MACHINE LEARNING ANOMALY DETECTION METHODS

KELVIN OFORI-MINTA

Contents

1 Introduction

This project seeks to compare four(4) different methods of machine learning models for outlier detection with the use of HTP data sets in R.
Essentially we seek to find outliers in this data. This dataset contains 902 high-tech parts(HTP) designed for consumer products characterized by 88 tests. These tests are performed to ensure a high quality of the production.
All these 902 parts were considered functional and have been sold. However the two parts **581** and **619** showed defects in practical use and were returned to the manufacturer by the customer.Therefore these two can be considered as outliers.
We want to deploy different machine learning models with this data to confirm the authenticity of the customer's complaint to manager. MCD, ISOFOREST, LOF & One Class SVM

2 Anomaly Detection Machine learning Models

2.1 Bring in Data

```
#install.packages("ICSOutlier")
suppressPackageStartupMessages(library("ICSOutlier"))
#library("ICSOutlier")
data(HTP)
dat <- HTP; dim(dat); #labelled data: 88 features & 902 observations
```

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

```
outliers.true <- c(581, 619) # index of two defective products returned by customer.
```

This dataset contains 902 high-tech parts(HTP) designed for consumer products characterized by 88 tests. These tests are performed to ensure a high quality of the production.
All these 902 parts were considered functional and have been sold. However the two parts **581** and **619** showed defects in practical use and were returned to the manufacturer by the customer.Therefore these two can be considered as outliers.

3 Method 1: Minimum Covariance Determinant(MCD)

3.1 Robust Estimates of Mean Vector, VCOV Matrix with MCD

```
#First obtain robust estimates of the mean vector  and
#VCOV matrix of the data with MCD with a breakdown point of your choice
#install.packages("robustbase")
library(robustbase)
# Obtain MCD estimates with a breakdown point of 30%
fit.robust <- covMcd(dat, cor = FALSE, alpha = 0.70)
'fit.robust$center' #robust estimates of mean vector
```

```
## [1] "fit.robust$center"
```

```
'fit.robust$COV' #robust estimates of variance-covariance matrix
```

```
## [1] "fit.robust$COV"
```

3.2 Compute the Robust Mahalanobis Distance of each Observation

```
#Compute the robust Mahalanobis distance of each observation with respect to the MCD e
RD <- mahalanobis(dat, fit.robust$center, fit.robust$cov)
RD[1:30]
```

```
##  [1]   98.73034  226.60001   93.64501   79.31421   76.06548   79.56195
##  [7]  107.30439   90.65159   59.87810  864.23648   91.71575   76.99800
## [13]   99.51342   65.08976  172.96194   58.38593  104.26704   66.57062
## [19]  121.67104  104.75973   70.91819  822.24586   66.18853 1020.93879
## [25]   55.90454   61.11124  102.16523   94.15672   88.93099  113.39450
```

```
# Cut-off based on the chi-square distribution
cutoff.chi.sq <- qchisq(0.975, df = ncol(dat)); cutoff.chi.sq
```

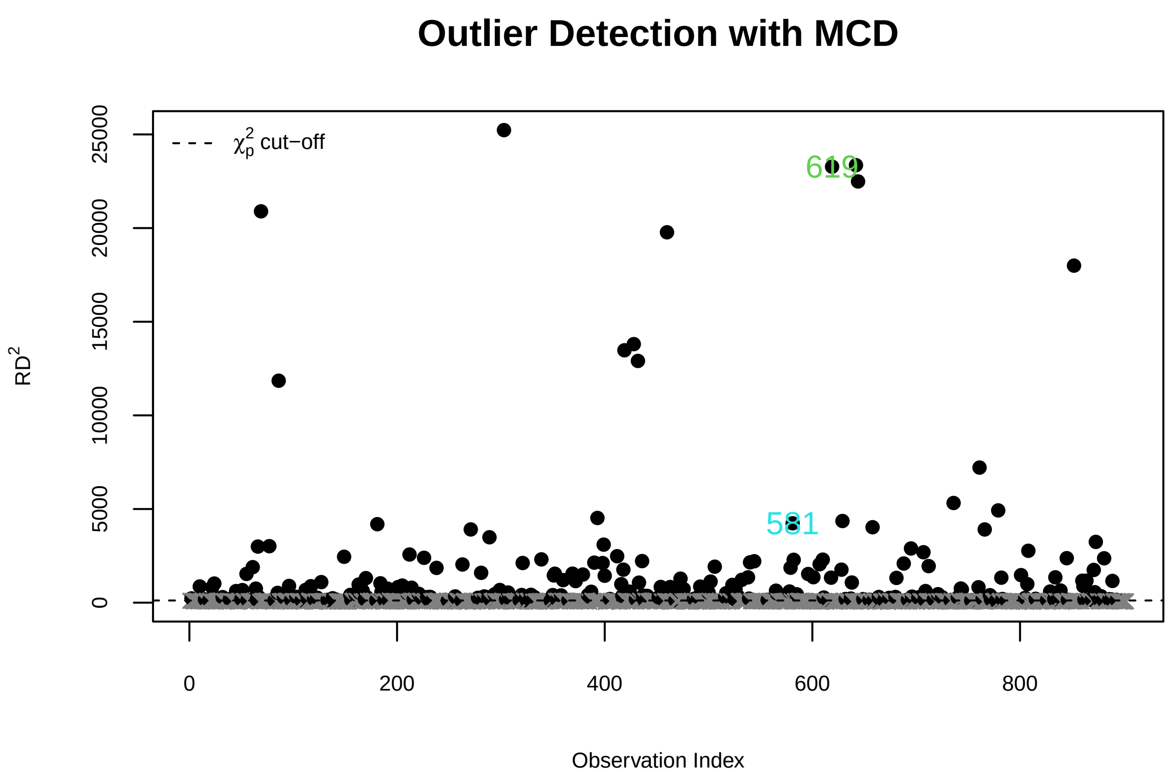
```
## [1] 115.8414
```

```
#All observations above this cutoff value are potential outliers.
```

All observations above this cutoff value are potential outliers.

3.3 Plot Results of MCD

```
# PLOT THE RESULTS
colPoints <- ifelse(RD >= min(c(cutoff.chi.sq)), 1, grey(0.5))
pchPoints <- ifelse(RD >= min(c(cutoff.chi.sq)), 16, 4)
plot(seq_along(RD), RD, pch = pchPoints, col = colPoints,
    ylim=c(0, max(RD, cutoff.chi.sq) + 2), cex.axis = 0.7, cex.lab = 0.7,
    ylab = expression(RD**2), xlab = "Observation Index", main="Outlier Detection with M
abline(h = c(cutoff.chi.sq), lty = c("dashed", "dotted"),      ) # col=c("blue", "red")
legend("topleft", lty = c("dashed", "dotted"), cex = 0.7, ncol = 2, bty = "n",
    legend = c(expression(paste(chi[p]**2, " cut-off"))),     ) # col=c("blue", "red")
text(619, RD[619], labels = 619, col=619)
text(581, RD[581], labels = 581, col=581)
```




```
which(RD >= cutoff.chi.sq) #outlier IDS - ALL POTENTIAL OUTLIERS
```

```
##   [1]   2  10  15  19  22  24  32  39  41  45  50  51  55  56  61  64  65  66
##  [19]  67  69  77  82  83  85  86  91  96  98 103 108 112 113 114 117 120 123
##  [37] 127 133 135 138 139 140 141 149 155 156 160 163 164 165 167 169 170 171
##  [55] 180 181 184 185 188 191 192 201 205 210 212 214 216 221 224 226 229 230
##  [73] 231 232 234 238 249 256 257 262 263 271 278 280 281 284 288 289 290 294
##  [91] 299 303 305 307 308 310 318 320 321 324 325 328 329 330 332 339 348 350
```

```
## [109] 351 352 354 358 360 365 369 372 379 384 386 387 390 393 398 399 400 405
## [127] 412 416 417 418 419 424 428 430 432 433 436 437 438 441 452 453 454 456
## [145] 457 460 463 468 472 473 474 476 486 490 492 500 502 506 516 517 520 523
## [163] 524 525 526 527 528 532 538 539 540 544 557 565 566 578 579 581 582 585
## [181] 596 601 607 610 611 615 618 619 628 629 632 637 638 642 644 649 655 658
## [199] 659 664 665 669 670 674 680 681 688 692 695 696 702 703 707 708 709 712
## [217] 717 718 721 725 733 736 740 743 744 753 760 761 764 766 771 772 773 777
## [235] 778 779 782 783 787 801 805 807 808 815 826 829 833 834 836 838 839 845
## [253] 846 852 860 861 862 864 865 869 871 872 873 874 876 878 880 881 886 888
## [271] 889 893
```

```
#inspect the most outlying observation
#Top list of potential outliers would actually be above an RD(Mahalanobis value of 225
most.outly<- which(RD > 22500)
most.outly
```

```
## [1] 303 619 642
```

Top list of potential outliers are shown above,(based on a mahalanobis distance-RD > 22500). This clearly indicates that only item 619 is among the top list of outliers, this also informs that item 581 is not captured in the “top list” of potential outliers, contrary to customer’s claim. Thus the MCD model was able to find only 1 defective item.

4 Method 2: Isolation Forest

4.1 Deploy Isoforest with isofor package

```
#ISOLATION FOREST
suppressPackageStartupMessages(library("devtools"))

#devtools::install_github("Zelazny7/isofor")
suppressPackageStartupMessages(library(isofor))

# help(package="isofor")

# An isolation forest model with 200 trees and 256 samples drawn  to construct each tr
fit.isoforest <- iForest(dat, nt=200, phi=256)
pred <- predict(fit.isoforest, newdata=dat)#pred
```

4.2 Visualizing Isolation Forest Results

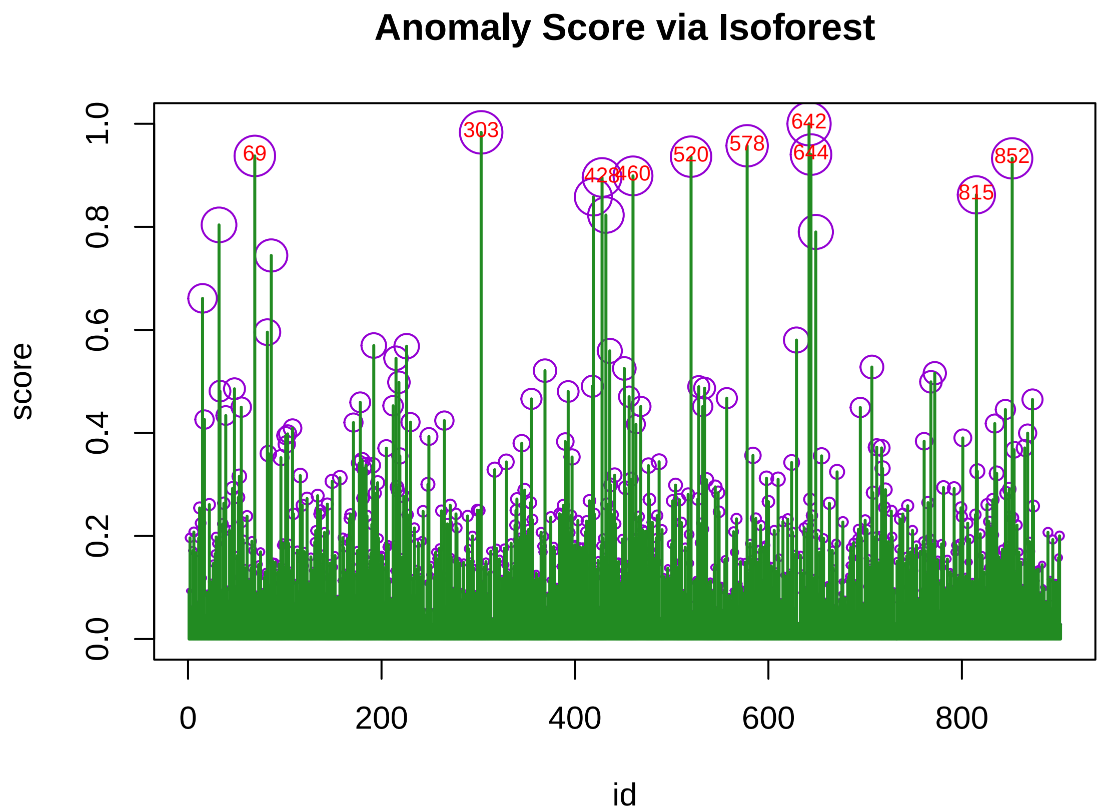
```
# PLOT OF THE SCORES
score <- scale(pred, center = min(pred), scale = max(pred)-min(pred))
par(mfrow=c(1,1), mar=rep(4,4))
plot(x=1:length(score), score, type="p", pch=1,
    main="Anomaly Score via Isoforest",
        xlab="id", ylab="score", cex=score*3, col="darkviolet")
add.seg <- function(x) segments(x0=x[1], y0=0, x1=x[1], y1=x[2],
    lty=1, lwd=1.5, col="forestgreen")
apply(data.frame(id=1:length(score), score=score), 1, FUN=add.seg)
```

```
## NULL
```

```
eps <- 0.99
id.outliers <- which(score > quantile(score, eps));id.outliers
```

```
##  [1]  69 303 428 460 520 578 642 644 815 852
```

```
text(id.outliers, score[id.outliers]+0.005, label=id.outliers,
    col="red", cex=0.7)
```



This isolation forest plot shown above could not specifically identify those two observations(defective items) as anomalies.

5 Method 3: Local Outlier Factor (LOF)

5.1 Deploy LOF with Rlof package

```
# LOCAL OUTLIER FACTOR
#install.packages("Rlof")
#help(package="Rlof")
library(Rlof)
```

```
## Loading required package: doParallel

## Loading required package: foreach

## Loading required package: iterators

## Loading required package: parallel
```

```
#Obtained the LOF of our dataset with k-nearest neighbour, which considers the density
outlier.scores <- lof(dat, k=10);  outlier.scores [1:20]
```

```
##  [1] 0.9995510 1.0067830 1.1478922 0.9928840 1.0782530 1.1618730 0.9703249
##  [8] 0.9809807 1.0052978 1.0461865 1.2271454 1.0440390 0.9797851 1.0367347
## [15] 1.1860166 1.0577649 1.1354620 1.0102131 1.0652250 0.9717027
```

```
which(outlier.scores > quantile(outlier.scores, 0.95))
```

```
##  [1]  33  61  69  82  83  86 137 221 237 268 279 289 290 303 347 419 428 432 436
## [20] 441 451 460 463 468 473 480 504 520 527 528 534 550 557 578 581 619 642 644
## [39] 649 687 736 787 788 815 852 859
```

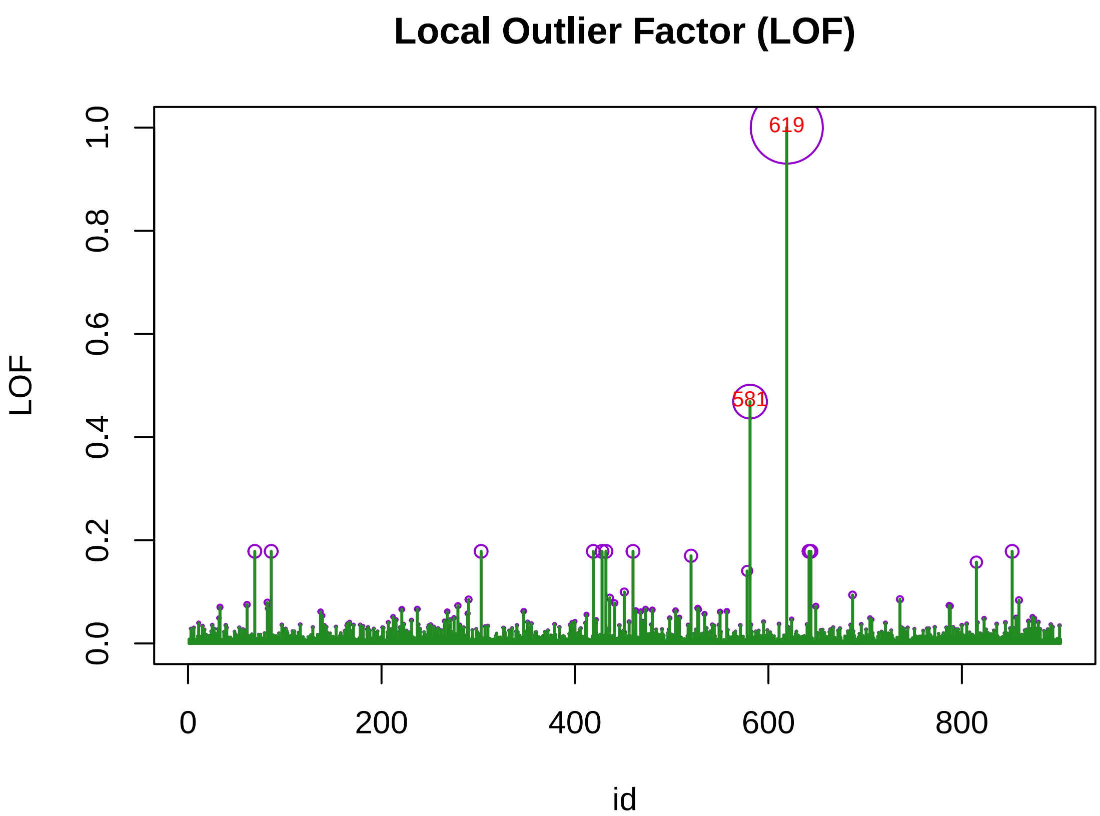
5.2 Visualize LOF Results

```
# PLOT OF THE LOF SCORES
score <- scale(outlier.scores, center = min(outlier.scores),
    scale = max(outlier.scores)-min(outlier.scores)) # NORMALIZED TO RANGE[0,1]
par(mfrow=c(1,1), mar=rep(4,4))
plot(x=1:length(score), score, type="p", pch=1,
    main="Local Outlier Factor (LOF)",
        xlab="id", ylab="LOF", cex=score*5, col="darkviolet")
add.seg <- function(x) segments(x0=x[1], y0=0, x1=x[1], y1=x[2],
    lty=1, lwd=1.5, col="forestgreen")
apply(data.frame(id=1:length(score), score=score), 1, FUN=add.seg)
```

```
## NULL
```

```
eps <- 0.99
id.outliers <- which(outlier.scores > quantile(outlier.scores, eps))
text(id.outliers, score[id.outliers]+0.005, label=id.outliers,
```

```
    col="red", cex=0.7)
```



This LOF plot above clearly brings out the two defective items(619 & 581) as observed anomalies.

FINDINGS: From the Anomaly Detection plots above: The graph of Local Outlier Factor(LOF) clearly depicts the outliers better than the Isolation forest graph.