Accelerate your ability to sniff out application exceptions and detect outliers in performance KPIs

PJ Pokhrel
Performance Engineer | Stubhub

Steve Veio

Ops Manager | StubHub

Eurus Kim

Staff ML Architect | Splunk

Agenda

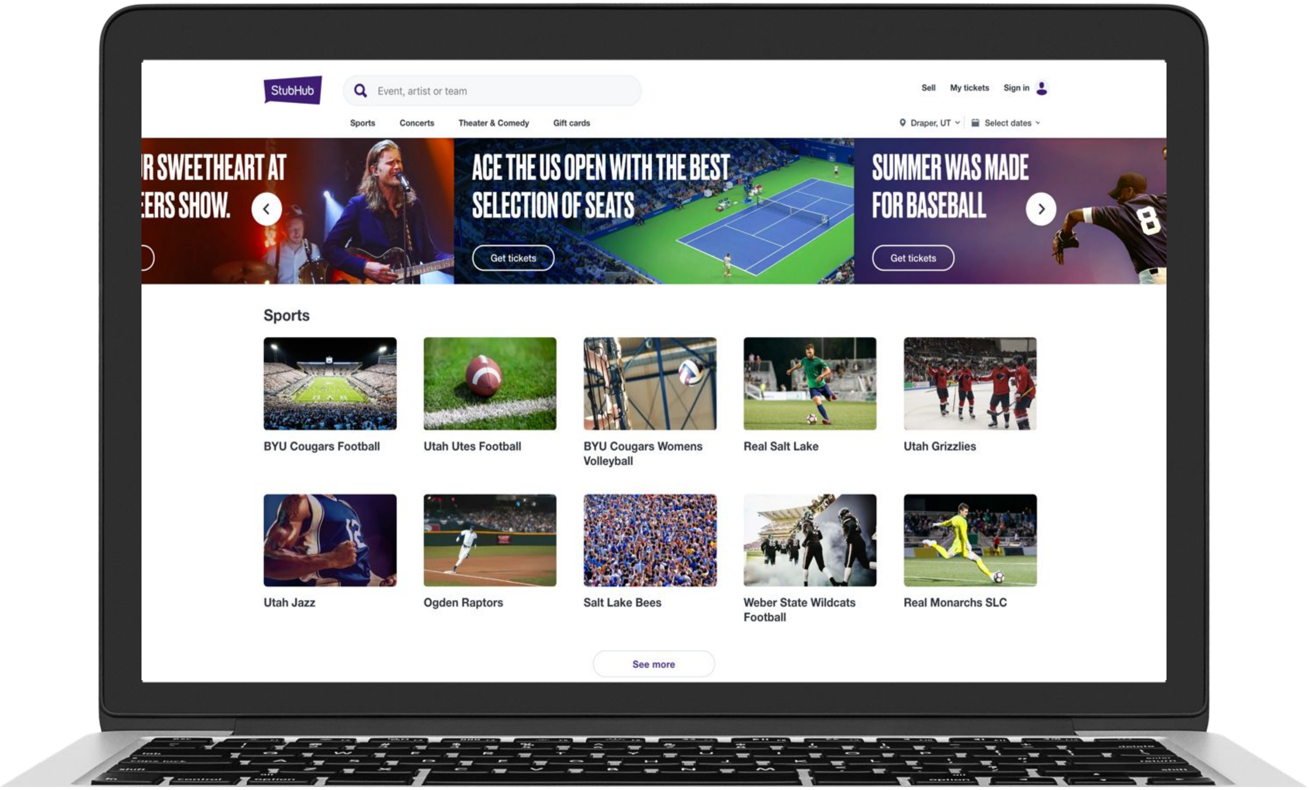
1. Building an Exception Sniffer
2. Process and setup
3. Applying the Use Cases
4. Splunk Metrics and Machine Learning
5. Enhanced Use Cases - Smarter Alerts
6. Summary

StubHub

Introduction to StubHub

StubHub is the world's most trusted ticket marketplace owned by eBay, which provides services for buyers and sellers of tickets for sports, concerts, theater and other live entertainment events

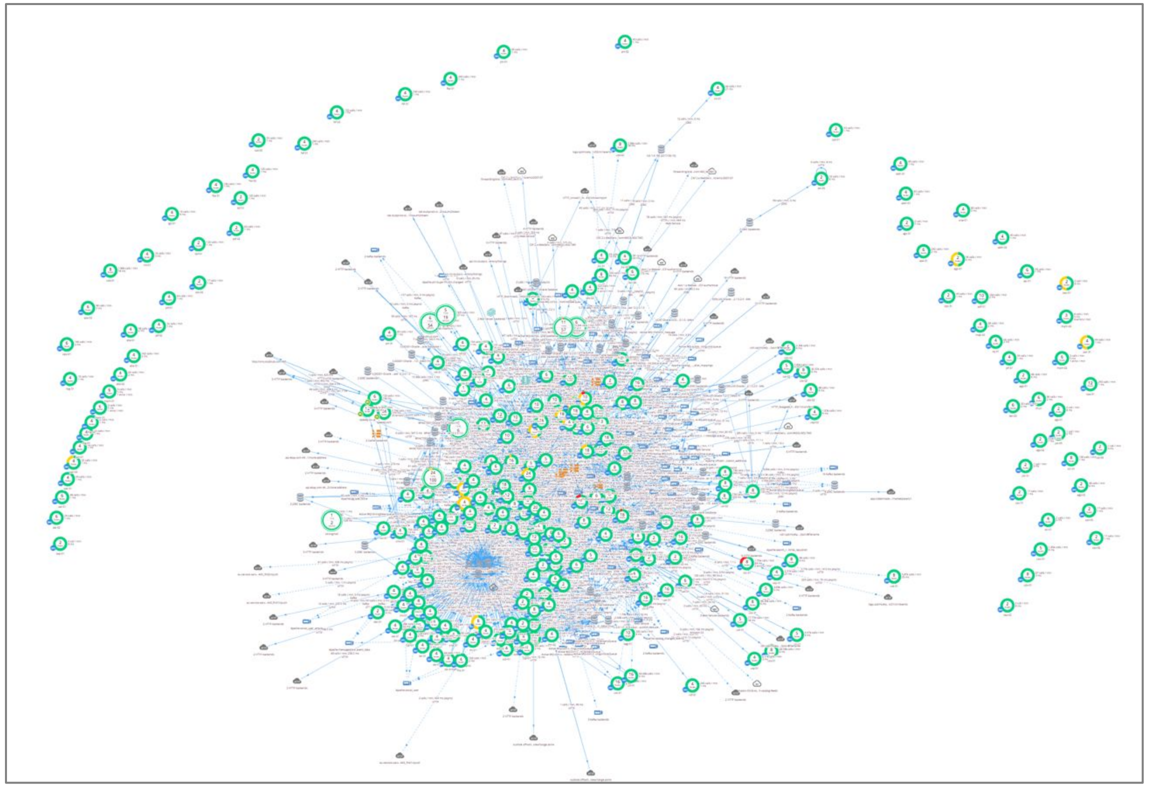
E-commerce site with desktop, mobile-web and native app products



Our Stack

How do you detect production issues early in this complexity?

- Distributed microservice architecture
- About 70 roles (pools/instance groups)
- About 4700 servers out of which 1450 servers running Java
- Three pods (production environment)
- Over 1000 endpoints



Exceptions Sniffer

What is the exceptions sniffer?

“Exception Sniffer” is the name we gave our tool that helps us extract, track and use exceptions data to gain insights into our application behavior and performance.

Tracks java and business exceptions on all of our servers running java

Common Types of Exceptions in our Stack

- java.io.IOException
- java.lang.NumberFormatException
- java.lang.NullPointerException
- java.net.SocketTimeoutException
- java.sql.SQLException
- …

- SHBadRequestException
- SHResourceNotFoundException
- UserNotAuthorizedException
- …

Exceptions Sniffer v1 (old version)

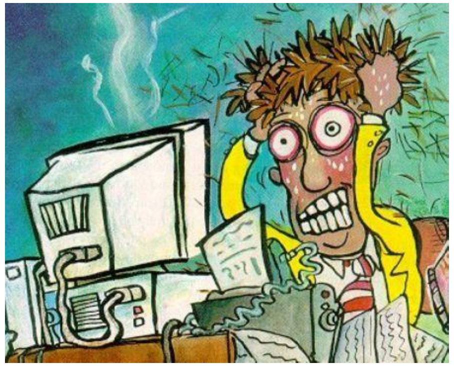
Architecture overview and issues

Overview:

- Internal java application which sniffs Errors and Exceptions in java apps from application logs
- Calls Splunk REST APIs
- Data processed by the sniffer and saved to PostgreSQL
- Rules engine
- Alert manager module to send alerts

Issues:

- Became slower as data grew
- Time consuming
- Server maintenance
- Dependent on Splunk REST APIs
- No native Machine Learning support



Exceptions Sniffer v2!

Overview of exceptions sniffer v2

- Built in Splunk
- Set of data models, metrics, dashboards and alerts
- Uses Splunk components: metric store, alert, dashboard, machine learning functions etc
- Allows us to store lot of data without worrying about space, reducing time to generate weekly and monthly reports

Canon EOS 350D DIGITAL |100 iso |1/2500 sec |f/2.8 |100 mm |28/04/2007 11:37:26 |© Andrea Denzler | www.AndreaPlanet.com

Requirements and Use Cases

- Deeper insights into data patterns and ability to use trends for debugging and troubleshooting
- Less time maintaining application and servers and lean hardware / storage
- Better alerting
- Fast searching of large amounts of historical data
- Creating month to date trends
- Whitelist functionality

Journey

How we got started?

- Started very simple :)
  - index=java AND exceptions
- Next added auto extracted **java_exceptions** field
  - index=java and java_exceptions=*
- Next added more dimensions
  - index=java and java_exceptions=* | stats count by role,pod,java_exceptions
- Next...

Journey: Event Logs

Event logs retention and performance

- Querying events logs was taking very long especially for weekly and monthly reports
- Event logs only had 30 days retention, so historical data was lost and we did not have enough data to make a good model

Journey: Splunk Metric

We used our initial search query and **mcollect** to store data as a **metric** and **mstats** to query the metric

Save:
(external version)

```
index=java java_exception=*
| stats count AS _value by
_time,host,dimension1,dimension2
| eval
metric_name="java.exceptions.count"
| mcollect index=metrics
```

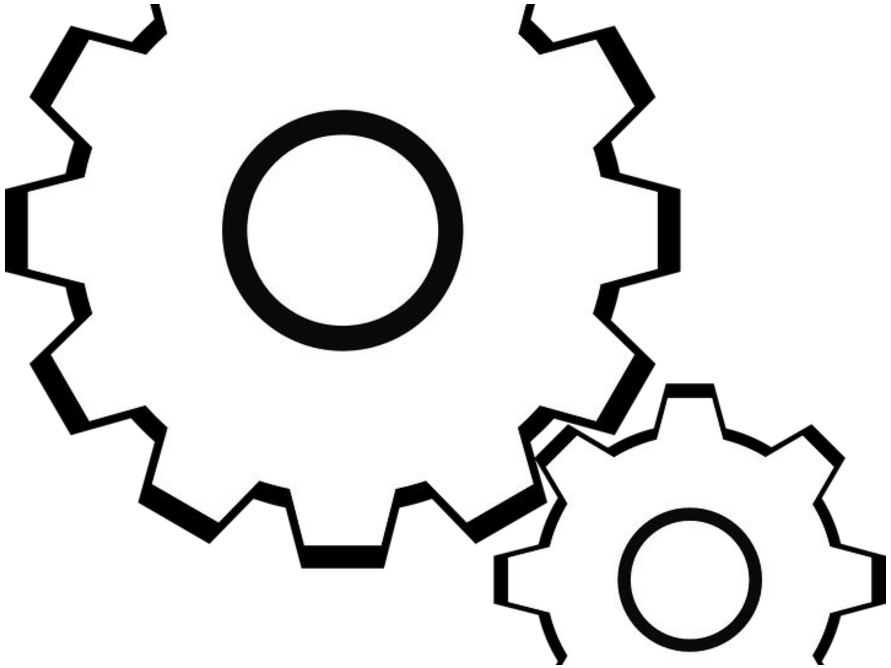
Query:
(external version)

```
| mstats sum(java.exceptions.count)
as "ExceptionCount" where
index=metrics earliest=-7d@w1
latest=@w1 span=1d
```

Splunk Metrics

Data retrieval performance before and after

- Searching 15 minutes of raw logs
  – 5 minutes 59 seconds
- Searching 15 minutes or metrics index data
  – 3 seconds

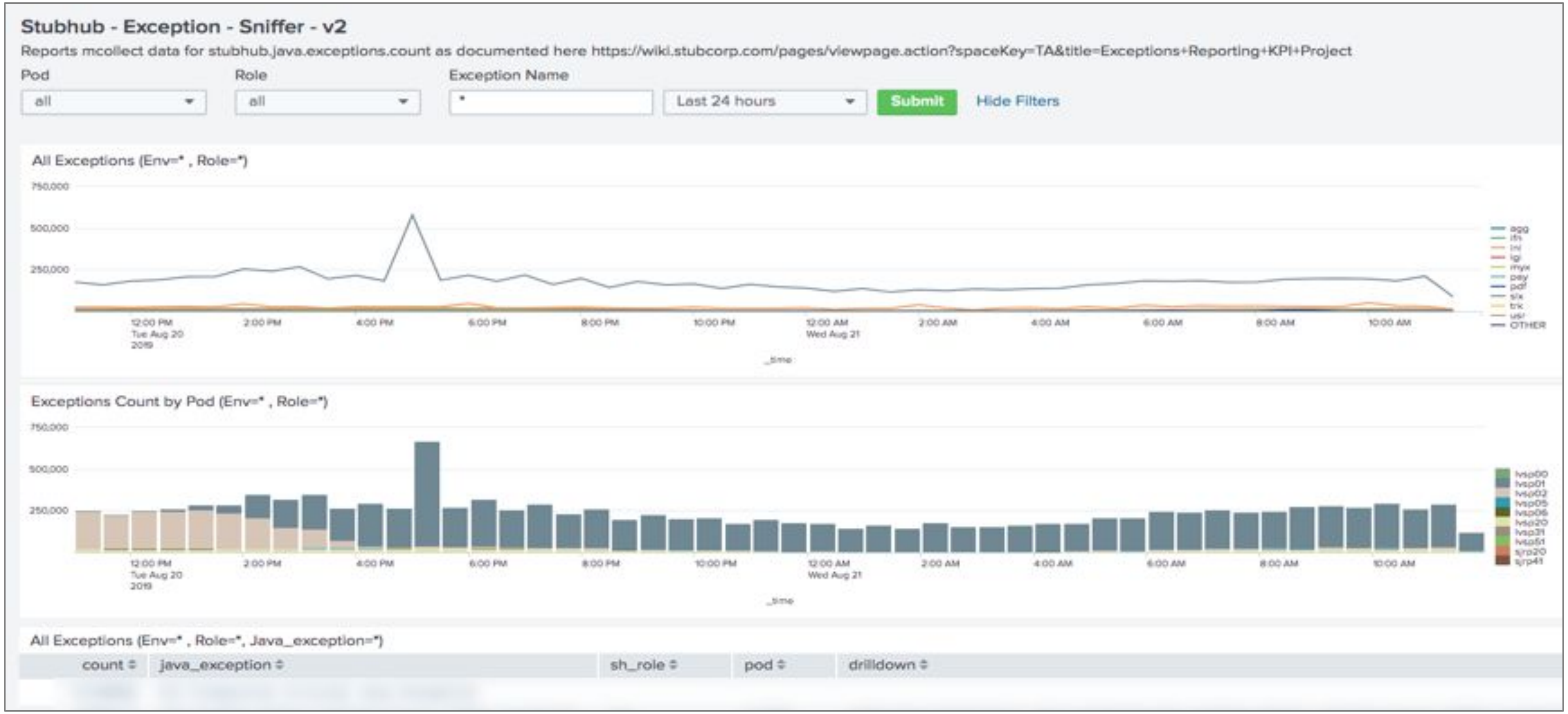


Journey: Exception Sniffer Use Cases

- Dashboards and Views
- Reports and Trending
- Used for DevOps Alerts
- Used for modeling for anomaly detection

Dashboard

Splunk dashboard that allows users to filter and group data



Weekly Report

Weekly exceptions report heatmap view by role

Exception Rate By Role

sh_role	2019- Mon	2019- Tue	2019- Wed	2019- Thu	2019- Fri	2019- Sat	2019- Sun
[illegible]	0.62	0.62	0.60	0.68	0.52	0.74	0.73
[illegible]	0.00	0.00	0.01	0.00	0.02	0.00	0.00
[illegible]			0.00		0.00		
[illegible]	0.02	0.02	0.03	0.05	0.04	0.03	0.03
[illegible]	0.08	0.12	0.09	0.16	0.12	0.09	0.06
[illegible]	0.00	0.00	0.00	0.00	0.00	0.00	0.00
[illegible]	0.46	0.50	0.49	0.51	0.50	0.48	0.48
[illegible]					0.00		
[illegible]	0.93	0.95	0.92	0.92	2.10	3.37	3.35
[illegible]		0.00	0.00	0.00	0.00	0.00	
[illegible]	0.01	0.01	0.01	0.01	0.04	0.01	0.00
[illegible]	0.07	0.07	0.19	0.10	0.09	0.12	0.14
[illegible]	0.02	0.02	0.02	0.02	0.05	0.02	0.02
[illegible]	0.03	0.03	0.03	0.03	0.04	0.03	0.03
[illegible]					0.00		
[illegible]	0.02	0.02	0.02	0.03	0.05	0.02	0.02

Error percentage calculation

Using tstats for calculating error rate

- We had exceptions count as a metric but no other reference to use for calculating error rate
- Used tstats to solve this problem to calculate error percentage

Tstats example:

```
| tstats count WHERE index=java
sourcetype=log4j by sh_role _time
span=1d
| outputlookup
exception_sniffer_tstat_output.cs
v
```

Monthly Report

Monthly exceptions trend report

Reports are great, but we also needed data in real-time to take action

Exploratory Data Analysis

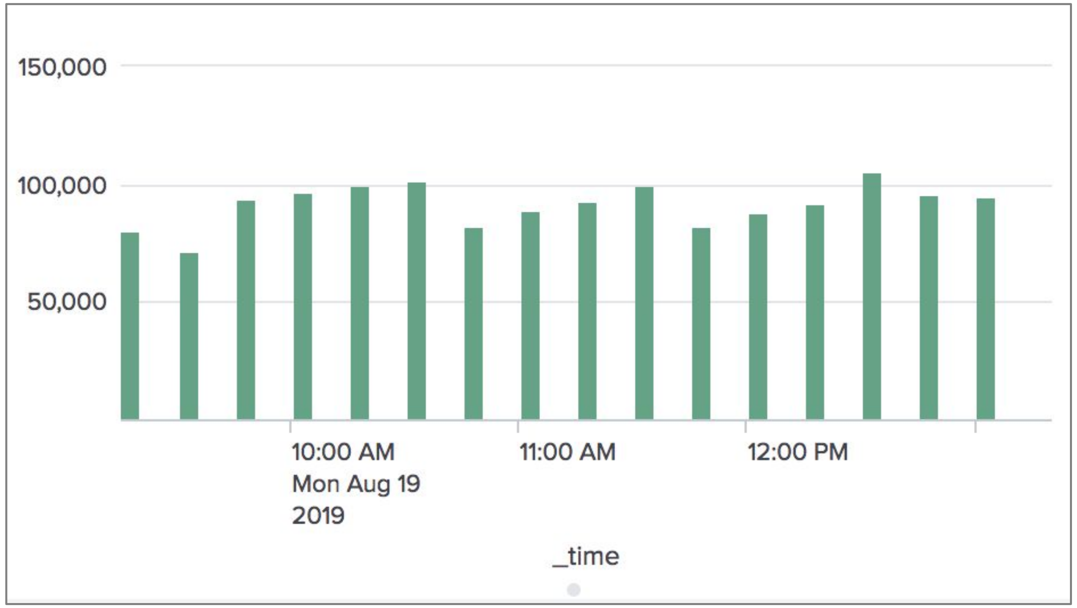
Overview of exceptions data

- About 70 roles (cluster/instance groups/pools) with different exception rate patterns
- Disparate distinct shape of data seasonality
- Lots of high and low values
- Same data different patterns
- Typically 40 million logging events per minute

Our Exception Data Pattern

Pattern 1

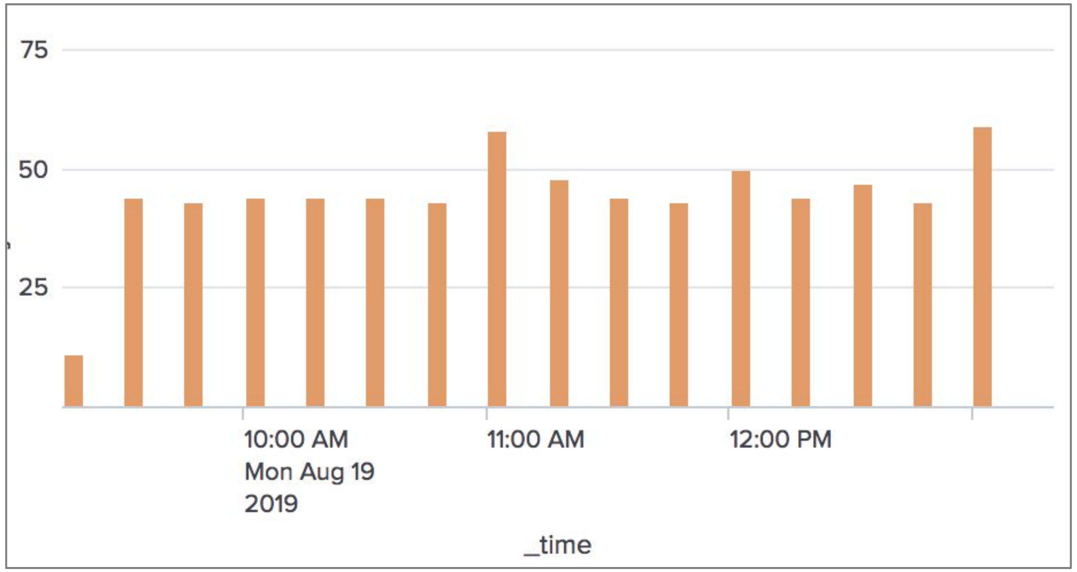
- Some roles have large number of ongoing exceptions (mostly business exceptions)
- Brokers and Selling related errors
- Exception rate is greater than 1000 exceptions per minute on average
- Some seasonality (Large spikes at different times of the day)



Our exception data pattern

Pattern 2

- Exception rate is greater than 100 but less than 1000 exceptions/min on average
- Large variance with lots of high low values
- Ancillary roles, page controllers, batch services



Our Exception Data Pattern

Pattern 3

- Very few or zero ongoing exceptions

Alerts

Actionable alert policies

New Exceptions Alert

- A scheduled job writes exceptions for that day to a outputlookup file at midnight
- Alert job queries exceptions for last 15 minutes and filters the result using the outputlookup file

Critical Exceptions Alert

- Created a outputlookup file which contains list of critical exceptions (ex java.lang.OutOfMemoryError)
- Alert job runs every 15 mins and checks results and reports if any exceptions from the critical exceptions list were found

Needed Intelligent Alerts

Experiments with creating smarter alerts

- Threshold method
  - Standard deviation
  - Standard deviation with sliding window
  - Median absolute deviation
- Other ML algorithms
  - Clustering to find underlying structure in exception data
  - Probability density function

Understanding a bit of the ML behind smarter alerts

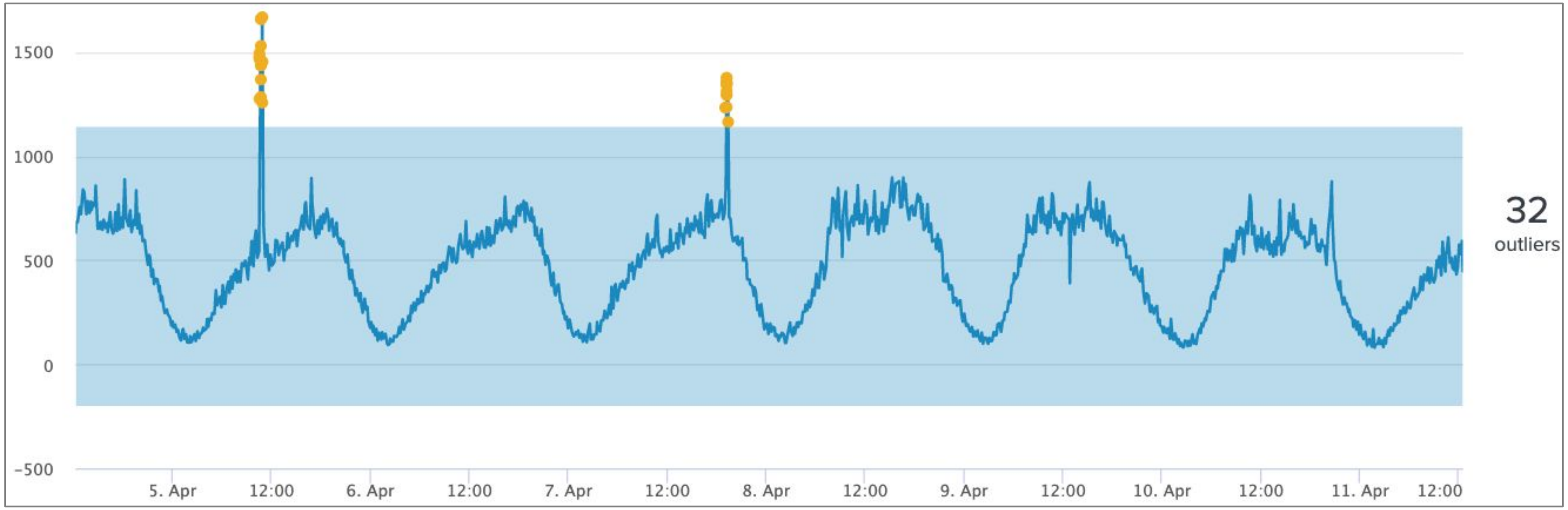
Eurus Kim | ML Architect | Splunk

splunk> .conf19

Numeric Outlier Detection with MLTK

Trying to create smarter alerts with statistics

Starting with basic thresholding using
Standard Deviation, Median Absolute Deviation, or Interquartile Range

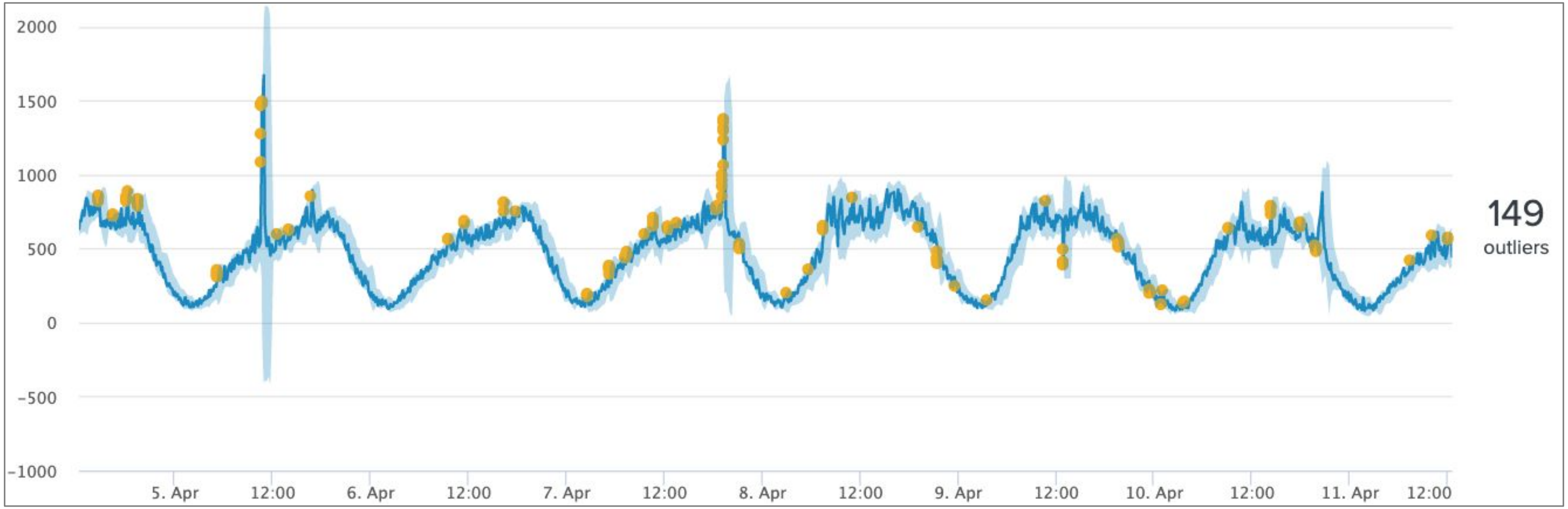


Global outliers are found, but local outliers are undetected
For seasonal data, thresholds are too large at certain times

Numeric Outlier Detection with MLTK

Getting a little more advanced

Using Standard Deviation with a sliding window

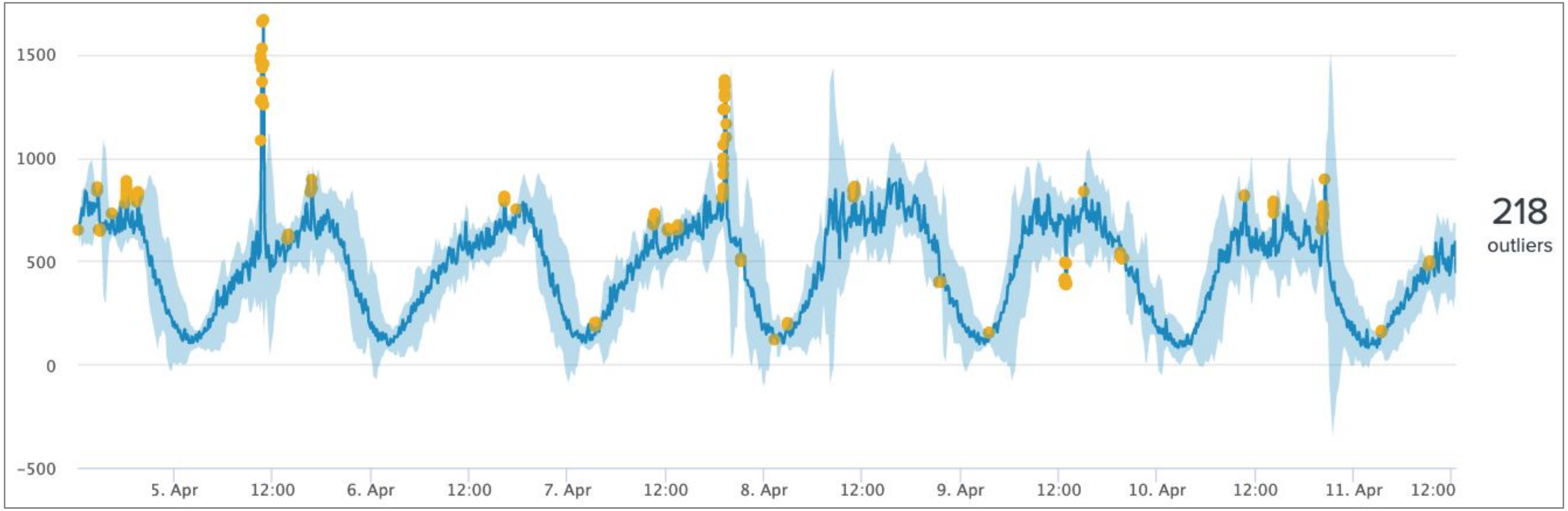


Both global and local outliers found, but now it’s way too noisy
Thresholds are too small or large at certain periods

Numeric Outlier Detection with MLTK

Getting a little more advanced

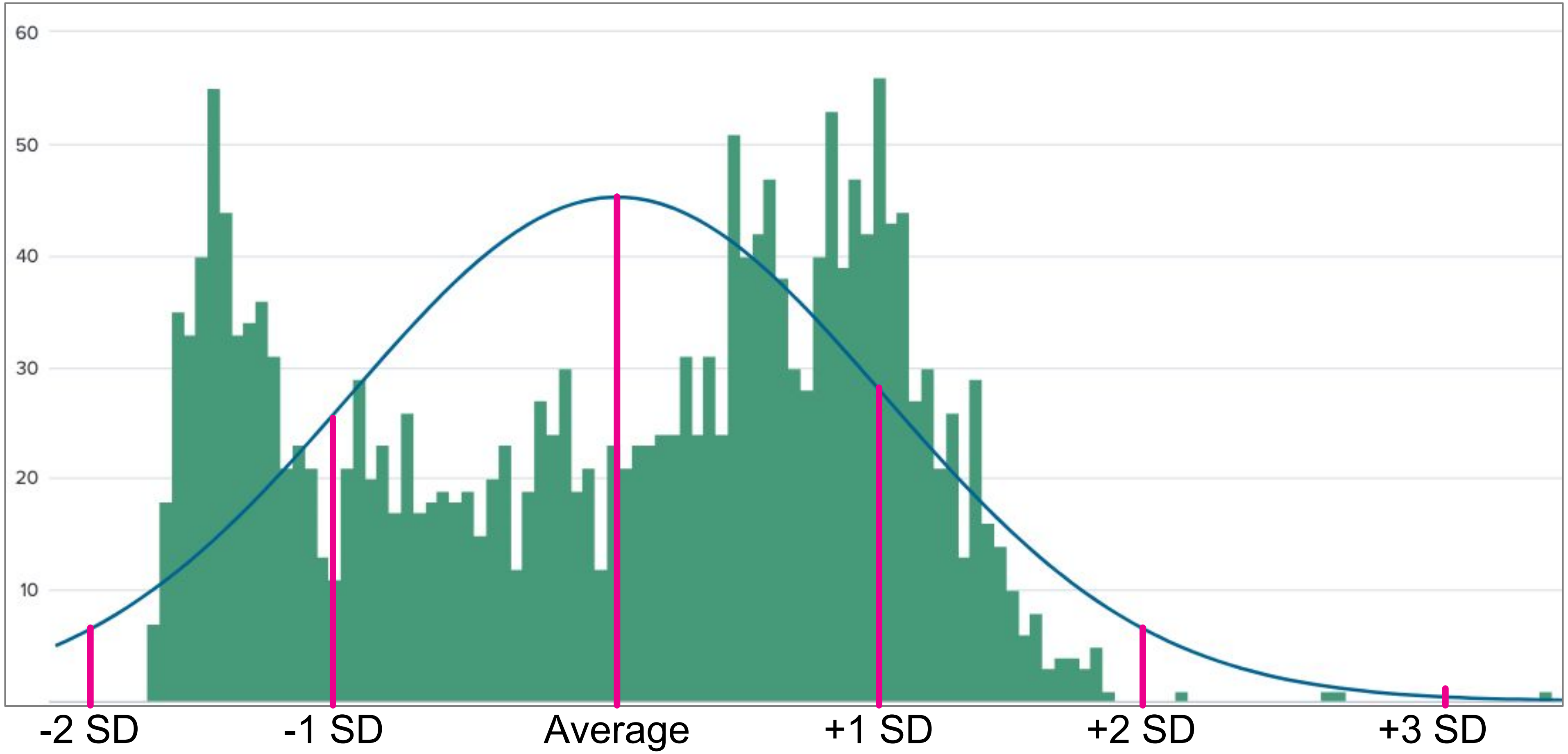
Further testing using Median Absolute Deviation with a sliding window



The boundaries look better, but stilll appears to be way too noisy

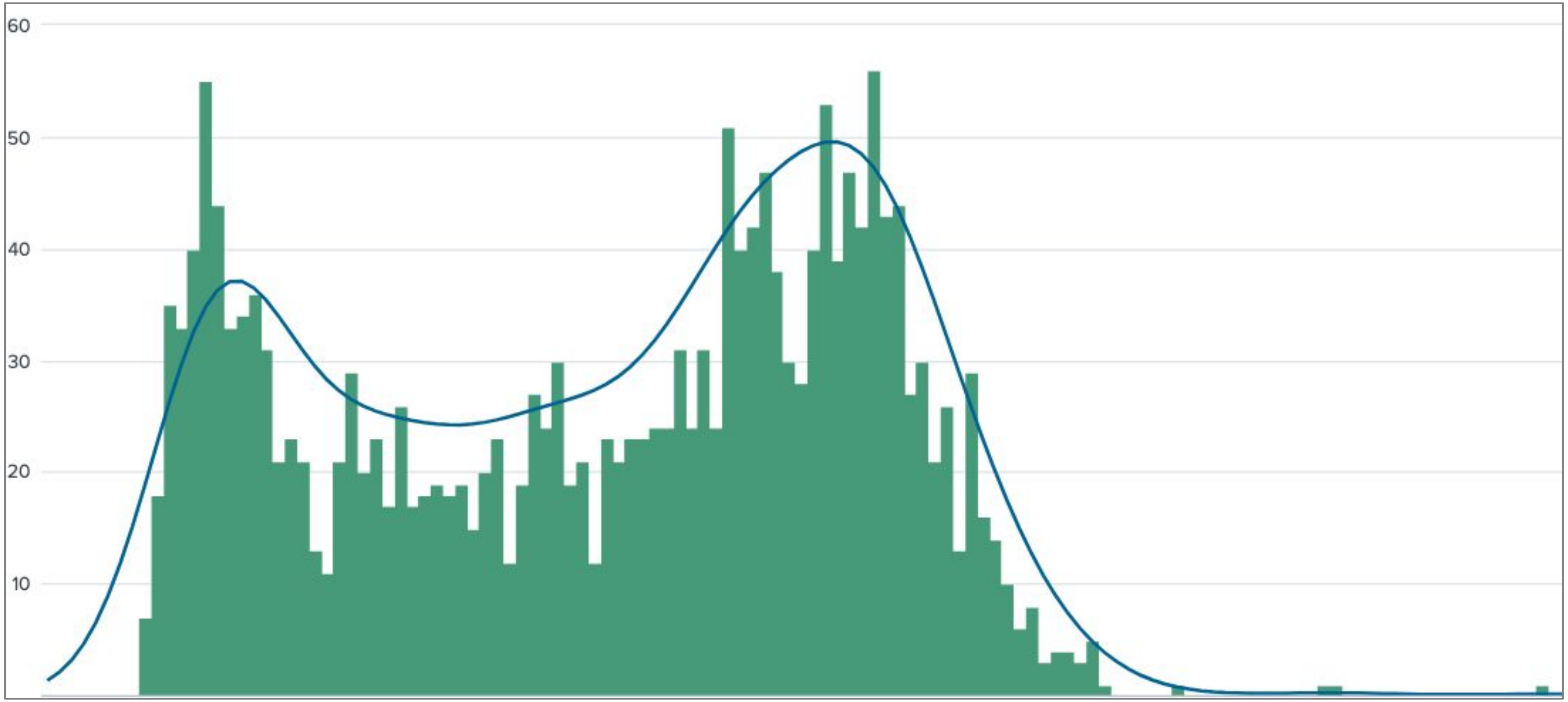
Understanding the Shape of our Data

Viewing the frequency of values (histogram) against a normal distribution curve



Understanding the Shape of our Data

What if we could draw a curve that better fits our data?



Leveraging the DensityFunction Algorithm

Allowing the math to figure out the best fit curve

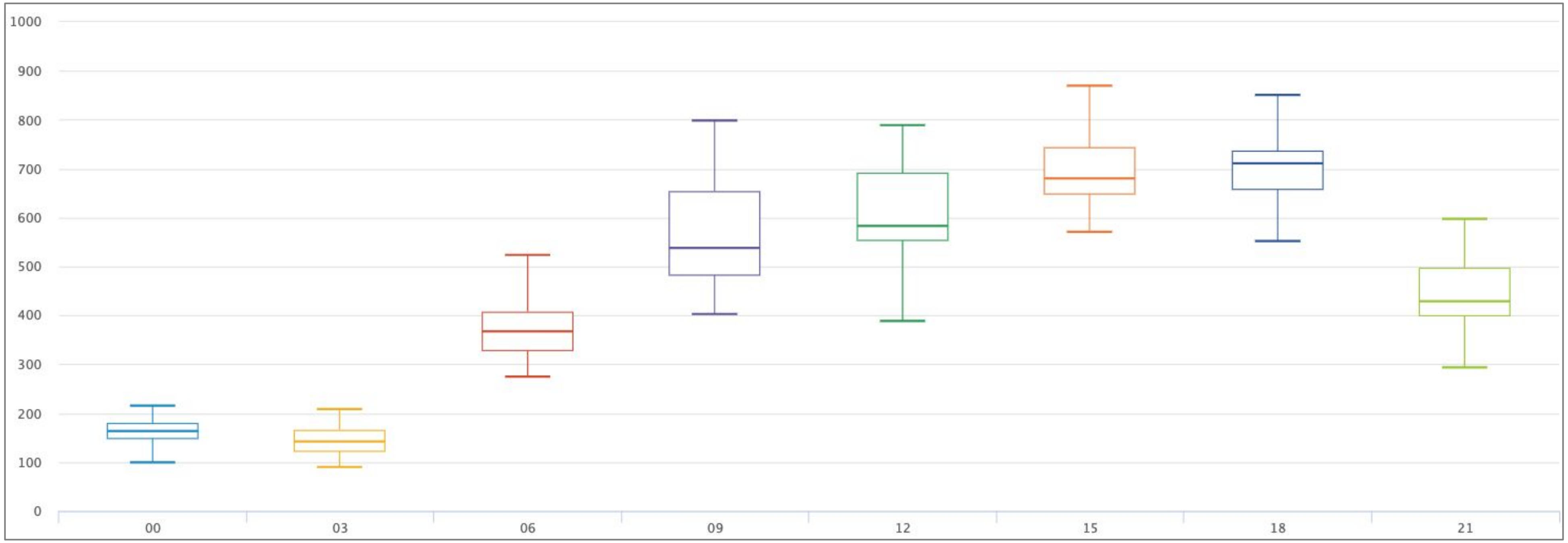

```
| fit DensityFunction requests threshold=0.01
  into MyModel

| apply MyModel
```

Splitting our Data with DensityFunction

Given that our data is cyclical, should we split the data by time?

The multi-modal nature of our data probably to the fact that our data is cyclical
Consider what else you may want to split your data by (app type, user group, etc)



Boxplot of every 3rd hour of the day

Splitting data using DensityFunction

| fit DensityFunction requests by "hour" into MyModel

date_hour	mean	std	type	cardinality
0	165.1809523809524	23.770241881246612	Auto: Gaussian KDE	420
3	148.09285714285716	27.473770112922015	Auto: Gaussian KDE	420
6	372.22857142857146	56.177356418308435	Auto: Gaussian KDE	420
9	578.2666666666667	106.01401674322008	Auto: Gaussian KDE	420
12	609.4965517241379	81.80150835845626	Auto: Gaussian KDE	435
15	700.6714285714286	74.3173481180035	Auto: Gaussian KDE	420
18	700.6619047619048	65.55501557574933	Auto: Gaussian KDE	420
21	445.99285714285713	66.18622678205986	Auto: Gaussian KDE	420

Splunk Machine Learning Advisory Program

- Get help from the Splunk Data Scientists to solve your business use case with Machine Learning Toolkit
- Complimentary support with your Enterprise or Cloud license
- Early access to new Machine Learning features
- Results in opportunity to tell your success story with Splunk
- Contact mlprogram@splunk.com for more information

What to Learn More About Density Function?

Additional sessions to further deep dive on the theory and example use cases

Foundations/Platform Intermediate

FN1213 - The Two Most Common Machine Learning Solutions Everyone Needs to Know

SCHEDULE Wednesday, October 23, 12:30 PM - 01:15 PM

Eurus Kim, Staff ML Architect, Splunk

Amir Malekpour, Principal Software Engineer, Machine Learning, Splunk

Foundations/Platform Intermediate

FN1366 - Enhanced Anomaly Detection: Join T-Mobile and Splunk as we Deep Dive an Enterprise-IT Operational Use Case

SCHEDULE Wednesday, October 23, 01:45 PM - 02:30 PM

Iman Makaremi, Principal Product Manager – Machine Learning and AI, Splunk

Scott Garcia, MTS - Member Technical Staff, T-Mobile

Security, Compliance and Fraud Intermediate

SEC1374 - Augment Your Security Monitoring Use Cases with Splunk's Machine Learning Toolkit

SCHEDULE Thursday, October 24, 11:45 AM - 12:30 PM

Oliver Kollenberg, Security Consultant, Siemens AG

Philipp Drieger, Staff Machine Learning Architect , Splunk

Comparing Threshold vs Density Function

Actionable alert policies

Threshold Method:

Pros

- Easy to understand
- MLTK assistant available

Cons

- Doesn't support fit or apply
- Complicated to use for alerts

Probability Density Function (PDF):

Pros

- Better for outlier detection
- Supports fit and apply, so easier to setup

Cons

- Not available in MLTK assistant yet

Effective Alerts

Final findings for the most effective alert for each exception pattern

Pattern 3 (zero ongoing)

- Diff alert and basic static threshold

Pattern 2 (medium ongoing exception)

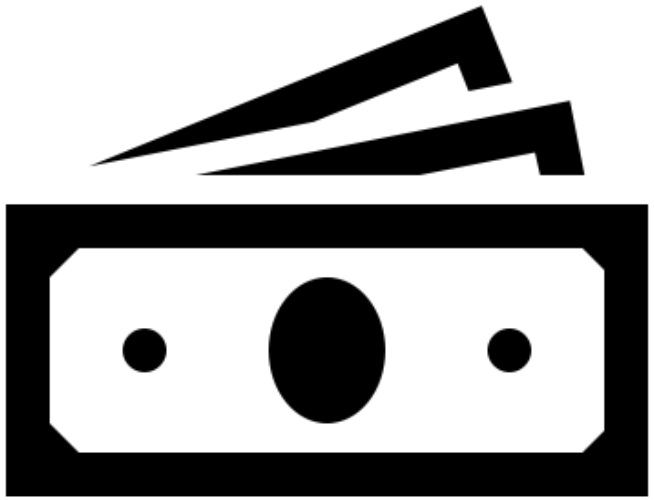
- Diff alert and probability density function

Pattern 1 (ongoing pattern)

- Diff method (Newexception alert and Critical exceptions)

How Tracking Exceptions Has Helped Us?

- 50% reduction in exceptions logging
  - Duplicate logging
  - Don’t log stack trace for business exception
- Found several hidden application issues
- Using exceptions as one indicator of an issue
- More accurate alerts
- Cleaner logs, reduced noise
- More accountability, better code quality



Lessons Learned

The lessons we learned…

1. Create metrics for faster data retrieval

2. Know your dataset! (avg, p90, median, standard deviation..etc)
   - Use histogram to get a better understanding of the distribution
   - Use PDF function to figure out the distribution pattern for you

3. Threshold method works well when data is normally distributed but can be a little complicated to create

4. For more complex data, create different alert policy for each pool

5. Each use case is different

.conf19
splunk>

Thank You!

Go to the .conf19 mobile app to

RATE THIS SESSION

Q&A

Steve Veio | Ops Manager
PJ Pokhrel | Performance Engineer
Eurus Kim | Staff ML Architect | Splunk