Using Machine Learning to Detect Traffic Anomalies

Jim Goodrich
Senior Sales Engineer | Splunk

Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or plans of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results may differ materially. The forward-looking statements made in the this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, it may not contain current or accurate information. We do not assume any obligation to update any forward-looking statements made herein.

In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only, and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionalities described or to include any such feature or functionality in a future release.

This session is a step-by-step tutorial on how to use **machine learning** to detect time series **anomalies**.

Based on real world experiences from a global content delivery network provider

The Scenario

Detecting Anomalies in Network Traffic Patterns

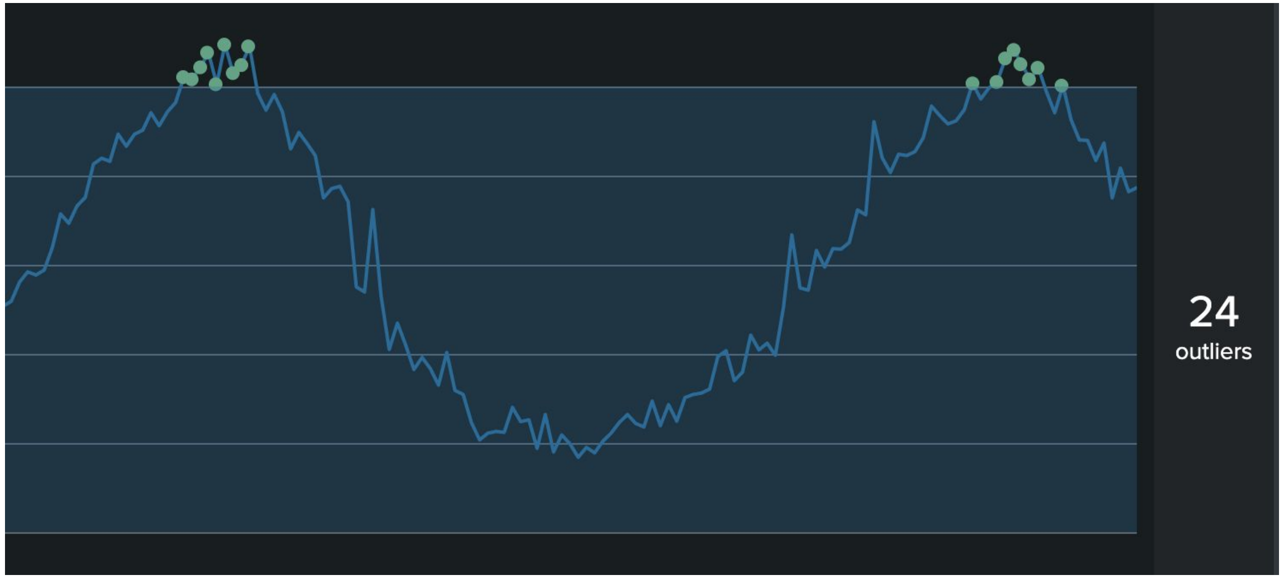
E-commerce Example

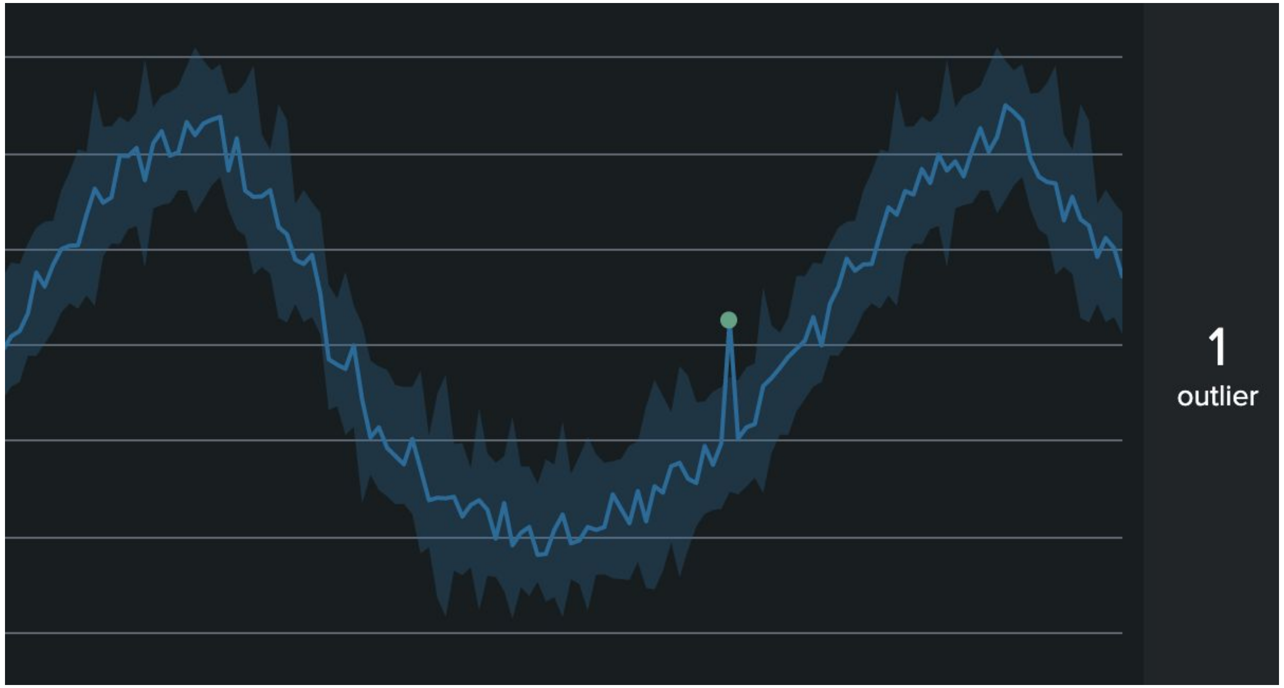
- Commerce site hosted by CDN
- Traffic is analyzed by multiple security toolsets
- Single highly enriched network log generated

Customer Goal

- Use ML to detect anomalies in specific network traffic patterns (i.e. bot traffic, DoS attacks, false positive/negative conditions)

Why Use Machine Learning?





Technique #1

Simple Spike Detection via Last Hour Analysis

Develop Your Base Search – Part 1

Search the last 60 minutes of logs for the specific condition

Set your latest to “-1m@m” to prevent measuring a partial minute

```
index=mydata condA=0 condB=1 earliest=-61m@m latest=-1m@m
```

Develop Your Base Search – Part 2

Group events into buckets (e.g. “bins”) of time

A short span (i.e. 1m) will be highly sensitive to short duration spikes

A long span (i.e. 15m) will be less sensitive to short duration spikes

```
index=mydata condA=0 condB=1 earliest=-61m@m latest=-1m@m
| bin _time span=1m
```

Develop Your Base Search – Part 3

Count the number of events

Split by _time and a field that differentiates entities

Limit your search to one entity. This is useful for visualization and tuning

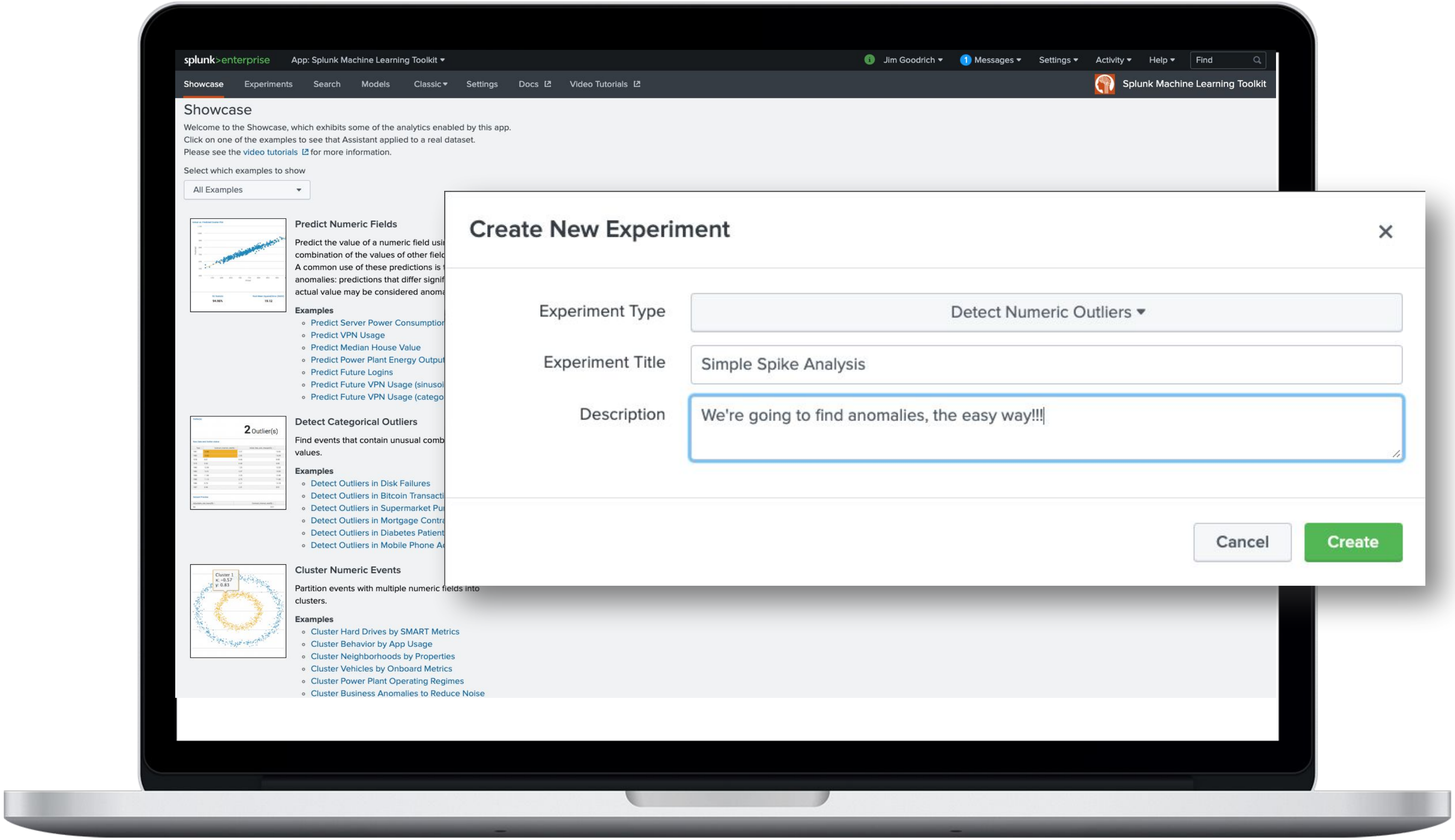
```
index=mydata condA=0 condB=1 earliest=-61m@m latest=-1m@m
| bin _time span=1m
| stats count by _time endpoint
| search endpoint="12345"
```

Data Quality Warning

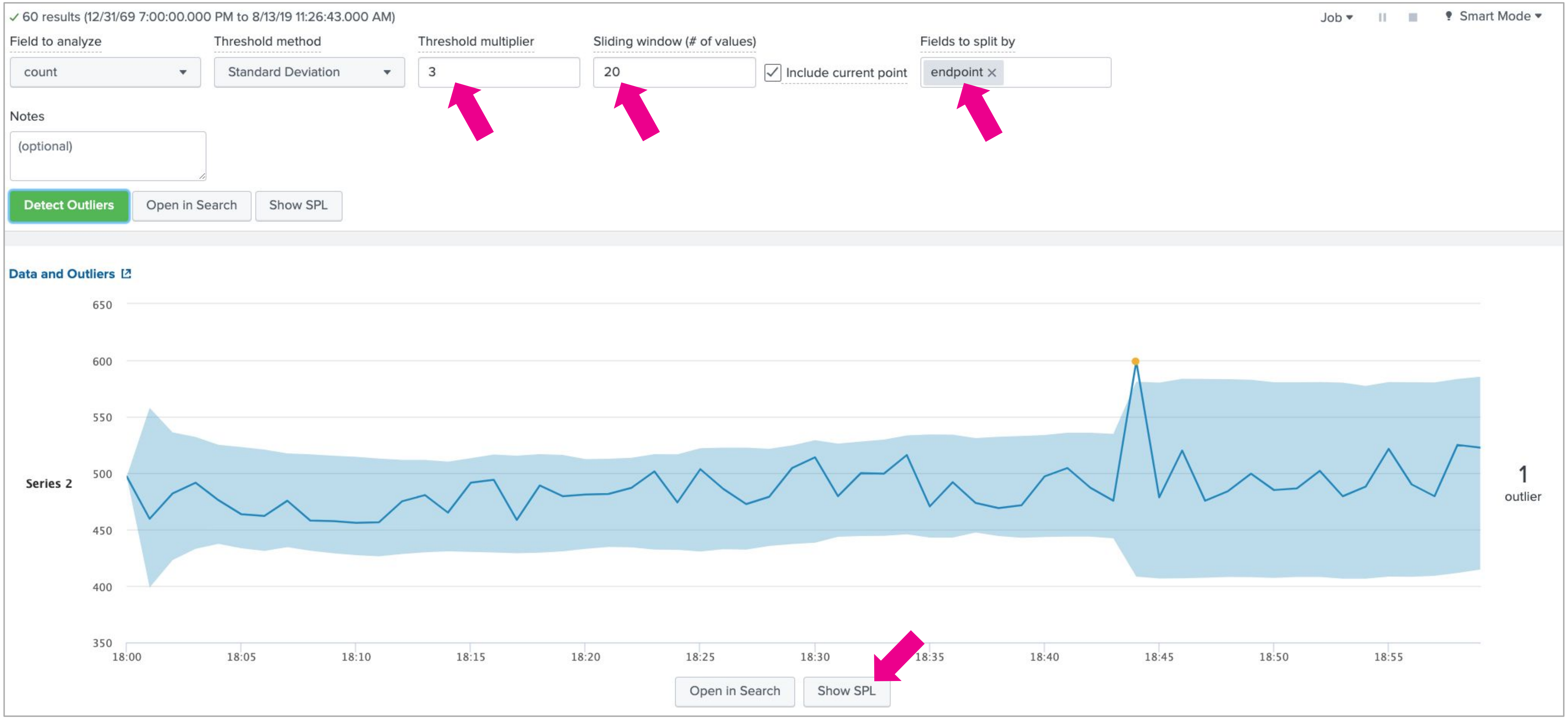
What Does This Look Like?

_time	count	endpoint
2019-07-01 00:00:00	291	12345
2019-07-01 00:01:00	294.5	12345
2019-07-01 00:02:00	303	12345
2019-07-01 00:03:00	301.5	12345
2019-07-01 00:04:00	295	12345
2019-07-01 00:05:00	301.5	12345
2019-07-01 00:06:00	300	12345
2019-07-01 00:07:00	312.5	12345
2019-07-01 00:08:00	293	12345

Source: https://giphy.com/gifs/spaceballs-password-12345-xT0GqJfdLcrcpSbZf2


splunk>enterprise
App: Splunk Machine Learning Toolkit
Jim Goodrich
Messages
Settings
Activity
Help
Find
Showcase
Experiments
Search
Models
Classic
Settings
Docs
Video Tutorials
Splunk Machine Learning Toolkit
Showcase
Welcome to the Showcase, which exhibits some of the analytics enabled by this app.
Click on one of the examples to see that Assistant applied to a real dataset.
Please see the video tutorials for more information.
Select which examples to show
All Examples
Predict Numeric Fields
Examples
Predict Server Power Consumption
Predict VPN Usage
Predict Median House Value
Predict Power Plant Energy Output
Predict Future Logins
Detect Categorical Outliers
Examples
Detect Outliers in Disk Failures
Cluster Numeric Events
Partition events with multiple numeric fields into clusters.
Examples
Cluster Hard Drives by SMART Metrics
Cluster Behavior by App Usage
Cluster Neighborhoods by Properties
Cluster Vehicles by Onboard Metrics
Cluster Power Plant Operating Regimes
Cluster Business Anomalies to Reduce Noise
Create New Experiment
Experiment Type
Detect Numeric Outliers
Experiment Title
Simple Spike Analysis
Description
We're going to find anomalies, the easy way!!!
Cancel
Create

Simple Spike Detection = Success



Operationalize the Alert

Use the “Show SPL” button and grab the new code

Search for only outliers

Schedule this an an alert

```
index=mydata condA=0 condB=1 earliest=-61m@m latest=-1m@m
| bin _time span=1m
| stats count by _time endpoint
| search endpoint="12345"
| streamstats window=20 current=true avg("count") as avg stdev("count") as stdev  by "endpoint"
| eval lowerBound=(avg-stdev*exact(3)), upperBound=(avg+stdev*exact(3))
| eval isOutlier=if('count' < lowerBound OR 'count' > upperBound, 1, 0)
| search isOutlier=1
```

Technique #2

Detecting Anomalies with Probability Density Functions (PDF)

We are now looking for events that are **abnormal** based on history

Use each endpoints past to learn “normal” on an individual basis

Understanding the Machine Learning Process

Training

The “learning” part

Scheduled once per week

Testing

The “alerting” part

Scheduled periodically

Tuning

Visualize and tweak

Optimize the models

Probability Density Function Basics

Visualizing a **Normal Distribution**

The Empirical Rule

The Left & Right Boundaries

Training the Model Based on Time Variables

Time variables are used to train the model

These allow the model to isolate time into different buckets

You need a minimum of 5 training points per buckets (30 - 50 recommended)

Example Time Variable	How it Trains The Model	Minimum Training History
date_minutebin (i.e. 15m)	Allows the training to see 0, 15, 30, and 45 minutes as separate slices of time	5 hours
date_hour	Allows the training to know that hour 1 and hour 14 are different	5 days
date_wday	Allows the training to differentiate between Monday and Saturday	5 weeks

Training the Model – Part 1

Adjust the earliest and latest to achieve cardinality

Consider a longer span. This is to smooth/average the training data

```
index=mydata condA=0 condB=1 earliest=-36d@d latest=-1d@d
| bin _time span=15m
```

Training the Model – Part 2

Add your time variables

```
index=mydata condA=0 condB=1 earliest=-36d@d latest=-1d@d
| bin _time span=15m
| eval date_minutebin=strftime(_time, "%M")
| eval date_hour=strftime(_time, "%H")
| eval date_wday=strftime(_time, "%A")
```

Training the Model – Part 3

Add back your stats command.

Split by all time variables (as well as endpoint)

We'll call this the base search on later slides

```
index=mydata condA=0 condB=1 earliest=-36d@d latest=-1d@d
| bin _time span=15m
| eval date_minutebin=strftime(_time, "%M")
| eval date_hour=strftime(_time, "%H")
| eval date_wday=strftime(_time, "%A")
| stats count by _time date_minutebin date_hour date_wday endpoint
```

What Does This Look Like?

_time	count	date_minutebin	date_hour	date_wday	endpoint
2019-07-01 18:00:00	487	00	18	Monday	12345
2019-07-01 18:15:00	461.5	15	18	Monday	12345
2019-07-01 18:30:00	500	30	18	Monday	12345
2019-07-01 18:45:00	519.5	45	18	Monday	12345
2019-07-01 19:00:00	503	00	19	Monday	12345
2019-07-01 19:15:00	487.5	15	19	Monday	12345
2019-07-01 19:30:00	493	30	19	Monday	12345
2019-07-01 19:45:00	541.5	45	19	Monday	12345

Training the Model – Part 4

Use the fit command to train your data and store the model

Allow 5% of the area to be considered anomalous (similar to 2 stdev)

Manually set the distribution when you have a small training sample

```
<base search>
| fit DensityFunction count by "date_minutebin,date_hour,date_wday,endpoint"
into mydensitymodel threshold=0.05 dist=norm
```

Testing the Model

Search a recent time range
Apply the model and search for outliers
Schedule as an alert

```
<base search>
| apply mydensitymodel
| search "IsOutlier(count)"=1
```

Visualization and Tuning – Part 1

LowerBound

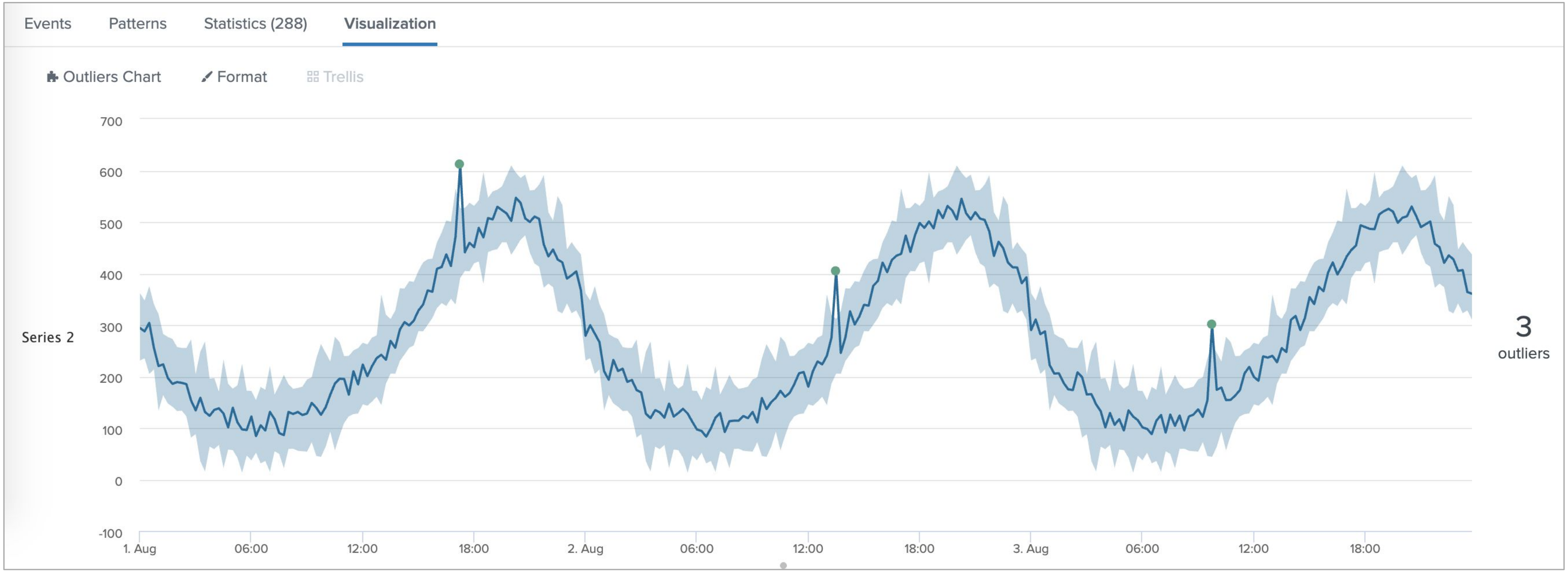
_time	count	date_hour	date_minutebin	date_wday	BoundaryRanges	IsOutlier(count)
2019-08-01 05:30:00	244.0	5	30	Thursday	-Infinity:13.1509:5e-06 224.2039:Infinity:5e-06	1.0

UpperBound

```
<your base search>
| apply mydensitymodel
| eval leftRange=mvindex(BoundaryRanges,0)
| eval rightRange=mvindex(BoundaryRanges,1)
| rex field=leftRange "-Infinity:(?<lowerBound>[^:]*):"
| rex field=rightRange "(?<upperBound>[^:]*):Infinity"
| fields _time, count, lowerBound, upperBound, "IsOutlier(count)", *
```

Visualization and Tuning – Part 2

Select the Outliers Chart visualization



Key Takeaways

Best Practices from Real World Testing

Performanc
e

Cool Kids Use TSTATS

1. Implemented an accelerated data model

2. Replaced base search with TSTATS

3. Achieved **99.7X** (not percent) query performance increase

```
| tstats count prestats=true FROM datamodel=MYD.MYD WHERE
MYD.condA=0 MYD.condB=1 earliest=-61m@m latest=-1m@m BY
_time MYD.endpoint span=15m
| eval date_minutebin=strftime(_time, "%M")
| eval date_hour=strftime(_time, "%H")
| eval date_wday=strftime(_time,"%A")
| stats count by _time date_minutebin date_hour date_wday endpoint
```

Quality

Getting Actionable Results

1. Implemented multiple anomaly detection techniques (simple spike & PDF)

2. Stored all detections in an anomaly index

3. Used aggregate analysis to determine when scale of anomalies was actionable

.conf19

splunk>

Thank You!

Go to the .conf19 mobile app to

RATE THIS SESSION