Intrusion Detection using NASA HTTP Logs

AHMAD ARIDA
DA CHEN
Presentation Overview

- Background
- Preprocessing
- Data Mining Methods to Determine Outliers
- Finding Outliers
- Outlier Validation
- Summary
What is an Outlier?

Definition:

- an outlier is an observation point that is distant from other observations
- Could occur randomly in any distribution
  - Measurement error
  - Irregular distribution
- Point of reference:
  - In normally distributed data, 1 in every 22 observations will differ by 2x the standard deviation (or more) from the mean
  - *** 1 in 370 will deviation by 3x
NASA HTTP Logs

- Collection of all HTTP requests to the NASA Kennedy Space Center server in Florida
- Collected from July 1\textsuperscript{st} 1995 – July 31\textsuperscript{st} 1995

**The Good:**

- (Potentially) “clean” log files

**The Bad:**

- Total of ~1.9 million access requests
Data Collection

Note:
- ~1.9 million rows of data is beyond what Excel can handle
Pre-Processing of the Log File

```r
# importing the access logs from text file
fileName <- 'C:/Users/osuki/Desktop/access_log_Jul95.txt'

# these were two packages that needed to be installed
# in order to use the str_split feature
# install.packages("stringr")
# install.packages("dplyr")

# referencing the package libraries
library(stringr)
library(dplyr)

# opening the file
conn <- file(fileName, open="r")
# reading each line from the file
linn <- readLines(conn)

# removing special characters
lines <- gsub("\\[\\]", "", linn)
lines2 <- gsub("\\", "", lines)

# splitting the string into a data frame
df <- str_split(lines2, " ") %>% do.call(rbind, .) %>% as.data.frame()

# selectively picking order and columns for output
final <- df[c(1, 4, 5, 6, 7, 8, 9, 10)]

# labeling the columns
colnames(final) <- c("Host", "TimeStamp", "TimeZone", "Command", "RequestLink", "HTTP", "ReplyCode", "Bytes")

# writing the output file
write.csv(final, 'C:/Users/osuki/Desktop/output9.csv')

# closing the original file connection
close(conn)
```
The CSV File....

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Host</td>
<td>TimeStamp</td>
<td>TimeZone</td>
<td>Command</td>
<td>RequestLink</td>
<td>HTTP</td>
<td>ReplyCode</td>
<td>Bytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>199.72.81.55</td>
<td>01/Jul/1995:00:01</td>
<td>-400</td>
<td>GET</td>
<td>/history/spella/</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>6245</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>unicom6.unicomp.net</td>
<td>01/Jul/1995:00:06</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/countdown/</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>3985</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>199.120.110.21</td>
<td>01/Jul/1995:00:09</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/missions/sts-73/mission-sts-73.html</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>4083</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>burger.letters.com</td>
<td>01/Jul/1995:00:11</td>
<td>-400</td>
<td>GET</td>
<td>/images/NASA-logo.png</td>
<td>HTTP/1.0</td>
<td>304</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7burger.letters.com</td>
<td>01/Jul/1995:00:12</td>
<td>-400</td>
<td>GET</td>
<td>/images/NASA-logo.png</td>
<td>HTTP/1.0</td>
<td>304</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8205.212.113.100</td>
<td>01/Jul/1995:00:12</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/countdown/countdown.html</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>3983</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>5d104.aa.net</td>
<td>01/Jul/1995:00:13</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/countdown/</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>3985</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10.129.94.144.152</td>
<td>01/Jul/1995:00:13</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/countdown/</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>7074</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>unicom6.unicomp.net</td>
<td>01/Jul/1995:00:14</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/countdown/countdown.html</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>46310</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>unicom6.unicomp.net</td>
<td>01/Jul/1995:00:14</td>
<td>-400</td>
<td>GET</td>
<td>/images/KSC-logosmall.gif</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>786</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>unicom6.unicomp.net</td>
<td>01/Jul/1995:00:14</td>
<td>-400</td>
<td>GET</td>
<td>/images/KSC-logosmall.gif</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>1204</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>d104.aa.net</td>
<td>01/Jul/1995:00:15</td>
<td>-400</td>
<td>GET</td>
<td>/images/KSC-logosmall.gif</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>46310</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>d104.aa.net</td>
<td>01/Jul/1995:00:15</td>
<td>-400</td>
<td>GET</td>
<td>/images/KSC-logosmall.gif</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>786</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>d104.aa.net</td>
<td>01/Jul/1995:00:15</td>
<td>-400</td>
<td>GET</td>
<td>/images/KSC-logosmall.gif</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>1204</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>17.129.94.144.152</td>
<td>01/Jul/1995:00:17</td>
<td>-400</td>
<td>GET</td>
<td>/images/KSC-logosmall.gif</td>
<td>HTTP/1.0</td>
<td>304</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>18.199.120.110.21</td>
<td>01/Jul/1995:00:17</td>
<td>-400</td>
<td>GET</td>
<td>/images/KSC-logosmall.gif</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>1212</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>ppptty291.ash.sh漠or.jp</td>
<td>01/Jul/1995:00:18</td>
<td>-400</td>
<td>GET</td>
<td>/facts/about_ksc.html</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>2977</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>net-1.141.eden.com</td>
<td>01/Jul/1995:00:19</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/missions/sts-71/images/KSC-95EC1-0018.jpg</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>34029</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>ppptty291.ash.sh漠or.jp</td>
<td>01/Jul/1995:00:19</td>
<td>-400</td>
<td>GET</td>
<td>/images/launchpalms-small.gif</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>11473</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>22.205.189.134.54</td>
<td>01/Jul/1995:00:24</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/countdown/</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>3985</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>waters.gu.stryx.yr.net.au</td>
<td>01/Jul/1995:00:25</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/missions/51.1/mission-51.1.html</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>6723</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>ppp-pia-80-shadow.net</td>
<td>01/Jul/1995:00:27</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/missions/51.1/mission-51.1.html</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>7074</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>25.205.189.134.54</td>
<td>01/Jul/1995:00:29</td>
<td>-400</td>
<td>GET</td>
<td>/shuttle/countdown/countdown.html</td>
<td>HTTP/1.0</td>
<td>200</td>
<td>46310</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
How to Start Looking at 1.9 million Rows?

- We have no real idea of the scope of our data
- Random outliers in our sample set:
  - 2x mean - ~86,000
  - 3x mean - ~5,000
- Unsupervised Cluster Method
  - Cluster Plot
First Attempt...
Use all rows and see if any patterns emerge

CLUSPLOT (plot)

These two components explain 29.29% of the point variability.
Second Attempt...
Use all rows and see if any patterns emerge

These two components explain 29.27% of the point variability.
Now What?

- Cluster Plot helped to give us an idea of the general spread of the data
- Need to figure out if we can eliminate any rows
- Discretize some data
- Gain some statistical data
SQL Server to the Rescue

![Success message from SQL Server Import and Export Wizard](image)

- **Action**: Initializing Data Flow Task, Initializing Connections, Setting SQL Command, Setting Source Connection, Setting Destination Connection, Validating, Prepare for Execute, Pre-execute, Executing, Copying to "dbo"."NASALogs", Post-execute
- **Status**: Success
- **Total**: 11
- **Success**: 11
- **Error**: 0
- **Warning**: 0

Message: 139,715 rows transferred
Learning about our data

<table>
<thead>
<tr>
<th>Host</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>piweba3y.prodigy.com</td>
<td>17572</td>
</tr>
<tr>
<td>piweba4y.prodigy.com</td>
<td>11591</td>
</tr>
<tr>
<td>piweba1y.prodigy.com</td>
<td>9868</td>
</tr>
<tr>
<td>alyssa.prodigy.com</td>
<td>7852</td>
</tr>
<tr>
<td>siltb10.orl.mmc.com</td>
<td>7573</td>
</tr>
<tr>
<td>piweba2y.prodigy.com</td>
<td>5922</td>
</tr>
<tr>
<td>edams.ksc.nasa.gov</td>
<td>5434</td>
</tr>
<tr>
<td>163.206.89.4</td>
<td>4906</td>
</tr>
<tr>
<td>news.ti.com</td>
<td>4863</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Host</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>p19.t0.halsey.com</td>
<td>1</td>
</tr>
<tr>
<td>p131.ac.duke.edu</td>
<td>1</td>
</tr>
<tr>
<td>p18-14.dialup.uvic.ca</td>
<td>1</td>
</tr>
<tr>
<td>p211.cc.uch.gr</td>
<td>1</td>
</tr>
<tr>
<td>pandora.physics.ox.ac.uk</td>
<td>1</td>
</tr>
<tr>
<td>panther1297.eiu.edu</td>
<td>1</td>
</tr>
<tr>
<td>panther1657.eiu.edu</td>
<td>1</td>
</tr>
<tr>
<td>p6mac3.lanl.gov</td>
<td>1</td>
</tr>
</tbody>
</table>
Learning about our data

Number of Accesses By Host

SELECT avg(count) as 'Average', STDDEV(count) as 'Standard Deviation'
FROM
(SELECT COUNT (n.Host) AS Count
FROM dbo.NASALogs n
GROUP BY n.Host
) as counts
Learning about our data

```
SELECT avg(count) as 'Average', STDEV(Count) as 'Standard Deviation'
FROM
  (SELECT COUNT (n.bytes) AS Count
   FROM dbo.NASALogs n
   GROUP BY n.bytes
  ) as counts
```

<table>
<thead>
<tr>
<th>Average</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>142</td>
<td>2150.78114683868</td>
</tr>
</tbody>
</table>
Learning about our data
Discretization
Cluster Plot 3.0

These two components explain 42.83% of the point variability.
These two components explain 77.71% of the point variability.
Cluster Plot 3.1

**Group 1**
134.57.9.77, POST, /ksc.html, 501,-
ramsay.ann-arbor.mi.us, HEAD, /shuttle/technology/sts-newsref/sts-lcc.html#sts-countdown, 404,-

**Group 2**
This point was near the middle of the screen and likely associated with the “POST” grouping:
newcastle03.nbnet.nb.ca, POST, /ksc.html, 501,-

**Group 3**
The one in the middle of the screen is associated with a byte amount of 7634:
ajohnson.ssc.nasa.gov, GET, /shuttle/missions/sts-71/images/images.html, 200, 7634
It’s Wabbit Season...

- Now that we have some potential outliers, what do they have in common?

- Are they real outliers?

- Are they intrusion attempts?
Common / Uncommon Traits

SQL Query:

```sql
Select n.command, Count(n.command) as 'Count'
From dbo.NASALogs3 n
where bytes like '%-
Group by n.command
```

<table>
<thead>
<tr>
<th>command</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>19506</td>
</tr>
<tr>
<td>HEAD</td>
<td>4</td>
</tr>
<tr>
<td>POST</td>
<td>19</td>
</tr>
</tbody>
</table>

SQL Query:

```sql
Select n.replycode, Count(n.replycode) as 'Count'
From dbo.NASALogs3 n
where bytes like '%-
Group by n.replycode
order by Count desc
```

<table>
<thead>
<tr>
<th>replycode</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>404</td>
<td>10673</td>
</tr>
<tr>
<td>302</td>
<td>8705</td>
</tr>
<tr>
<td>200</td>
<td>83</td>
</tr>
<tr>
<td>403</td>
<td>54</td>
</tr>
<tr>
<td>501</td>
<td>14</td>
</tr>
</tbody>
</table>
Common / Uncommon Traits
Common / Uncommon Traits

- Looked very promising, but didn’t help predict an outlier
- Requested links with bytes of ‘-’
- Bytes of ‘-’ with Reply Code of 501
Mahalanobis Distance

- Gather information about multi-dimensional datasets that measures the standard deviations from the mean of the distribution of the points.
Mahalanobis Distance

```r
# Ahmad Arif and Da Chen
# CIS560 NASA HTTP Log Project

filename <- 'c:/users/osuki/desktop/Mahalanobis.txt'

library(stringr)
library(dplyr)

opening the file
conn <- file(filename, open="r")
# reading each line from the file
linn <- readLines(conn)

# splitting the string into a data frame
df <- str_split(linn, '\t') %>% do.call(rbind, .) %>% as.data.frame()

> head(df)
  V1 V2 V3
1 3755 200 3258
2 3755 200 650
3 3822 200 5866
4 4755 200 3047
5  8 200 12451
6  5 200 3258

> df2 <- as.matrix(as.data.frame(lapply(df, as.numeric)))
> mean <- colMeans(df2, na.rm=TRUE)
> sx <- cov(df2)
> d2 <- mahalanobis(df2, mean, sx)
> d3 <- density(d2, bw=0.5)
> plot(d3, main="Squared Mahalanobis distances")
> write.csv(d3, 'c:/users/osuki/desktop/Mahalanobisoutput', row.names = F)
```

Squared Mahalanobis distances

N = 1891714  Bandwidth = 0.5
Identify Mahalanobis Outliers

<table>
<thead>
<tr>
<th>Mahalanobis Distance</th>
<th>Count</th>
<th>% of Total (1891716)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10+</td>
<td>33250</td>
<td>1.758</td>
</tr>
<tr>
<td>20+</td>
<td>14651</td>
<td>0.774</td>
</tr>
<tr>
<td>25+</td>
<td>8989</td>
<td>0.475</td>
</tr>
<tr>
<td>50+</td>
<td>2639</td>
<td>0.140</td>
</tr>
<tr>
<td>100+</td>
<td>1798</td>
<td>0.095</td>
</tr>
<tr>
<td>150+</td>
<td>1441</td>
<td>0.076</td>
</tr>
<tr>
<td>173+</td>
<td>46</td>
<td>0.002</td>
</tr>
</tbody>
</table>
Re-Clean Data...

- The largest deviations turned out to be due to null values that were left from our original pre-processing steps
- Re-pre-process
- Try cluster plot analysis again
Re-Clean Data...

- The largest deviations turned out to be due to null values that were left from our original pre-processing steps
- Re-pre-process
- Try cluster plot analysis again
Cluster Plot 4.0

These two components explain 58.77% of the point variability.
Mahalanobis 2.0

Squared Mahalanobis distances

N = 1849546  Bandwidth = 0.5
## Common Features!!

<table>
<thead>
<tr>
<th>Host</th>
<th>TimeStamp</th>
<th>TimeZone</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>tia1.eskimo.com</td>
<td>20/Jul/1995:08:29:45</td>
<td>-400</td>
<td>HEAD</td>
</tr>
<tr>
<td>slip165-175.on.ca.ibm.net</td>
<td>16/Jul/1995:03:07:18</td>
<td>-400</td>
<td>POST</td>
</tr>
<tr>
<td>ramsay.ann-arbor.mi.us</td>
<td>06/Jul/1995:02:29:58</td>
<td>-400</td>
<td>HEAD</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RequestLink</th>
<th>HTTP</th>
<th>ReplyCode</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>/robots.txt</td>
<td>HTTP/1.0</td>
<td>404</td>
<td>-</td>
</tr>
<tr>
<td>/cgi-bin/WebQuery</td>
<td>HTTP/1.0</td>
<td>404</td>
<td>-</td>
</tr>
<tr>
<td>/shuttle/technology/sts-newsref/sts-lcc.html#sts-countdown</td>
<td>HTTP/1.0</td>
<td>404</td>
<td>-</td>
</tr>
</tbody>
</table>
Finding Outliers

```sql
/* Script for SelectTopNRows command from SSMS */
SELECT *
FROM [CIS660-Project].[dbo].[NASALogs3]
WHERE ReplyCode like '404'
AND bytes like ':-'
AND (command = 'POST' OR command = 'HEAD')
```
Are We Sure?

```
/* Script for SelectTopNRows command from SMS */

SELECT * 
FROM [CIS661-Project].[dbo].[NASALogs3]
where
  --ReplyCode like '404'
  --and bytes like '-'
  --and
RequestLink = '/cgi-bin/WebQuery'
```
Likely Intrusion Attempts

- **404 reply**
  - Tracked by security sites and can reveal hacking attempts

- **POST**
  - Command to a web server to accept and store data in a message
  - Is used in malicious attempts to embed information within a server without authorization

- **HEAD**
  - Command to retrieve representation of a specific source
  - Used to retrieve meta-data
Outliers are points in a data set that lie far away from the estimated value of the center of the data set. This estimated center could be either the mean, or median, depending on what kind of point or interval estimate you’re using.

Put a good picture here
Chi-Squared Test

The **chi-squared test of independence** is one of the most basic tests in the statistical analysis. When you are given 2 categorical random variables, the chi-squared test of independence determines whether or not there exists a statistical dependence.
Grubb’s Test

```
> library(outliers)
> d <- read.csv("C:/Users/Ahead/desktop/output9.csv")
> bytes <- d$bytes
> ttest <- chisq.test(bytes, var.test = var.dbytes)
> print(ttest)

> chi-squared test for outlier

data: d$bytes
X-squared = 0.1972, p-value = 0.9993
alternative hypothesis: highest value 13816 is an outlier

> for opposite
> ttest2 <- chisq.test(bytes, var.test = var.dbytes, opposite = TRUE)
> print(ttest2)

> chi-squared test for outlier

data: d$bytes
X-squared = 1.1864, p-value = 0.28
alternative hypothesis: lowest value 1 is an outlier
```
Grubb’s test can be used to determine whether or not a single outlying value within a set of measurements varies sufficiently from the mean value that it can be statistically classified as not belonging to the same population.
How Good Is Our Search For Intrusion Attempts?

- NASA has a 2\textsuperscript{nd} set of HTTP logs from Aug 1\textsuperscript{st} to Aug 31\textsuperscript{st} 1995
- This data has not been through any of the above tests / steps
- See if we can directly find our intrusion attempts instead of having to perform data mining
Steps for New Log Files

- Download log file
- R Studio to clean, create CSV file
- Import into SQL Server
- Use original query to find intrusion attempts
Cleaning / Importing August Logs

```
# importing the access logs from text file
filename <- C:/Users/osuki/Desktop/access_log_Aug95.txt
# these were two packages that needed to be installed
# in order to use the str_split feature
# install.packages("stringr")
# install.packages("dplyr")
# referencing the package libraries
library(stringr)
library(dplyr)
# opening the file
conn <- file(filename, open = "r")
# reading each line from the file
lines <- readLines(conn)
```

```
# selectively picking order and columns for output
final <- df[c(1, 4, 5, 6, 7, 8, 9, 10)]
# labeling the columns
colnames(final) <- c("Host", "Timestamp", "TimeZone", "Command"
writing the output file
write.csv(final, C:/Users/osuki/Desktop/aug.csv)
# closing the original file connection
close(conn)
writing the output file
write.csv(final, C:/Users/osuki/Desktop/aug.txt)
```

![SQL Server Import and Export Wizard](image)

The execution was successful

<table>
<thead>
<tr>
<th>Details:</th>
<th>Status</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initializing Data Flow Task</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Initializing Connections</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Setting SQL Command</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Setting Source Connection</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Setting Destination Connection</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Validating</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Prepare for Execute</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Pre-execute</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Executing</td>
<td>Success</td>
<td></td>
</tr>
<tr>
<td>Copying to &quot;dbo&quot;,&quot;NASALogsG&quot;</td>
<td>Success</td>
<td>156993 rows transferred</td>
</tr>
<tr>
<td>Post-execute</td>
<td>Success</td>
<td></td>
</tr>
</tbody>
</table>
Successful Implementation
Significance

- Using our query, we were able to detect highly likely intrusion attempts in our sample.
- Rules from original set applied to a completely different data set, with similar results.
- Found 15 intrusion access attempts per ~3.4 million records (0.0004 %).
- Can be used to find (in real time) malicious attacks and/or track previous attempts over time.
Methodology Review

- Download and clean data using R Studio
- Rounds of pre-processing
- Cluster plot, Mahalanobis, Outliers, Chi-Squared, Grubb’s Tests
- Identification of potential outliers
- Determination of common traits
- Application of rules to new data set
Conclusions

- Pre-processing is the most important step!
- Unsupervised methods are useful when dealing with large data sets with no clear starting point
  - Cluster plot analysis
- Many tests are needed to help identify potential outliers
- Rule generation may take multiple rounds/attempts
A Robust Decision Tree Algorithm for Imbalanced Data Sets

Wei Liu*  Sanjay Chawla*  David A. Cieslak†  Nitesh V. Chawla†
Summary

- Designed of a new decision tree algorithm
  - Robust and insensitive to size of classes
- Creation of a new measure, CCP, to counter the bias of Information Gain that towards the majority class.
- Top down and bottom up approach that use Fisher’s exact test, and they yield a classifier that performs statistically better than traditional decision trees.
Traditional Decision Trees

- Decision trees such as C4.5 split an attribute whose partition provides the highest confidence.
- High confidence rules do not necessarily imply high significance in imbalanced data.

Figure 1: Approximation of information gain in the formula formed by $\text{Conf}(X \rightarrow y)$ and $\text{Conf}(\neg X \rightarrow y)$ from Equation 2.7. Information gain is the lowest when $\text{Conf}(X \rightarrow y)$ and $\text{Conf}(\neg X \rightarrow y)$ are both close to 0.5, and is the highest when both $\text{Conf}(X \rightarrow y)$ and $\text{Conf}(\neg X \rightarrow y)$ reaches 1 or 0.
Class Confidence Proportion Decision Tree (CCPDT)

- Developed combat Information Gain that results in decision tree rules which are biased towards the majority class.
- Method uses top-down plus bottom-up approach and the Fisher’s exact test to prune branches of the tree which are not statistically significant.
- CCP entropy is insensitive to class skewness (will always have a fixed pattern)
- Entropy is maximized when a node has an equal number of elements from both splitting classes