V.10
1.54e-05	-3e-07	4.4e-06	-3e-07	-3e-07	2.89e-05	-3e-07	8.6e-06	-0.0001017	-1.94e-05

```
data.frame(VCOV[1:10, 1:10]) %>%
    kable(caption = "Estimated mean vector for first 10 variables", booktabs=T)%>%
    kable_styling(latex_options = c("HOLD_position")) %>%
    kable_classic()
```

Table 4: Estimated mean vector for first 10 variables

	V.1	V.2	V.3	V.4	V.5	V.6	V.7	V.8	V.9	V.10
V.1	2e-07	0	2e-07	0	0	1e-07	0	0e+00	1e-07	2e-07
V.2	0e+00	0	0e+00	0	0	0e+00	0	0e+00	0e+00	0e+00
V.3	2e-07	0	2e-07	0	0	1e-07	0	1e-07	2e-07	2e-07
V.4	0e+00	0	0e+00	0	0	0e+00	0	0e+00	0e+00	0e+00
V.5	0e+00	0	0e+00	0	0	0e+00	0	0e+00	0e+00	0e+00
V.6	1e-07	0	1e-07	0	0	1e-07	0	0e+00	1e-07	1e-07
V.7	0e+00	0	0e+00	0	0	0e+00	0	0e+00	0e+00	0e+00
V.8	0e+00	0	1e-07	0	0	0e+00	0	0e+00	0e+00	1e-07
V.9	1e-07	0	2e-07	0	0	1e-07	0	0e+00	6e-07	2e-07
V.10	2e-07	0	2e-07	0	0	1e-07	0	1e-07	2e-07	3e-07

We see that entries of the mean vector as well as the VCOV matrix are approximately 0. The actual data had very low values so this is not surprising (Please refer to Figure 4).

2.2.2 Plot of robust Mahalanobis distance of each observation with respect to the MCD estimates

We use the `mahalanobis()` function to compute the robust Mahalanobis diatance of each observation with respect to the MCD estimates of the mean vector and the VCOV matrix computed above. Figure 5 shows a plot of the resulting distances with an outlier threshold based on the $\chi^2_{p=88}$ distribution at 2.5% significance level. The two defective parts are highlighted in

```
# compare results to output from this plot function
# aq.plot(htp_dat, alpha=0.025) # pick one

# Compute Mahalanobis distance
robust_distances <- mahalanobis(htp_dat, center=mean_vector, cov=VCOV)
# fit.robust$mah contains the same values
```

```
# Cut-off based on the chi-square distribution
chisq_cutoff <- qchisq(0.975, df = ncol(htp_dat)) # at alpha = 0.025

# PLOT THE RESULTS
RD <- robust_distances
ids_most <- which(RD >= 22417)  # most outlying ids
id_most <- ids_most[which.max(RD[ids_most])]

par(mfrow=c(1,1), mar=rep(4,4))
colPoints <- ifelse(RD >= chisq_cutoff, 1, grey(0.5))
pchPoints <- ifelse(RD >= chisq_cutoff, 16, 4)
plot(seq_along(RD), RD, pch = pchPoints, col = colPoints,
    ylim=c(0, max(RD, chisq_cutoff) + 2), cex.axis = 0.7, cex.lab = 0.7,
    ylab = expression("(Mahalanobis Distance)"**2), xlab = "Observation Index")
abline(h = chisq_cutoff, lty = "dashed", col="red")
legend("topleft", lty = "dashed", cex = 0.7, ncol = 2, bty = "n",
    legend = expression(paste(chi[p]**2, " cut-off")), col="red")
text(619, RD[619], labels=619, col="blue")
text(581, RD[581], labels=581, col="blue")
text(id_most, RD[id_most], labels=id_most, col=id_most)
```

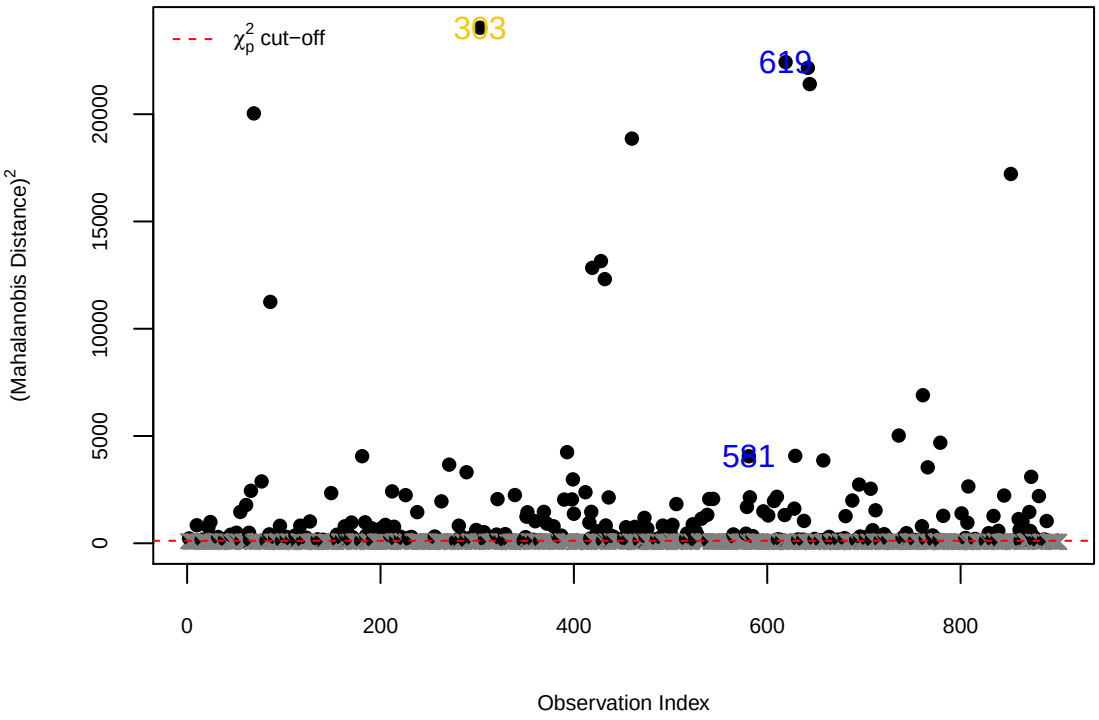

Figure 5: Plot of robust Mahalanobis distance of each observation with respect to the MCD estimates of location and scatter

Observations above or equal to the $\chi^2_{p=88} = 115.84$ represented by the red dashed line are considered potential outliers. Obviously, the two defective parts show up above the cut-off line among the

top list of potential outliers. However, they are not the most outlying observations since the 303^{th} observation can be seen as the most outlying.

2.3 Part (c)

2.3.1 Defining anomaly score plotting function

For convenience, we packaged the R codes for plotting the anomaly scores into a function called `plot_anomaly_score()`. By default if the argument `threshold` is NULL, the 99^{th} quantile of the anomaly scores supplied is computed as the cut-off or threshold for detecting potential outliers, otherwise whatever the user supplied is used.

```
plot_anomaly_score <- function (score, threshold=NULL, anomaly_id=NULL,pch=1,
                                xlab="Observation index", ylab="Anomaly score",
                                main="", seg_col="#7AD151FF", label_adj=0.03,
                                 cex_adj=4,
                                label_col="deepskyblue2", threshold_col="red") {

    # set threshold/cutoff point
    if (is.null(threshold)) threshold <- quantile(score, 0.99)

    # get anomaly ids
    if (is.null(anomaly_id)) anomaly_id <- which(score > threshold)

    plot(seq_along(score), score, type="p", pch=pch, main=main, xlab=xlab,
         ylab=ylab, cex=score*cex_adj, col="coral2")
    # add line segments
    add.seg <- function(x) segments(x0=x[1], y0=0, x1=x[1], y1=x[2],
                                lty=1, lwd=1.5, col="#7AD151FF")
    apply(data.frame(id=1:length(score), score=score), 1, FUN=add.seg)
    # add indices as labels
    text(anomaly_id, score[anomaly_id]+label_adj, label=anomaly_id,
     col=label_col, cex=0.7)
    # add a threshold line
    abline(h = threshold, lty = "dashed", col=threshold_col)
}
```

2.3.2 Isolation Forest (iForest)

To apply the isolation forest anomaly detection algorithm on our data, we used `IsolationTrees()` function from the `IsolationForest` package available for Mac users. 100 fully deterministic isolation trees were built by setting `ntree` to 100 and `rForest` to 0. All other parameters were left at their default values.

```
#Isolation Forest
library(IsolationForest)
```

```
## IsolationForest 0.0-26
```

```
# model specification
iso_tree <- IsolationTrees(htp_dat, rFactor=0, ntree = 100)
# get anomaly scores
anomaly_score <- AnomalyScore(htp_dat, iso_tree)
iso_scores <- anomaly_score$outF

# plot anomaly scores
plot_anomaly_score(iso_scores, main = "Anomaly detection via iForest",
                   label_adj = 0.003, threshold_col = "#414487FF")
```

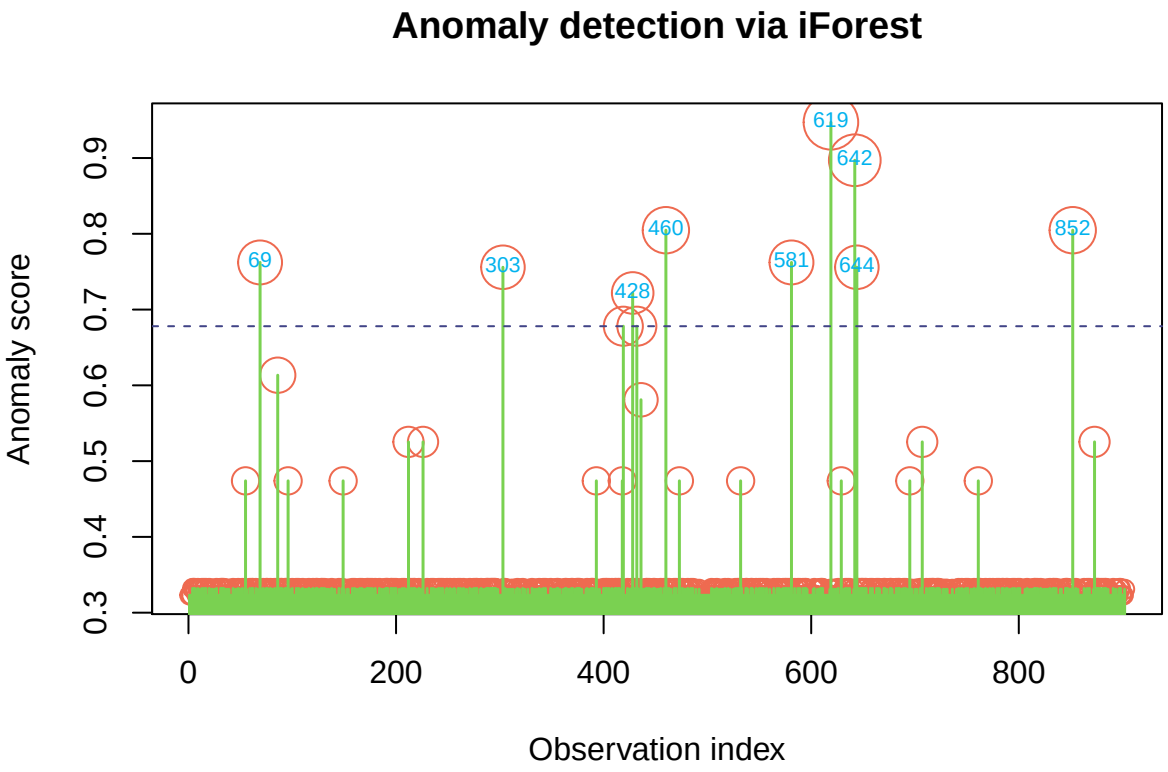

Figure 6: Anomaly detection via isolation forest (iForest)

An observation is considered a potential outlier if its anomaly score is close to 1. Thus, to detect the outlying observations, we used the 99^{th} quantile of the scores as a threshold indicated by the dashed horizontal line. The indices of the observations identified as potential outliers are labeled on the plot.

From the plot, the two defective parts 581 and 619 are considered outliers by the iForest anomaly detection method.

2.3.3 Local Outlier Factor (LOF)

We perform another anomaly detection using LOF with the help of the `lof()` function from the `Rlof` package. The k^{th} distance parameter was set to **8**. Figure 7 shows the result obtained.

```
library(Rlof)
# #2A788EFF
anomaly_scores <- lof(htp_dat, k=8)
quant_cutoff <- quantile(anomaly_scores, 0.99)
anomaly_id <- which(anomaly_scores > quant_cutoff)
lof_scores <- scale(anomaly_scores, center = min(anomaly_scores),
                    scale = max(anomaly_scores)-min(anomaly_scores))
# plot the anomaly scores
plot_anomaly_score(lof_scores, label_adj = 0.02, cex_adj = 5.5,
                   threshold_col = "#414487FF",main = "Anomaly detection via LOF")
```

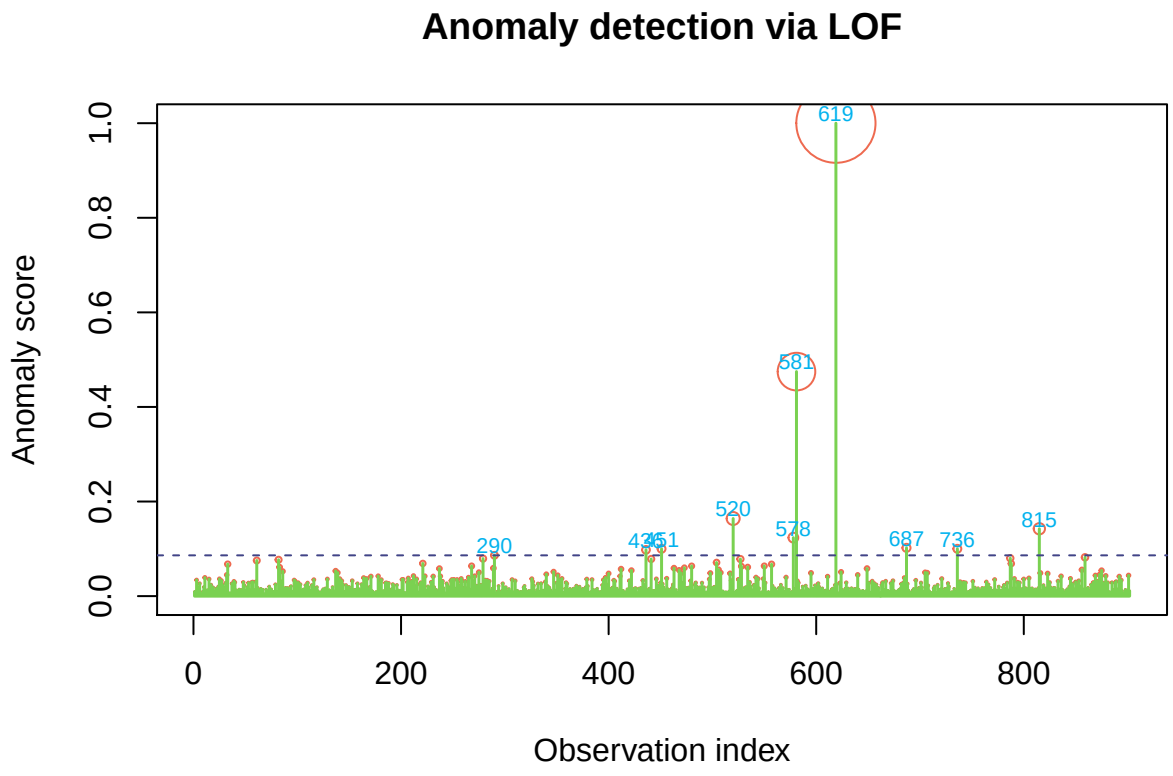

Figure 7: Anomaly detection via Local Outlier Factor (LOF)

From the graph, we observe that with a 99% quantile threshold, LOF clearly detected the two defective parts 581 and 619 as anomalies among its list of potential outliers.

2.3.4 Comparing results from iForest and LOF

```
par(mfrow=c(1,2))
# plot anomaly scores for iForest:
plot_anomaly_score(iso_scores, main = "Anomaly detection via iForest",
                   label_adj = 0.003, threshold_col = "#414487FF")

# plot the anomaly scores for lof:
```

```
plot_anomaly_score(lof_scores, label_adj = 0.02, cex_adj = 5.5,
                   threshold_col = "#414487FF", main = "Anomaly detection via LOF")
```

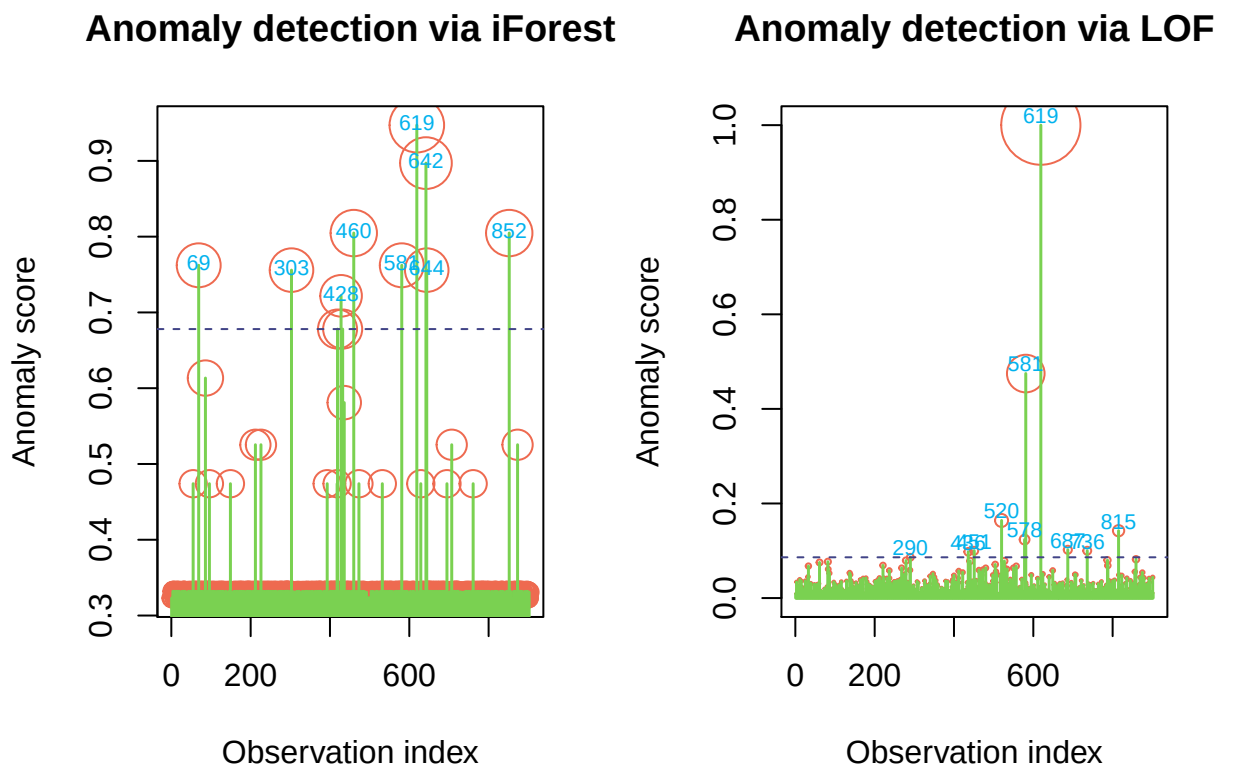

Figure 8: Comparing results from iForest and LOF

We observe from Figure 8 that, with the same 99% quantile threshold, both iForest and LOF succeeded in detecting the two defective parts as outliers, since the two observations **581** and **619** are part of the list of potential outliers for the two methods. Thus, these two anomaly detection methods suggest that the two defective parts are indeed anomalies. However, for this anomaly detection problem, we can see that LOF performed better than iForest since the LOF method clearly singled the two defective parts out from the other potential outliers, leaving no doubt that those two observations are indeed anomalies. It appears that the two defective parts are the only common outliers detected by the two methods.