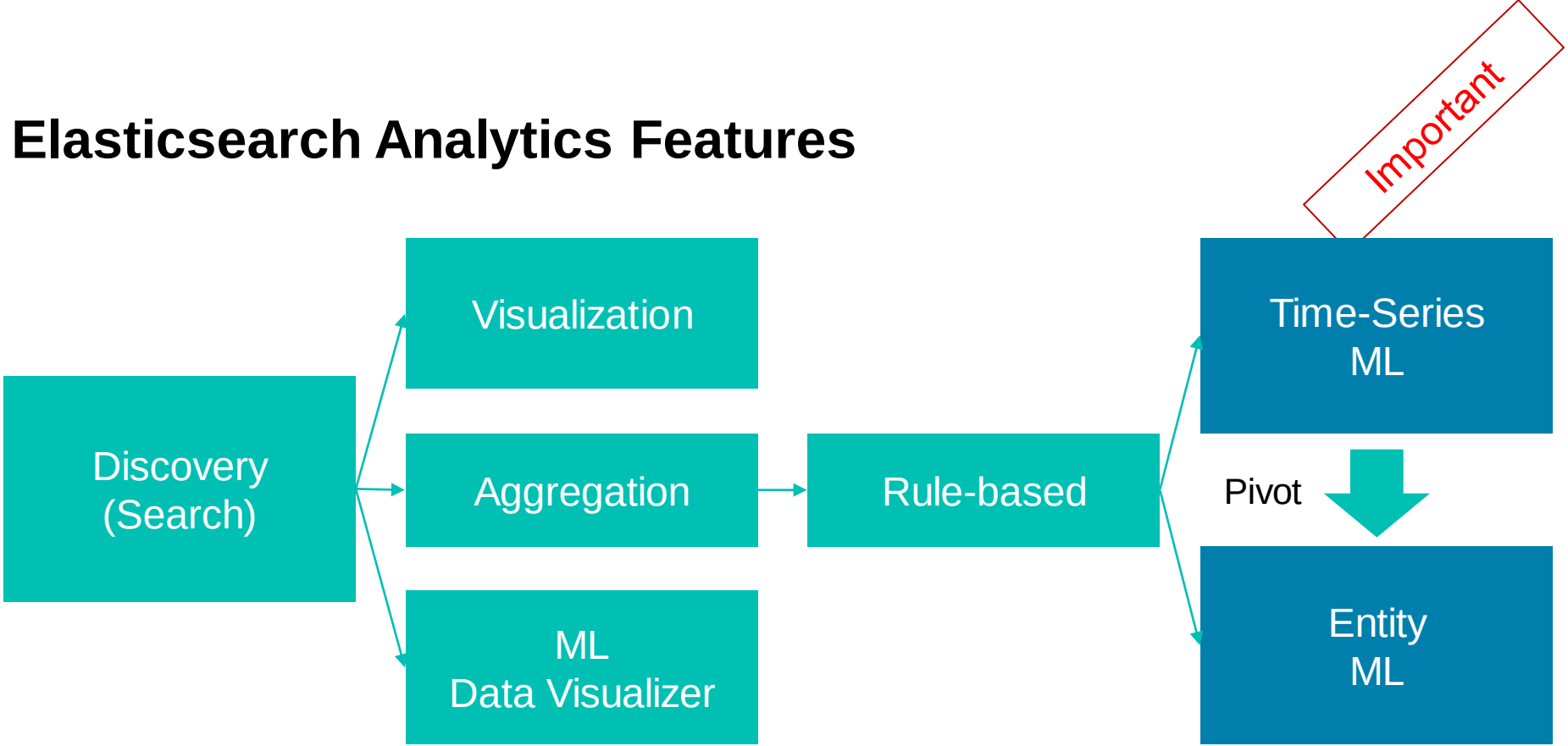
elastic

Elastic Machine Learning Workshop

Elastic
http://www.elastic.co

Understanding your DATA is Everything (Discussion)

Elasticsearch Analytics Features



elastic

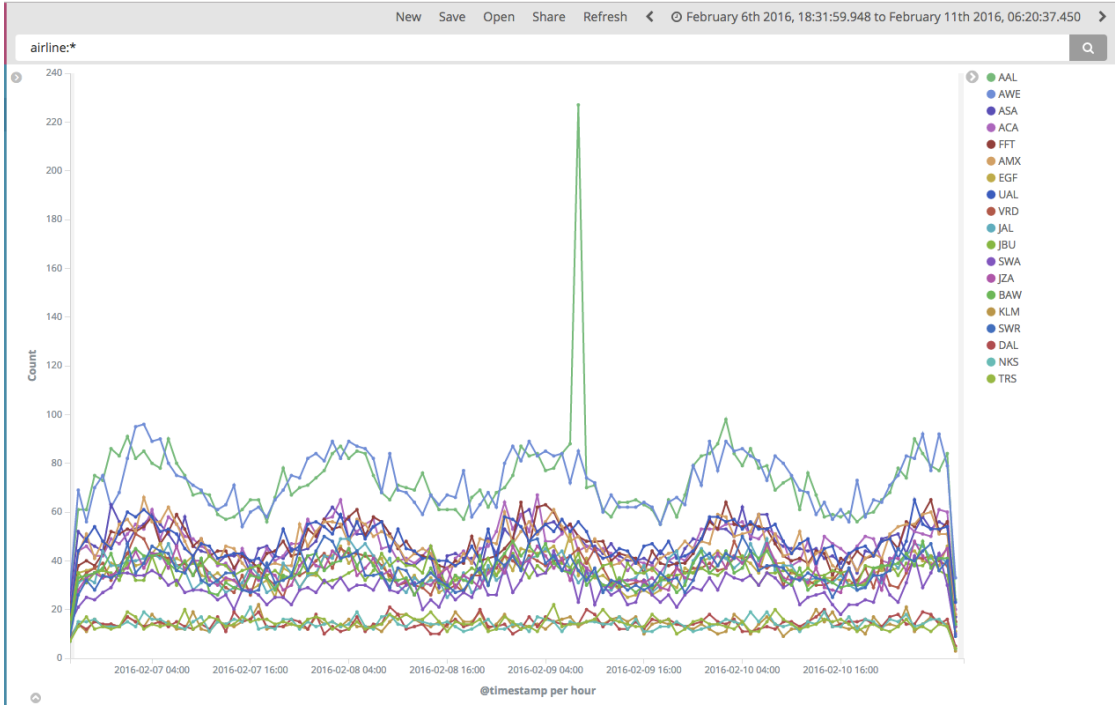
1. Times-Series based ML : Anomaly Detection Concepts

What is “Abnormal”

What is “Abnormal”?

What’s abnormal here?

Why?

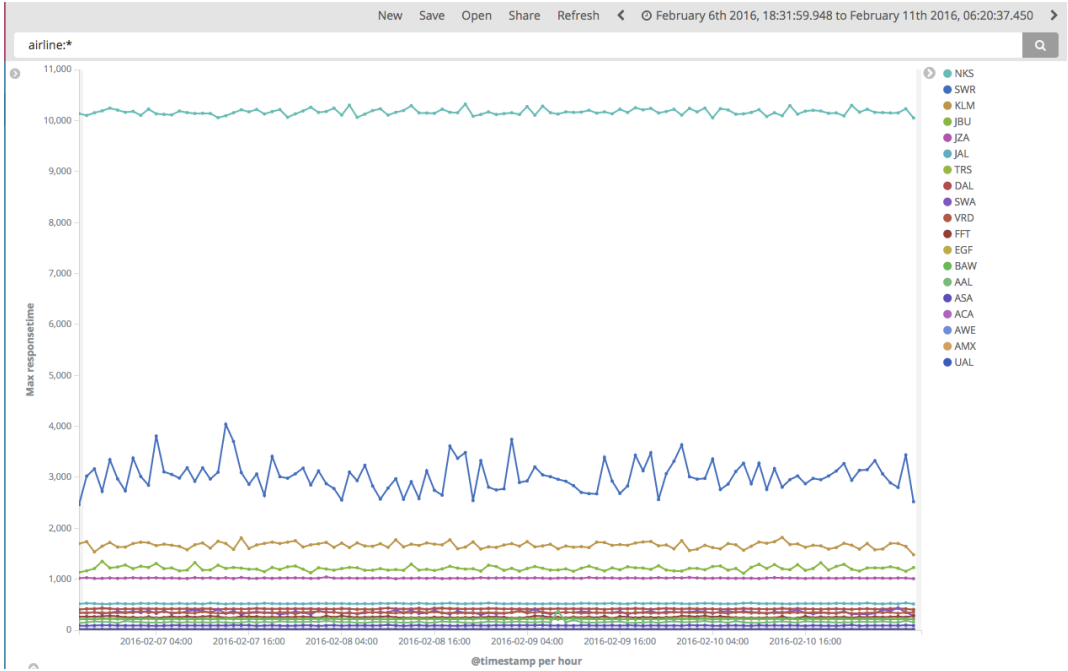


elastic

What is “Abnormal”?

What’s abnormal here?

Why?



elastic

What is “Abnormal”?

What’s abnormal here?

Why?

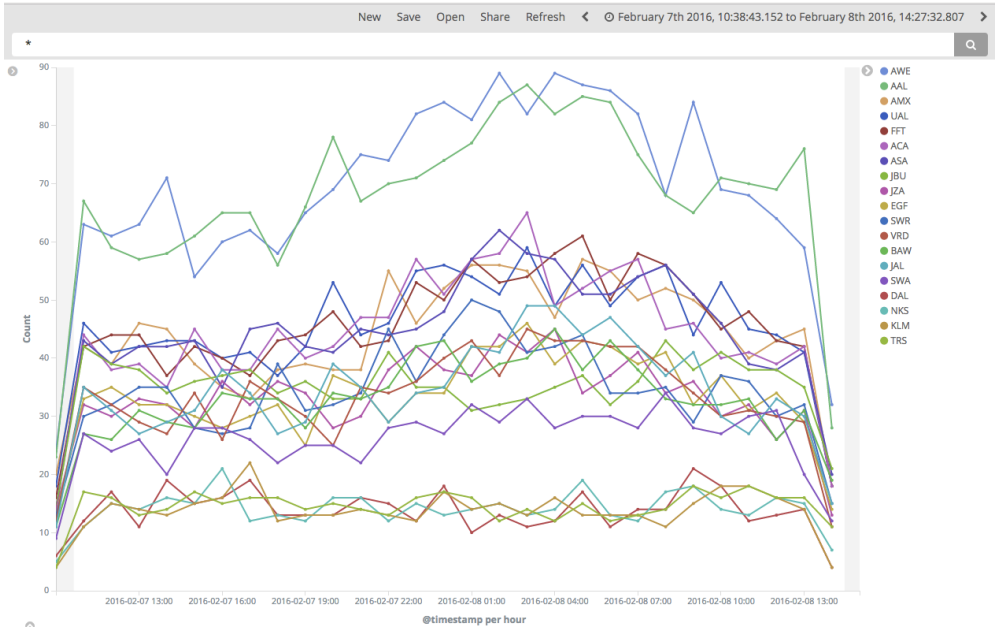
elastic

What is “Normal”?

In general, this question can be answered in two ways:

1) Something behaves in a consistent way with respect to itself, over time

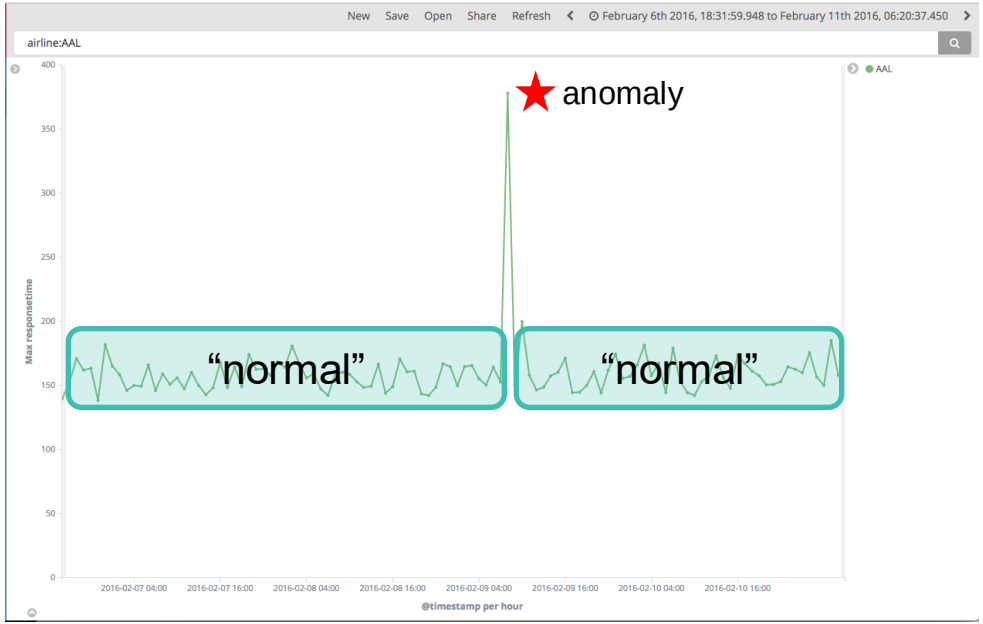
2) Something behaves in a consistent way compared against similar entities



elastic

What is “Abnormal”?

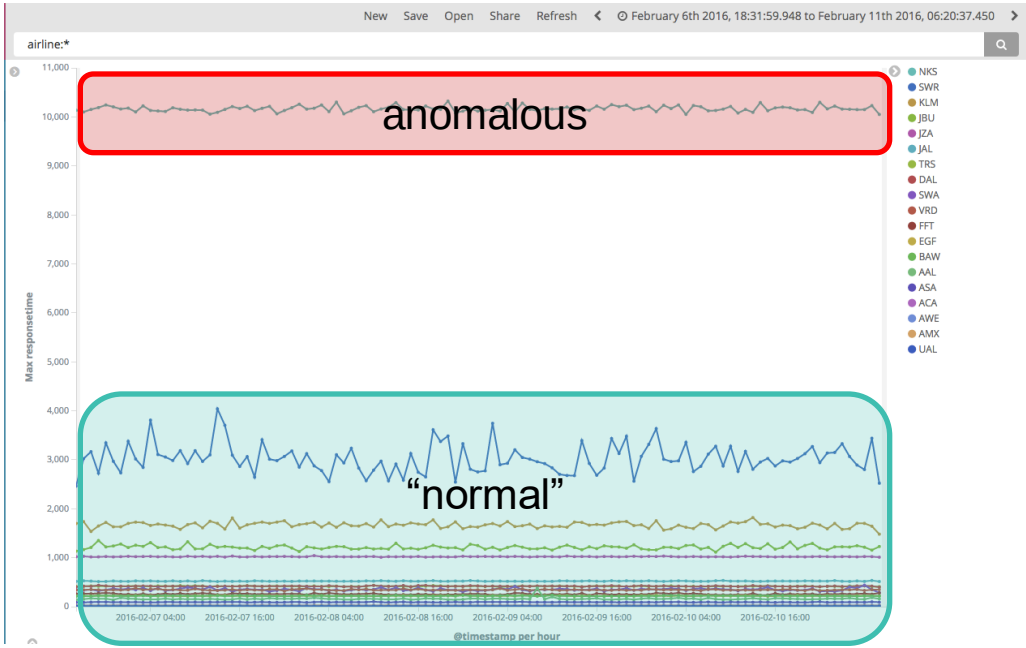
1) If something changes its behavior, compared to its own history – that change is ***anomalous***.



elastic

What is “Abnormal”?

2) If something is drastically different that others within a population, then that entity is ***anomalous***.

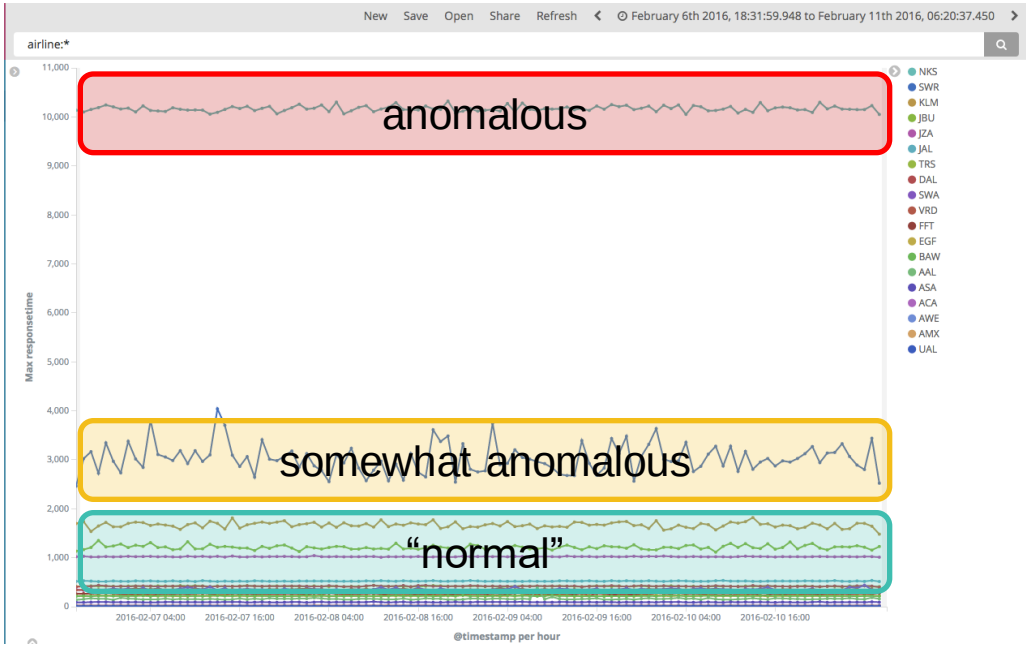


elastic

What is “Abnormal”?

2) If something is drastically different that others within a population, then that entity is ***anomalous***.

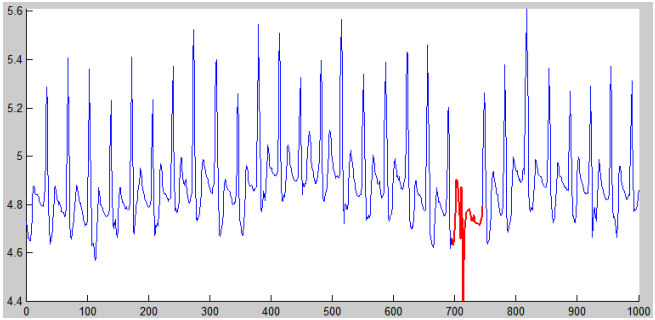
There’s also the concept of being “somewhat anomalous”



elastic

In Summary, Anomalousness is:

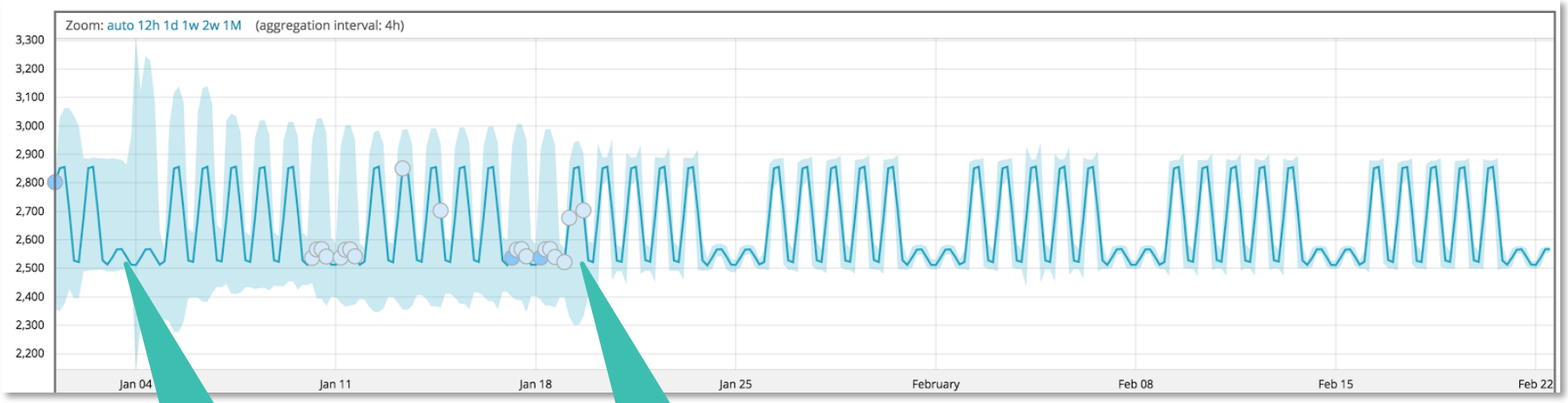
1) When an entities' ***behavior changes*** significantly and suddenly



2) When an entity is drastically ***different than others*** within a population

elastic

Also, model needs to account for any periodicity



After 2 full days, daily periodicity has been learned.

After 2 full weeks, weekly periodicity has been learned.

elastic

Elastic ML Framework

elastic

이상징후 탐지에 필수적인 16개의 알고리즘을 3 가지 방법론으로 구성해서 제안 – Elastic ML Framework

X축 : Single Metric, Multi Metric Time series – 과거와 다른 행동패턴(by)

Y축 : Population Analysis – 다른 것들과 비교해서 다른 행동패턴(over)

Z축 : Rare/Unusual Analysis – 보기 드문 행동패턴(rare)

* 몇 십년 경험을 가진 시스템 아키텍트/관리자 및 보안전문가의 노우하우(Know-How)를 시뮬레이션

elastic

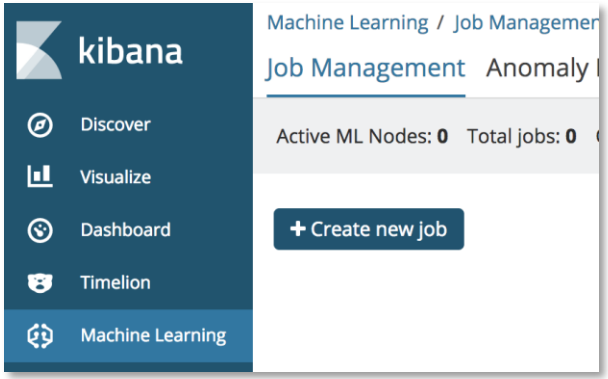
Lab 1: The Single Metric Job

Goal: Detect anomalies in a single time series

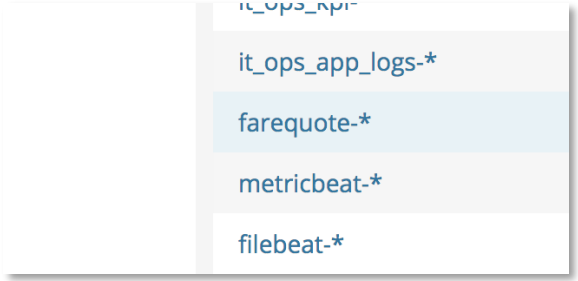
elastic

Steps to Complete

1) In Machine Learning, Create new job



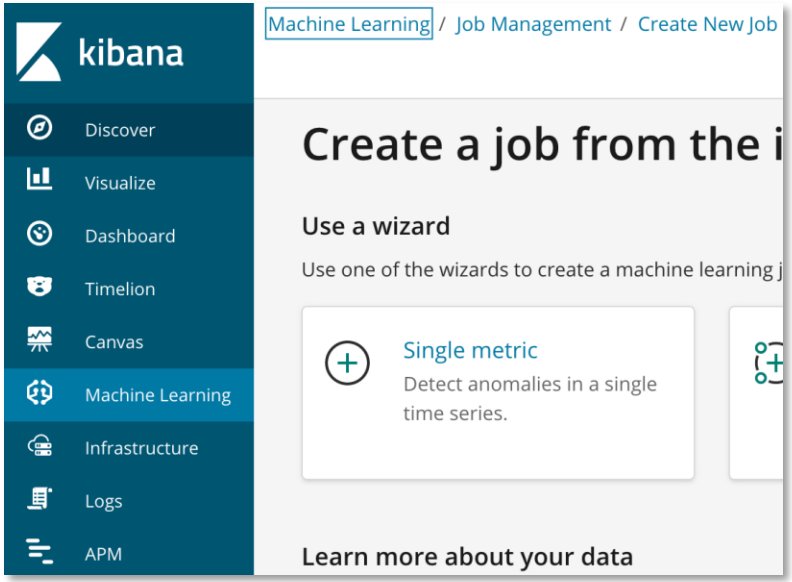
2) Choose the “farequote-*” index pattern



elastic

Steps to Complete

3) pick the Single Metric Wizard



elastic

Steps to Complete

4) Aggregation: count

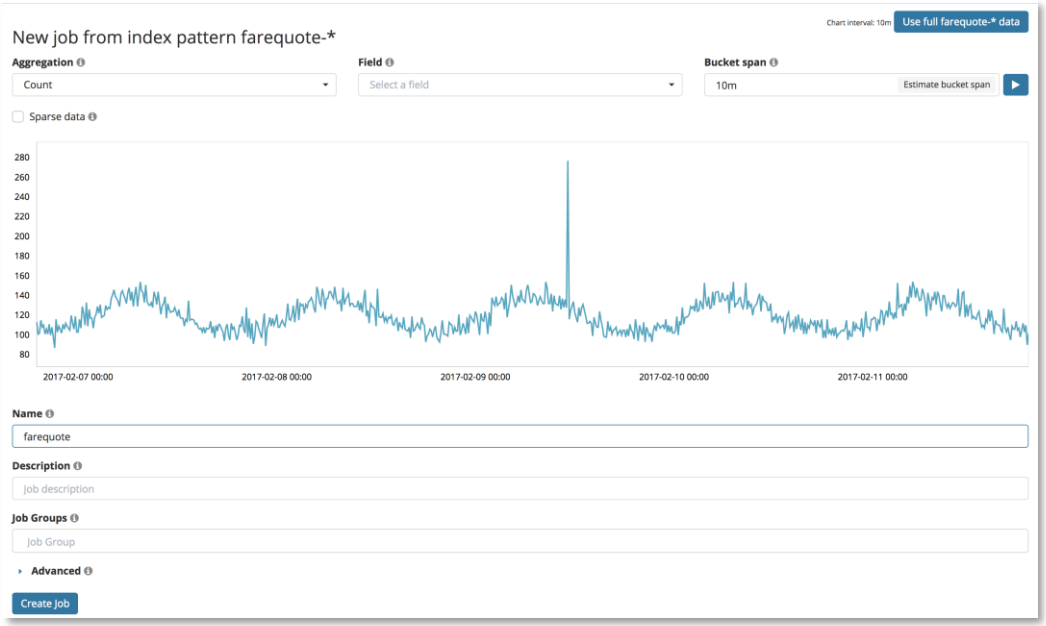
5) Field: <leave blank>

6) Bucket span: 10m

7) Click the “use full farequote data” button

8) Name: “farequote”

9) click “Create Job”



elastic

Steps to Complete

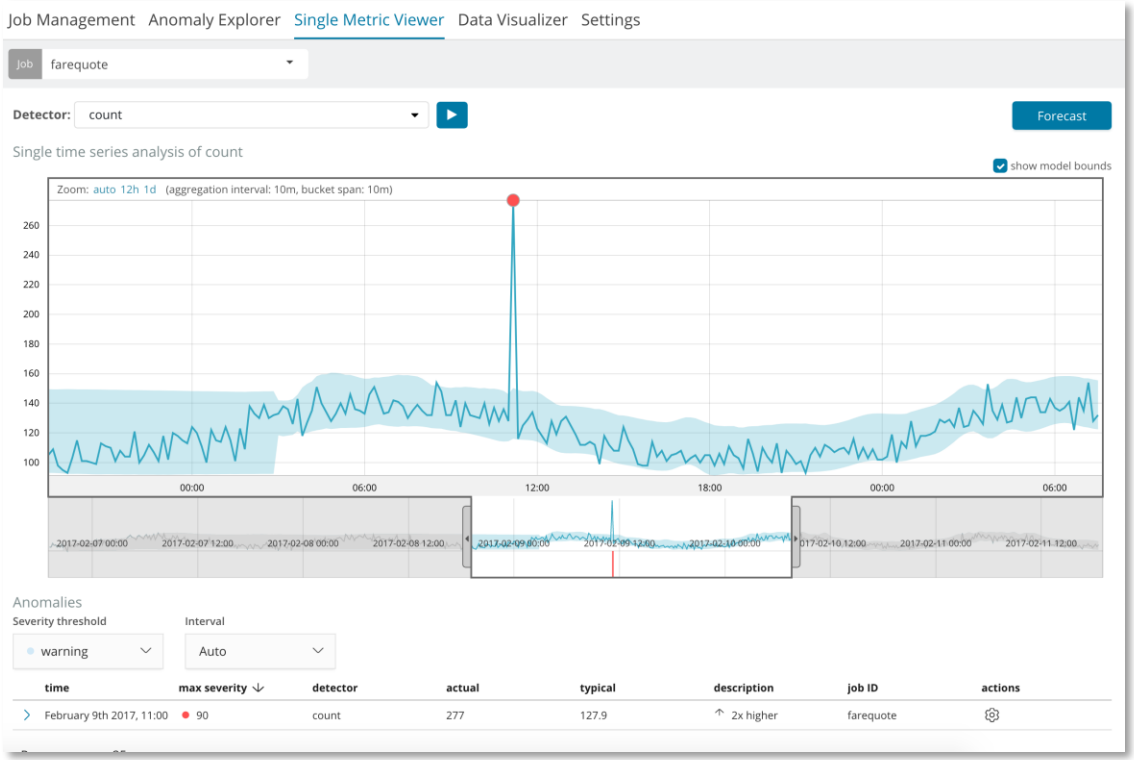
10) See animated learning

11) click “View Results”

elastic

Steps to Complete

12) Zoom in on anomaly



Splitting the Analysis

Feature Selection and Selection

- Which attributes of a this data could be used to judge its unusualness?

raw logs

```
2016/02/08 06:20:43 INFO  [http-8680]: FareQuoteImpl - FareQuoteImpl.getFare(AAL): exiting: 92.5638
2016/02/08 06:20:44 INFO  [http-8680]: FareQuoteImpl - FareQuoteImpl.getFare(JZA): exiting: 990.4628
2016/02/08 06:20:46 INFO  [http-8680]: FareQuoteImpl - FareQuoteImpl.getFare(JBU): exiting: 877.5927
...
```

ingest

document


```
{
  "_index": "farequote",
  "_type": "response",
  "_id": "AVNQ1__XRcuaRiYtw-jH",
  "_score": 3.290889,
  "_source": {
    "sourcetype": "farequote",
    "airline": "AAL",
    "responsetime": "92.5638",
    "time": "2016-02-08T06:20:43+0000"
  }
}
```


“airline” (categorical)

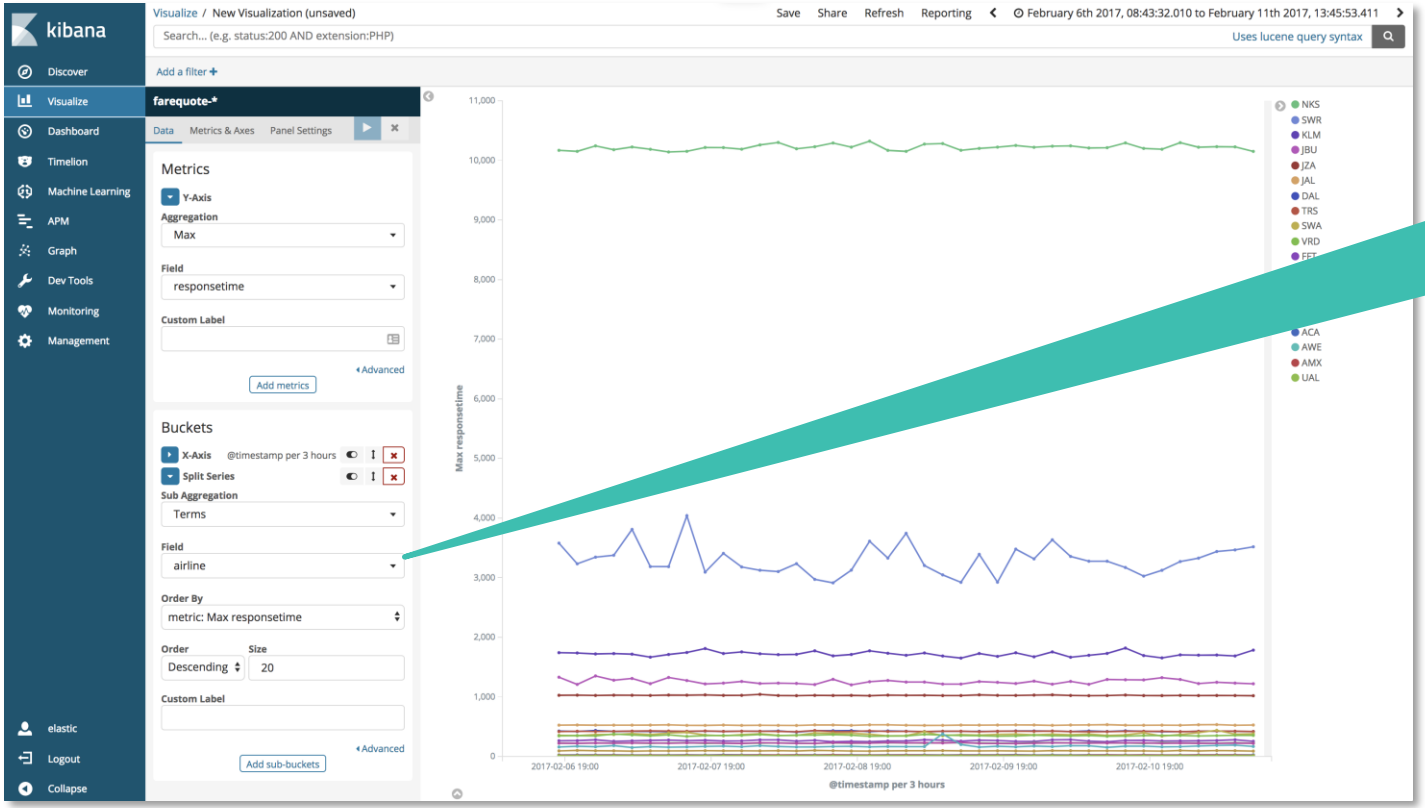
“responsetime” (metric)

elastic

What Kinds of Questions Can be Answered?

QUESTION	ANSWERABLE?
Is there an unusual ***amount*** of requests per unit time (total)?	Yes
Is there any particular airlines with unusual ***amounts*** of requests per unit time?	Yes
Is the ***total*** response time of ***all*** API calls unusually long?	Yes
Is the response time of API calls ***per airline*** unusually long?	Yes
Are there any airlines with excessive take-off delays?	No

In Kibana: How to Answer that Question



ML has the same notion of "splitting" along a categorical field

elastic

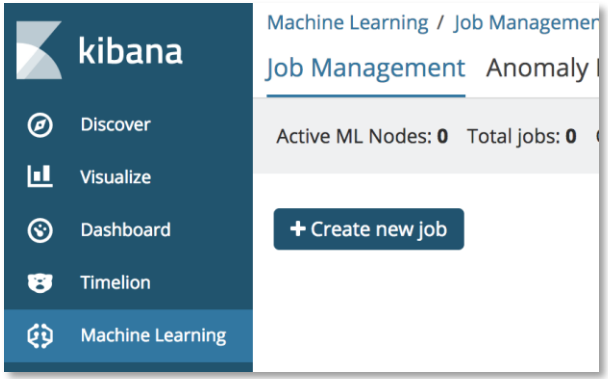
Lab 2: Multi-Metric Jobs

Goal: Detect anomalies in parallel time series

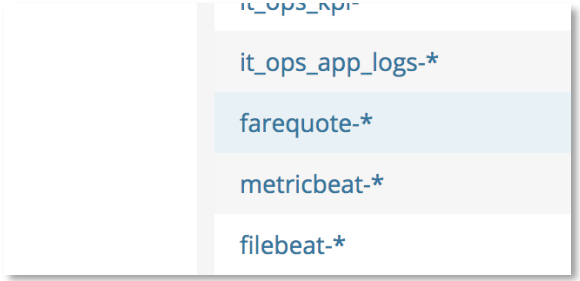
elastic

Steps to Complete

1) In Machine Learning, Create new job



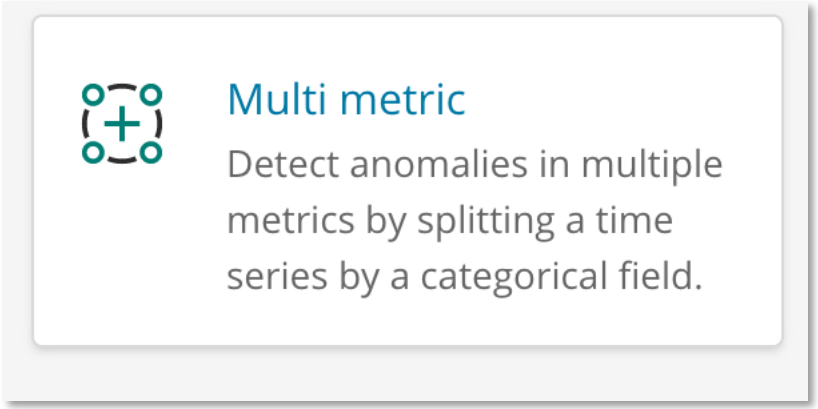
2) Choose the “farequote-*” index pattern



elastic

Steps to Complete

3) pick Multi metric Job Wizard



elastic

Steps to Complete

4) choose
- event rate, count
- responsetime, max

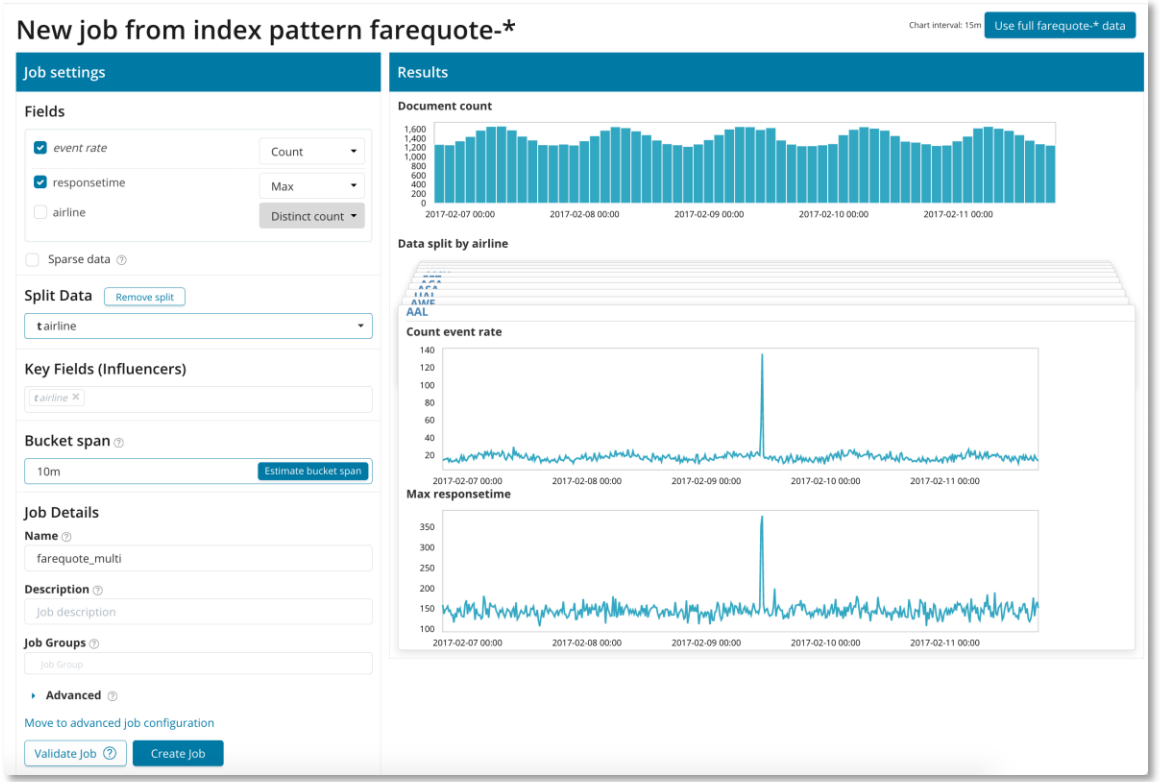
5) bucket span: 10m

6) Split Data: airline

7) click “use full farequote data”

8) Name: “farequote_multi”

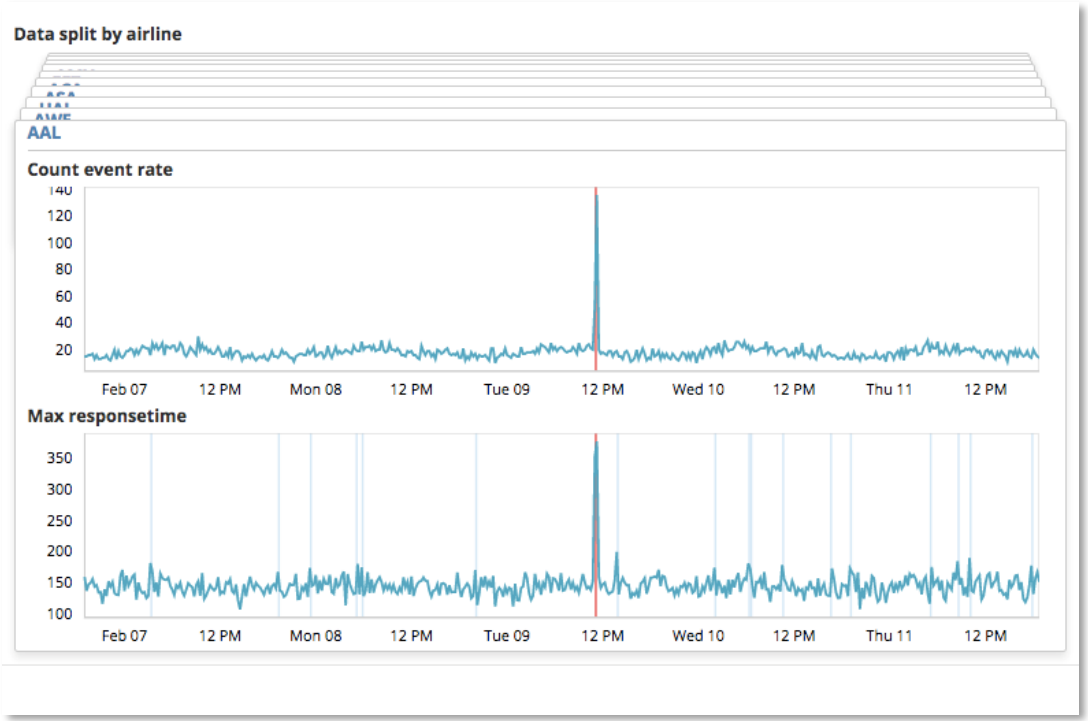
9) click “Create Job”



Steps to Complete

10) See animated learning

11) click “View Results”

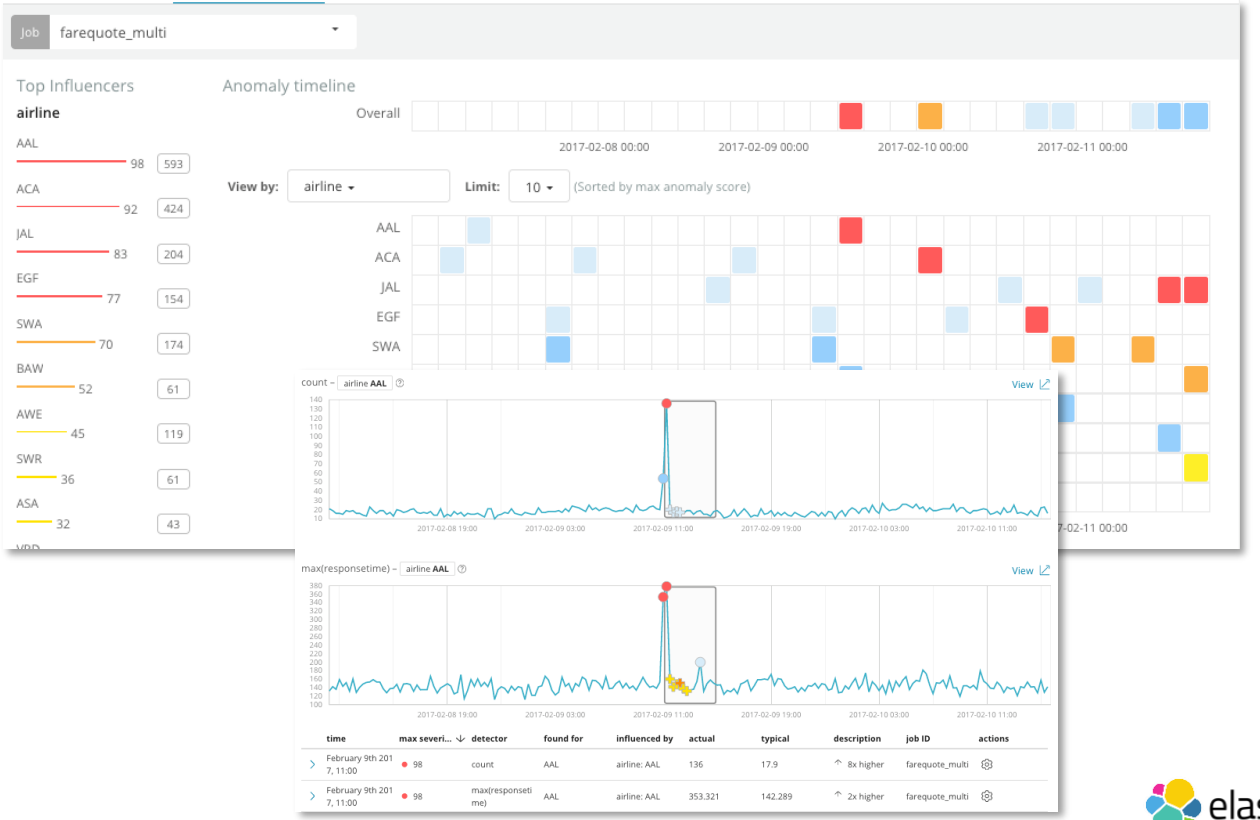


elastic

Steps to Complete

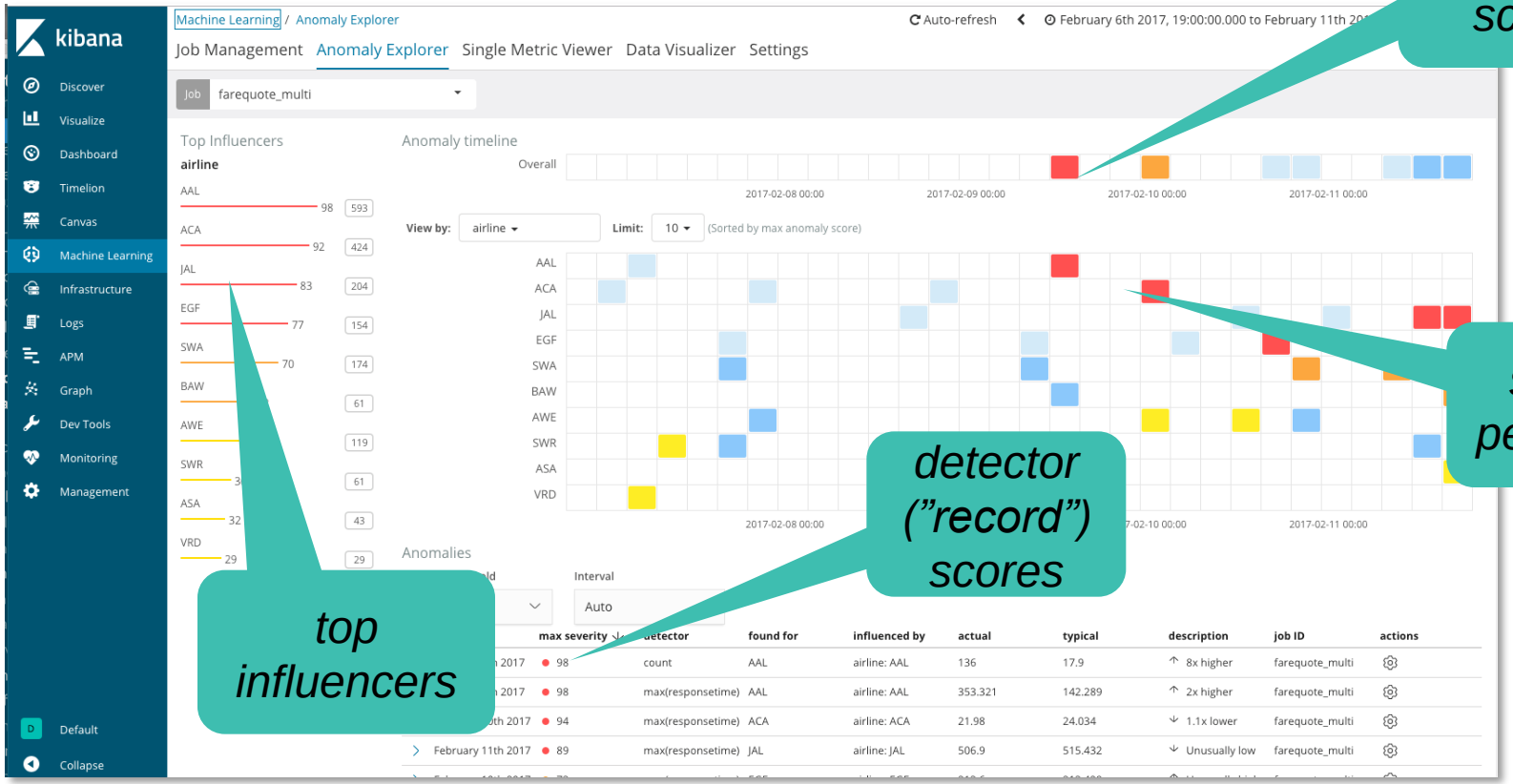
12) Result:

top unusual airline: AAL anomalies
in both count
and response time



The Anomaly Explorer View

Anomaly Explorer



elastic

Concept: What is an Influencer?

- An Influencer is a field, selected at configuration time, that would be a logical entity ”to blame” if an anomaly were to exist
- Doesn’t have to be a field in the actual detector, but fields used to split the data are often good candidates
- Will get its own score based upon how influential that entity is on the anomaly

Scoring

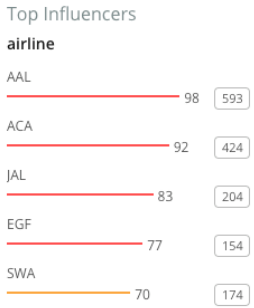
- Overall Job score is 90
  - How unusual is that bucket, given all airlines?
- Detector scores are 98
  - How unusual is the response time and count of airline=AAL?
- airline=AAL is the top influencer in this time range
  - 98 is the max score
  - 593 is the sum of all scores over the time range

Anomalies

Severity threshold: warning

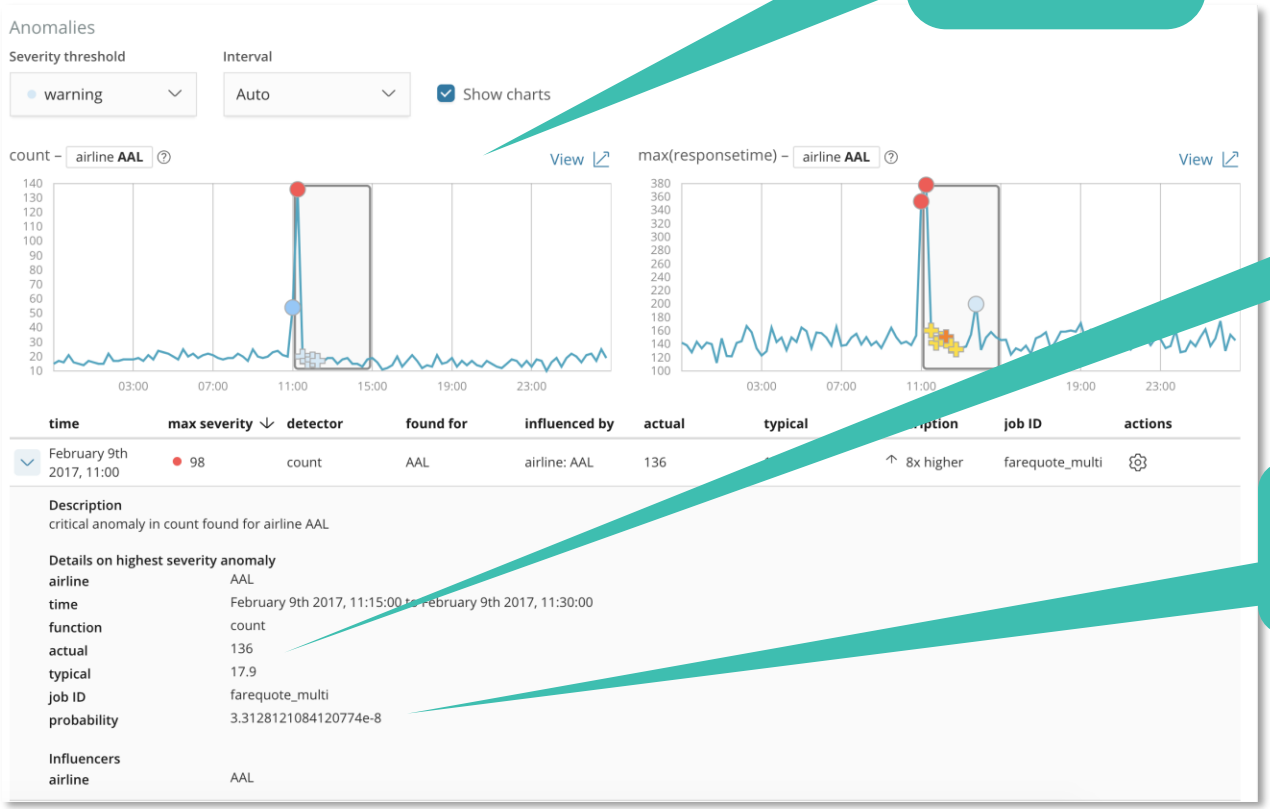
Interval: Auto

time	max severity	detector	found for	influenced by	actual	typical	description	job ID	actions
February 9th 2017	98	count	AAL	airline: AAL	136	17.9	8x higher	farequote_multi	
February 9th 2017	98	max(responsetime)	AAL	airline: AAL	353.321	142.289	2x higher	farequote_multi	



elastic

Anomaly Details



view individual anomalies

actual vs. "typical"

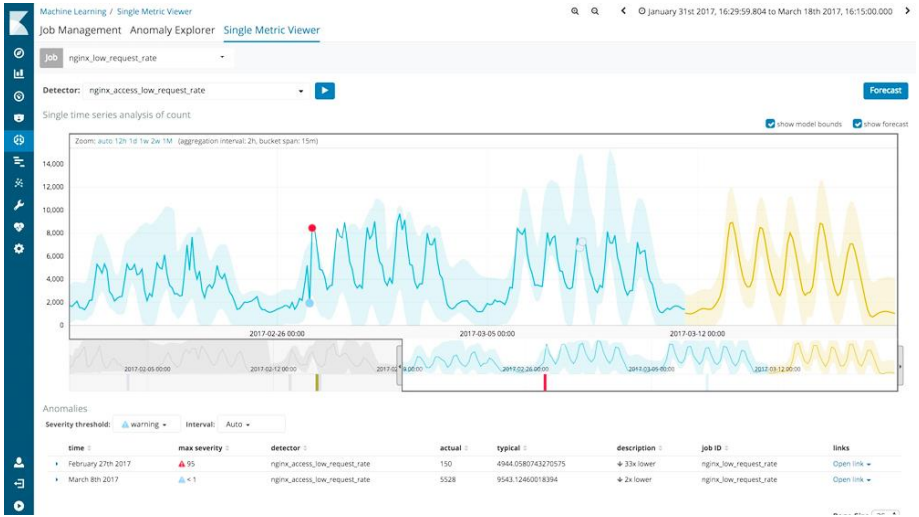
raw probability

elastic

Forecasting

elastic

Extrapolate into the future with forecasting



Example use cases

- I want to know when capacity will exceed 80%

- What is my expected visitor count next Saturday at 8am?

elastic

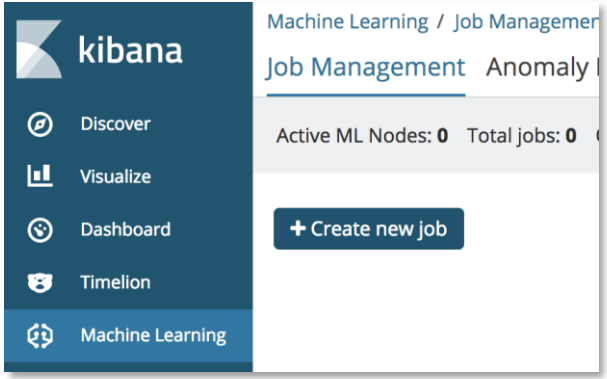
Lab 3: Forecasting

Goal: Extrapolate into the future

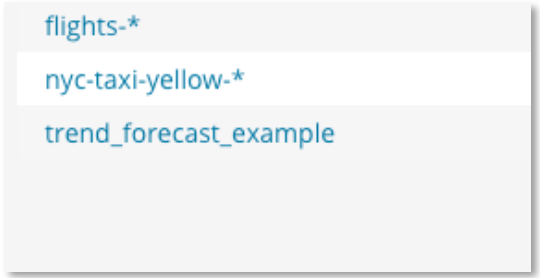
elastic

Steps to Complete

1) In Machine Learning, Create new job



2) Choose the “trend_forecast_example” index pattern



elastic

Steps to Complete

3) single metric job

4) sum of field “amount”

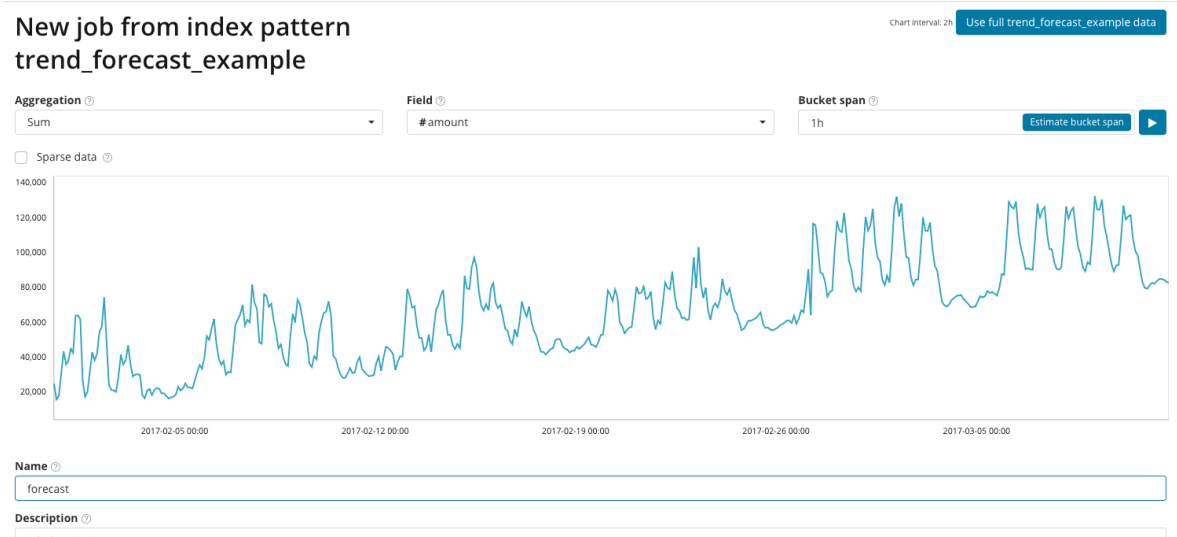
5) bucket span: 1h

6) click “use full data”

7) Name: “forecast”

8) click “Create Job”

9) click “View Results”



elastic

Steps to Complete

10) click “Forecast” button

11) Enter “7d” for duration

12) click “Run” button

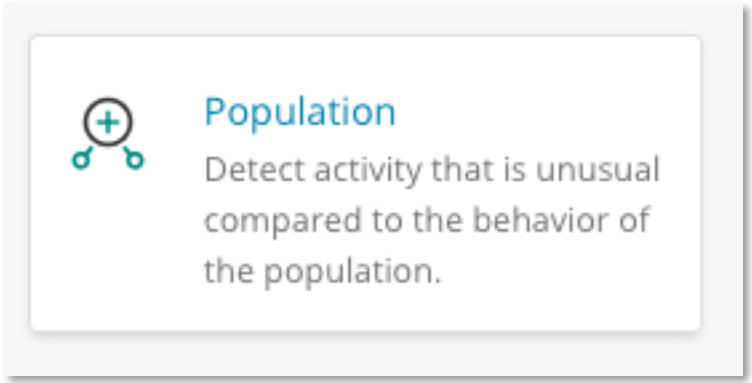
13) see forecast in UI
(results also available via API)

elastic

Population Analysis

Population Analysis

- Use when:
  - Compare entities against peers
  - Not against own history
- Helpful when:
  - Entities have high-cardinality (i.e. external IP addresses)
  - Data for specific entities may be sparse in time (individual customers placing orders)
  - The behavior of the population as a whole is mostly homogeneous
- Not appropriate when:
  - Members of the population have vastly different behavior inherently.



elastic

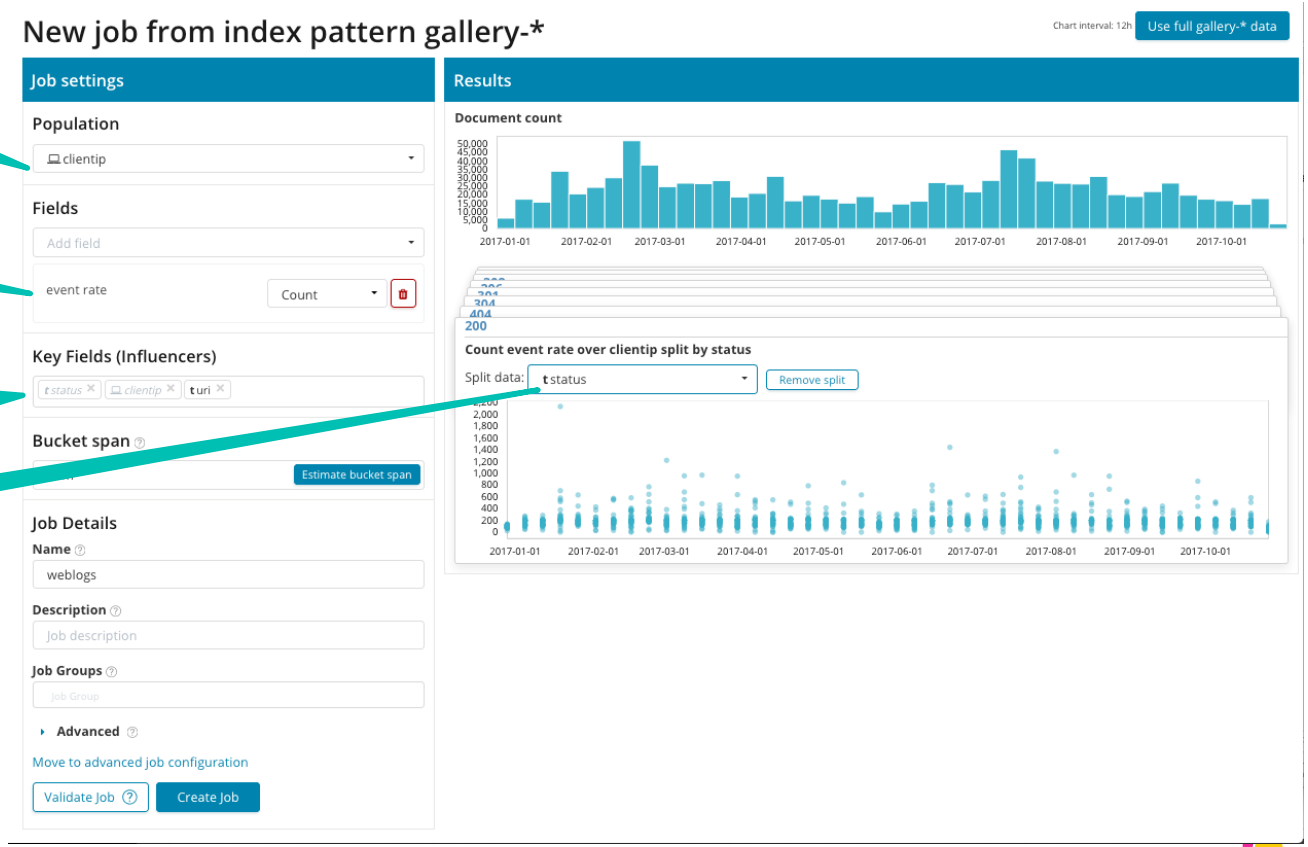
Population Analysis

population field

the detector

influencers

optional split



elastic

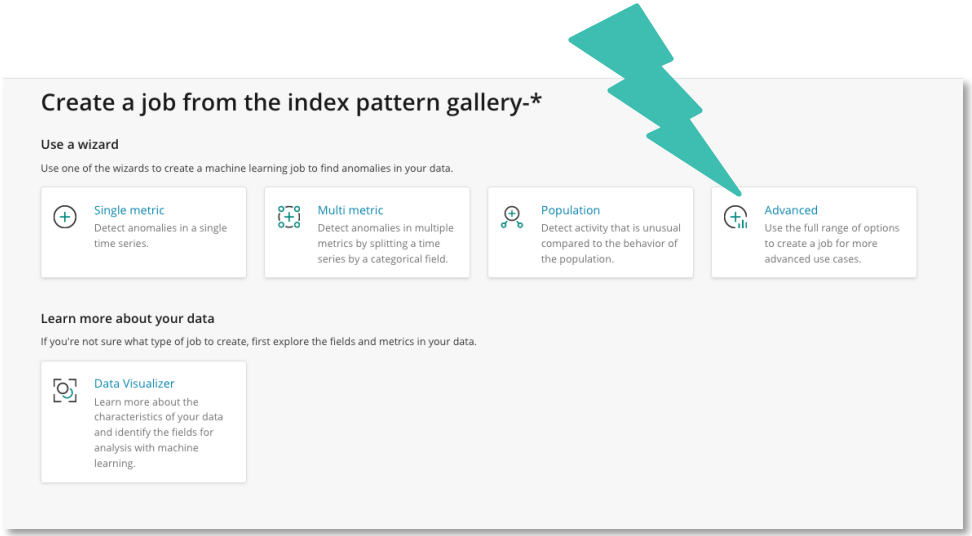
Population Analysis

clientip: 173.203.78.60
status: 404
uri: /wp-login.php

Advanced Jobs

Advanced Jobs

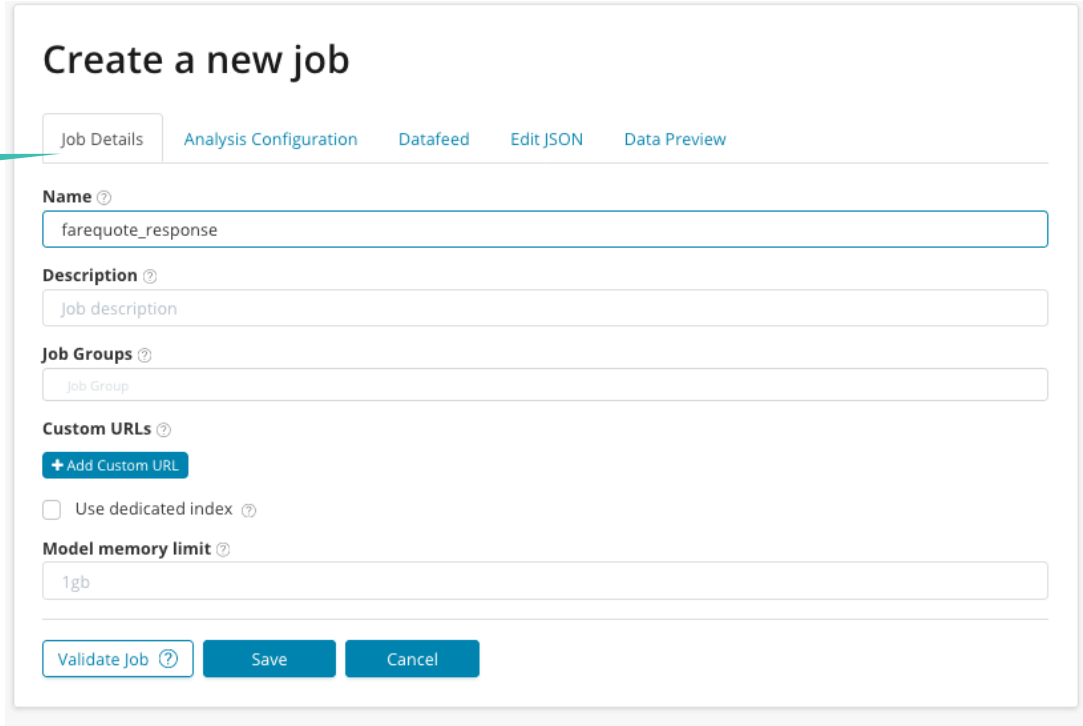
- Gives more flexibility
- Access to additional capabilities
- Requires more understanding of configuration parameters



elastic

Configuration of Advanced Job

Tabs of different information

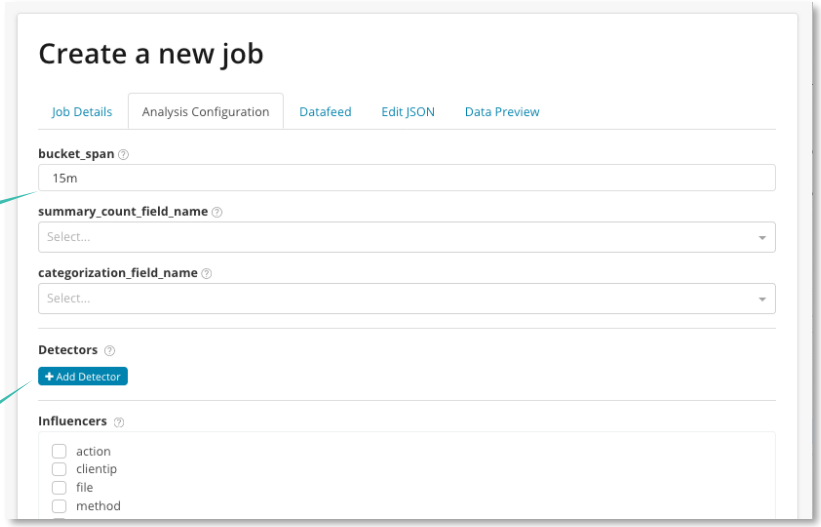


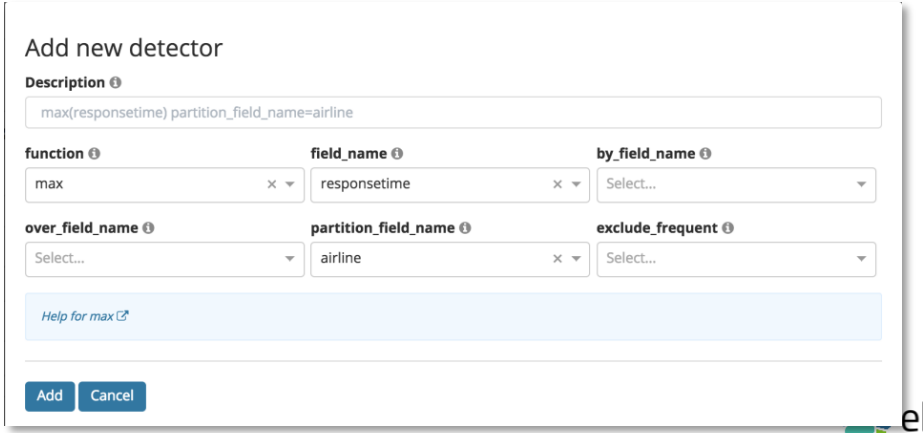
elastic

Analysis Configuration

Settting of bucket_span

Creation of "detector"

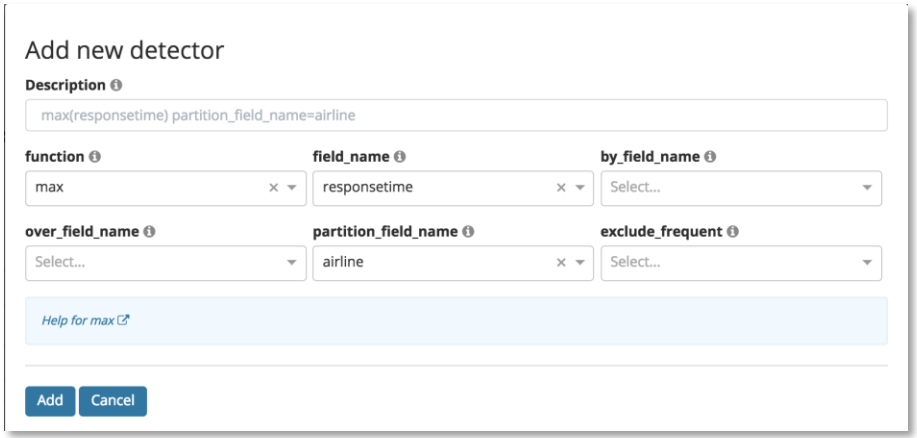




elastic

Detector Details

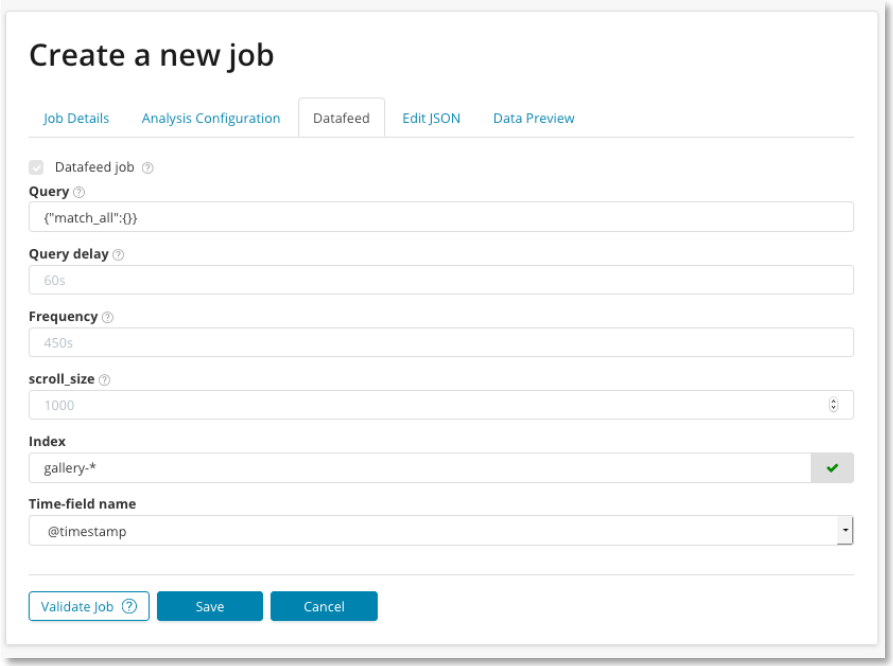
- **function** – min, max, mean, etc.
- **field_name** –field **function** “operates on”
- **by_field_name** –“dependent” split
- **partition_field_name** –“independent” split
- **over_field_name** – defines a population



elastic

Datafeed Details

- Datafeed manages how ML queries Elasticsearch
- See ML docs for configuration options



elastic

Rarity Analysis

Rare Analysis

- Finding items that rarely occur is also often useful
  - Rarely occurring log messages
  - Rare running process names
  - Rare connection destinations
- ML has a rare function, but it should be noted that:
  - It is relative, i.e. it takes into account the frequency of other field values

A,B,A,B,A,C,B,A,B,A,C,A,B,C,A,C,B,A,C,B,A,C,X <=X is rare
A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X <= X is not rare

  - Therefore it works best when there are plenty of routine messages to contrast the rare ones

Example of Rare Analysis

- Use Case: Security team @ services company
- Wanted to profile typical processes on each host using netstat

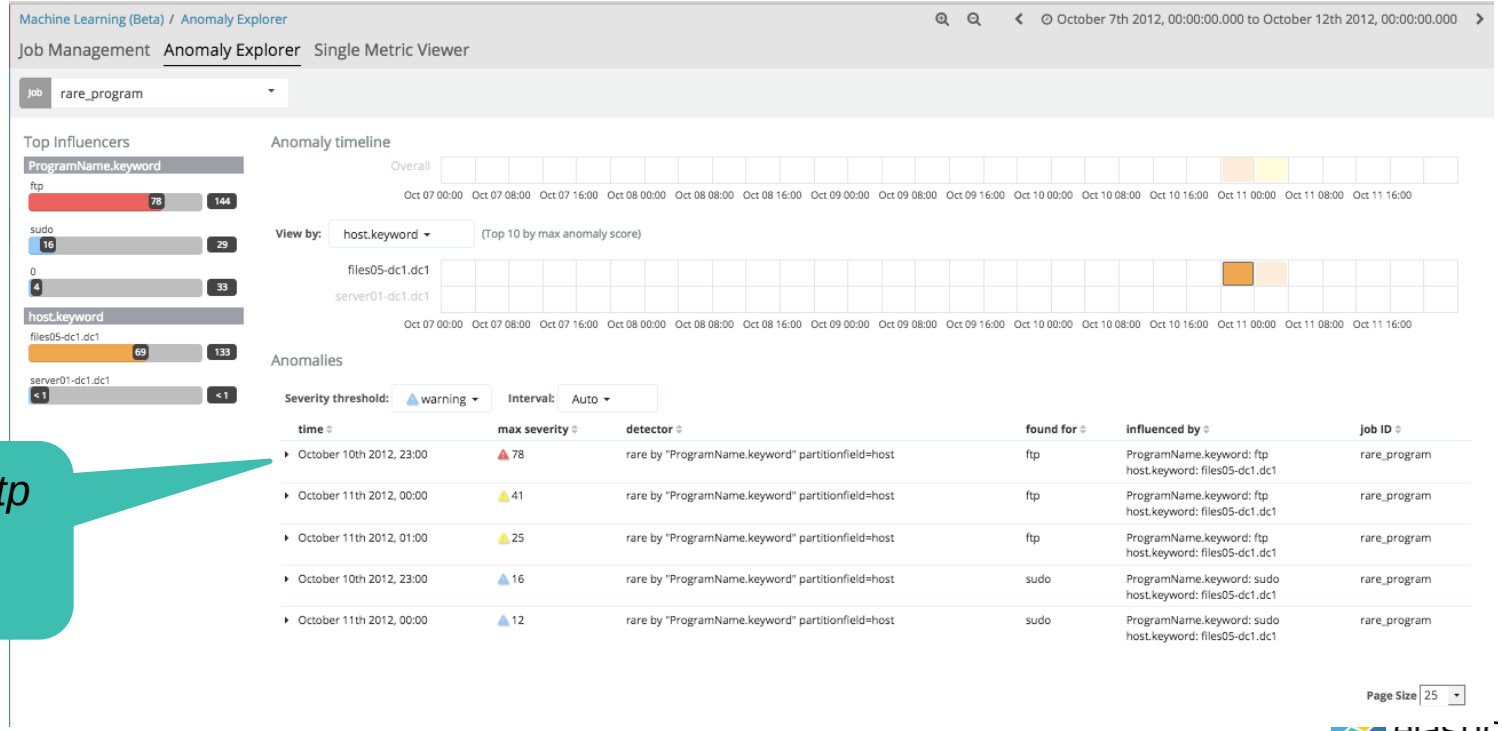
```
Active Internet connections (servers and established)
(index=netstat host="ids01-dc2" State=LISTEN (7/16/13 1:32:51AM))
Proto Recv-Q Send-Q Local Address               Foreign Address             State       PID/Program name
tcp        0      0  10.220.174.41:561         0.0.0.0:*                   LISTEN      8776/argus
tcp        0      0  0.0.0.0:22                0.0.0.0:*                   LISTEN      2033/sshd
tcp        0      0  0.0.0.0:1241              0.0.0.0:*                   LISTEN      4472/nessusd
tcp        0      0  0.0.0.0:8089              0.0.0.0:*                   LISTEN      4238/splunkd
tcp        0      0  127.0.0.1:25              0.0.0.0:*                   LISTEN      4194/master
tcp        0      0  0.0.0.0:8000              0.0.0.0:*                   LISTEN      4361/python
tcp        0      0  0.0.0.0:8834              0.0.0.0:*                   LISTEN      4472/nessusd
tcp        0      0  127.0.0.1:199             0.0.0.0:*                   LISTEN      2022/snmpd
```

- Goal was to identify rare processes that “start up and communicate” for each host, individually

elastic

Example of Rare Analysis

detector: *rare by ProgramName partition_field=host*



ProgramName: ftp
host: files05
rarely occurs

ML Categorization

Categorization

- What is it?
  - Automatically grouping similar log messages into the same “category”
- How?
  - Via an algorithm that looks at string similarity and clusters similar log lines
- Why?
  - Bring structure to semi-structured data so that it can be analyzed

Benefit: Find anomalies in logs without knowing what they contain

Example

```
Oct 22 18:17:58 localhost sshd[8903]: Invalid user admin from 41.43.112.199 port 41805
Oct 22 18:17:58 localhost sshd[8903]: input_userauth_request: invalid user admin [preauth]
Oct 22 18:17:59 localhost sshd[8903]: Connection closed by 41.43.112.199 port 41805 [preauth]
Oct 22 20:58:03 localhost sshd[2074]: Received disconnect from 72.93.85.203 port 53552:11: disconnected by user
Oct 22 20:58:03 localhost sshd[2074]: Disconnected from 72.93.85.203 port 53552
Oct 22 20:58:03 localhost sshd[2072]: pam_unix(sshd:session): session closed for user ec2-user
Oct 22 21:32:54 localhost sshd[8944]: pam_unix(sshd:session): session opened for user ec2-user by (uid=0)
Oct 22 21:35:15 localhost runuser: pam_unix(runuser-l:session): session closed for user ec2-user
Oct 22 21:35:15 localhost runuser: pam_unix(runuser-l:session): session opened for user ec2-user by ec2-user(uid=0)
Oct 22 21:35:16 localhost runuser: pam_unix(runuser-l:session): session closed for user ec2-user
```

Step 1 – remove mutable text

```
Oct 22 18:17:58 localhost sshd[8903]: Invalid user admin from 41.43.112.199 port 41805
Oct 22 18:17:58 localhost sshd[8903]: input_userauth_request: invalid user admin [preauth]
Oct 22 18:17:59 localhost sshd[8903]: Connection closed by 41.43.112.199 port 41805 [preauth]
Oct 22 20:58:03 localhost sshd[2074]: Received disconnect from 72.93.85.203 port 53552:11: disconnected by user
Oct 22 20:58:03 localhost sshd[2074]: Disconnected from 72.93.85.203 port 53552
Oct 22 20:58:03 localhost sshd[2072]: pam_unix(sshd:session): session closed for user ec2-user
Oct 22 21:32:54 localhost sshd[8944]: pam_unix(sshd:session): session opened for user ec2-user by (uid=0)
Oct 22 21:35:15 localhost runuser: pam_unix(runuser-l:session): session closed for user ec2-user
Oct 22 21:35:15 localhost runuser: pam_unix(runuser-l:session): session opened for user ec2-user by ec2-user(uid=0)
Oct 22 21:35:16 localhost runuser: pam_unix(runuser-l:session): session closed for user ec2-user
```

elastic

Step 2 – cluster similar messages together

Oct 22 18:17:58 localhost sshd[8903]: Invalid user admin from 41.43.112.199 port 41805

Oct 22 18:17:58 localhost sshd[8903]: input_userauth_request: invalid user admin [preauth]

Oct 22 18:17:59 localhost sshd[8903]: Connection closed by 41.43.112.199 port 41805 [preauth]

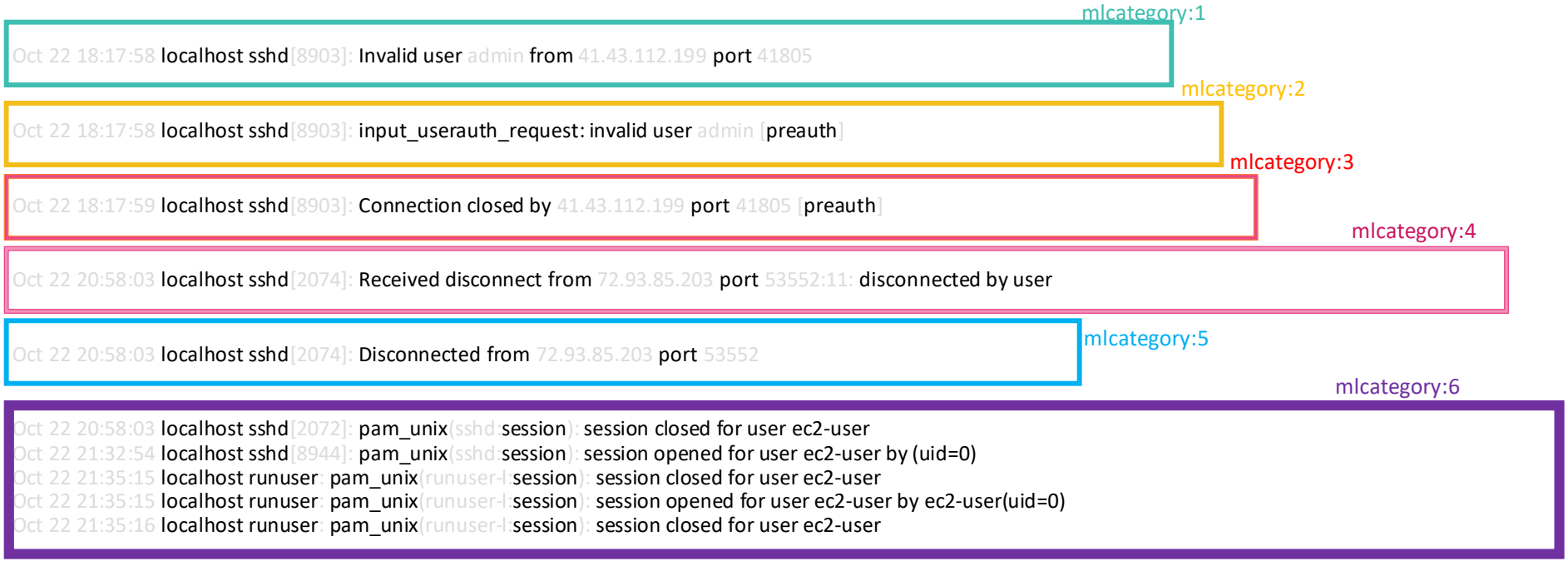
Oct 22 20:58:03 localhost sshd[2074]: Received disconnect from 72.93.85.203 port 53552:11: disconnected by user

Oct 22 20:58:03 localhost sshd[2074]: Disconnected from 72.93.85.203 port 53552

Oct 22 20:58:03 localhost sshd[2072]: pam_unix(sshd:session): session closed for user ec2-user
Oct 22 21:32:54 localhost sshd[8944]: pam_unix(sshd:session): session opened for user ec2-user by (uid=0)
Oct 22 21:35:15 localhost runuser: pam_unix(runuser-l:session): session closed for user ec2-user
Oct 22 21:35:15 localhost runuser: pam_unix(runuser-l:session): session opened for user ec2-user by ec2-user(uid=0)
Oct 22 21:35:16 localhost runuser: pam_unix(runuser-l:session): session closed for user ec2-user

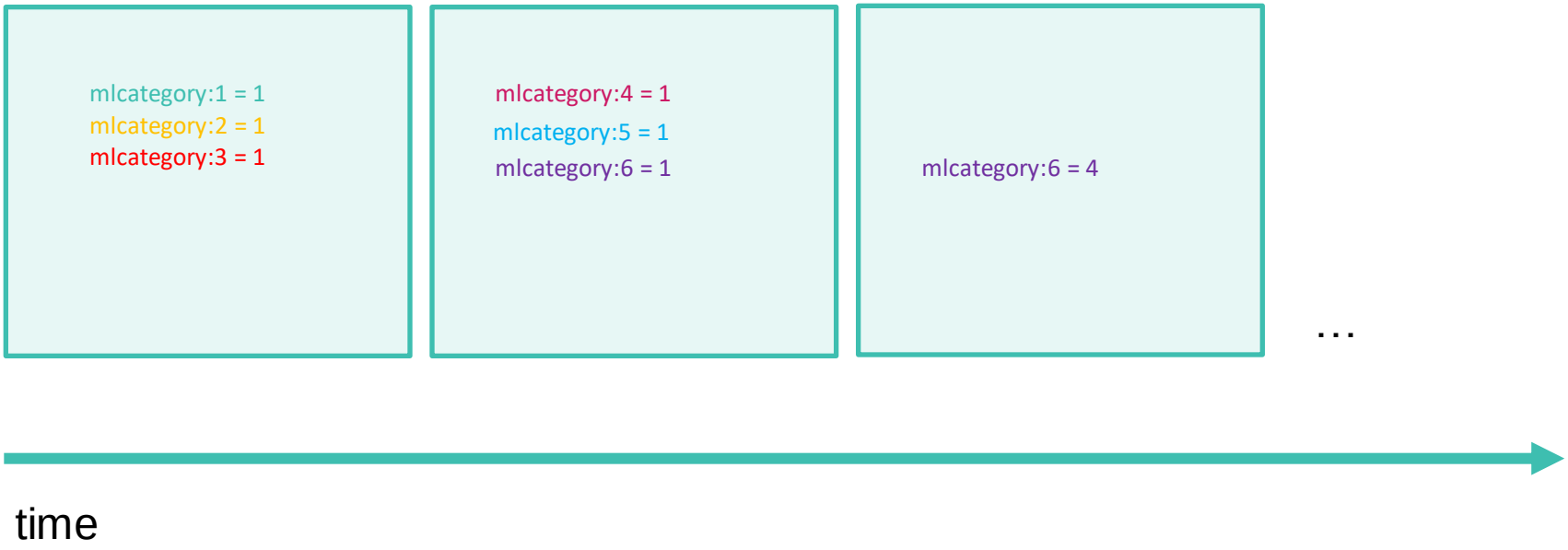
elastic

Step 2 – cluster similar messages together



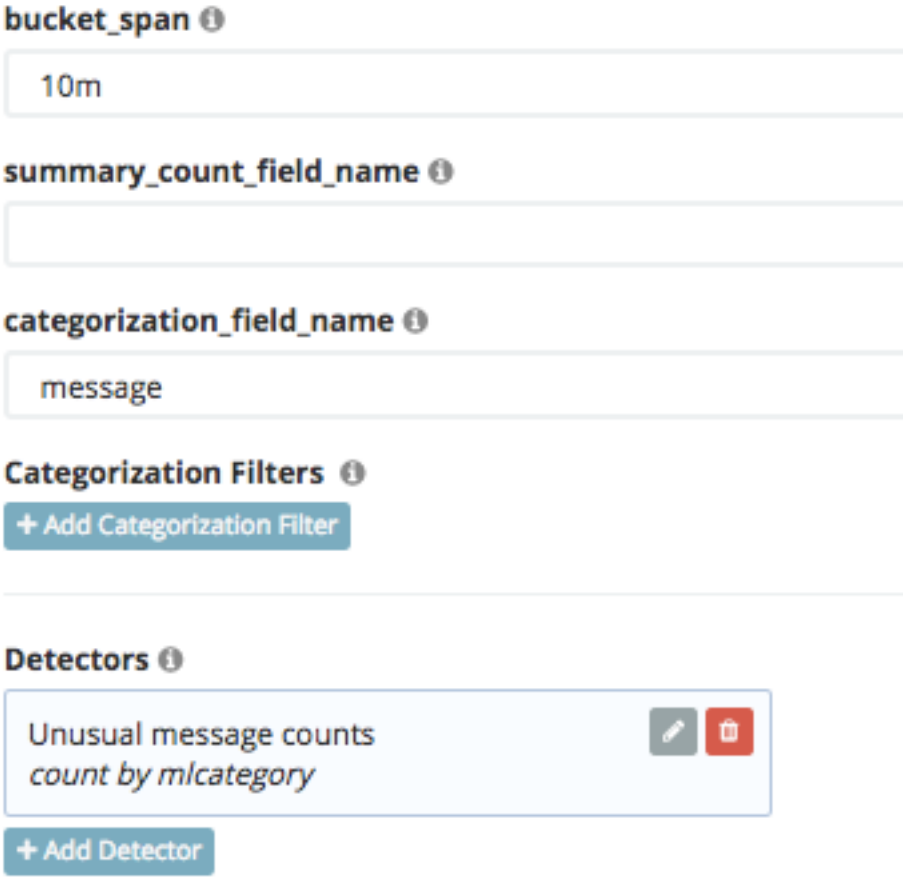
elastic

Step 3 – count per time bucket

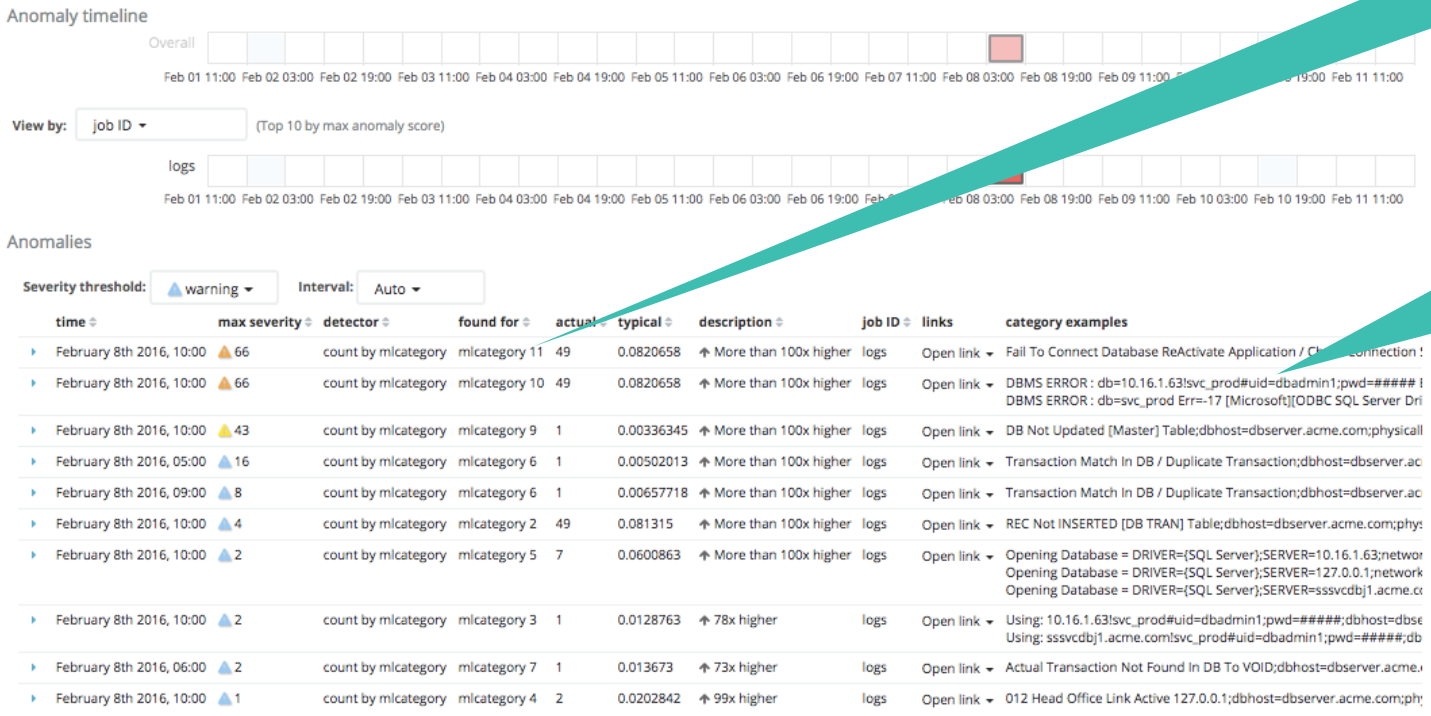


Categorization Example

- Configure an ML job to use:
  - “message” as the categorization_field_name
  - there will be a new, “magic” field called “mlcategory” that is dynamically created by ML to group similar messages together



Categorization Example – count by mlcategory



category name

example matching log messags

elastic

Lab 4, Track 1: Root Cause Analysis

elastic

Lab 4: Track 1 - Problem

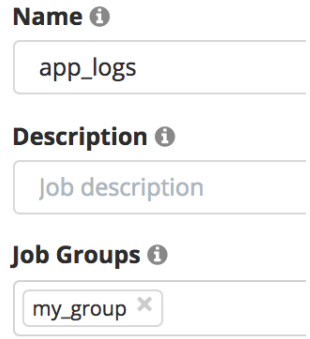
- Transaction processing app records # transactions per minute in an index called “it_ops_kpi-*”. Find the anomaly in it.
- Also, SQL Database and Network performance metrics were gathered for the application environment (it_ops_network-* and it_ops_sql-*). Use multi-metric jobs to see what might be going on in there.
- Lastly, there’s an index (it_ops_app_logs-*) that has the application’s log lines in them. See if there’s anything anomalous in there as well.
- View all jobs correlated together in the Anomaly Explorer

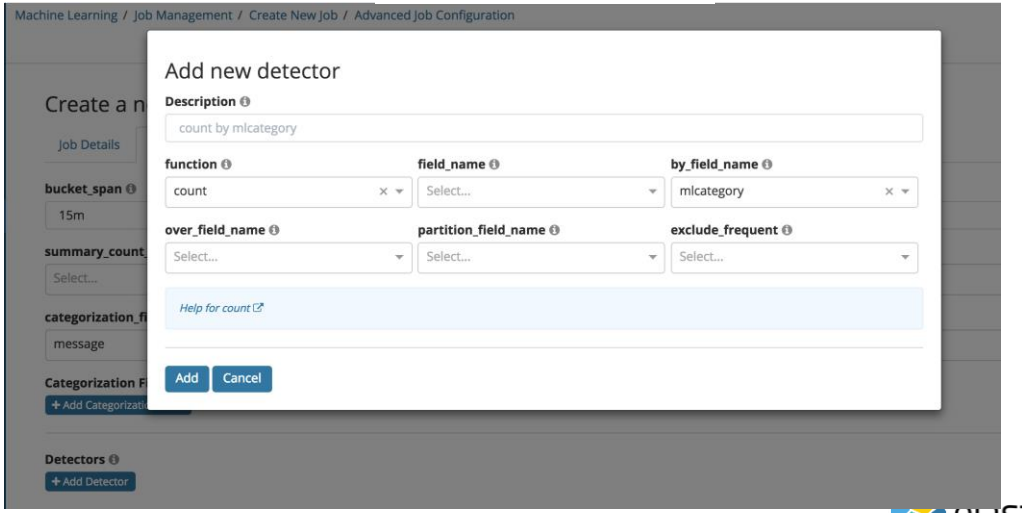
Lab 4: Track 1 - Solution

- Create and Advaned job for:
  - “it_ops_app_logs-*”
  - Create a “count by mlcategory” job for the log events
    - use “message” as the categorization_field_name
- Create Multi-Metric jobs for the following indices:
  - it_ops_sql-*
  - it_ops_network-*
- Create a “low_sum” Single-Metric job for the following :
  - it_ops_kpi-*
- Group all jobs into the same Job Group name
- View all jobs overlaid in the Explorer View

Lab 4: Track 1 - Solution

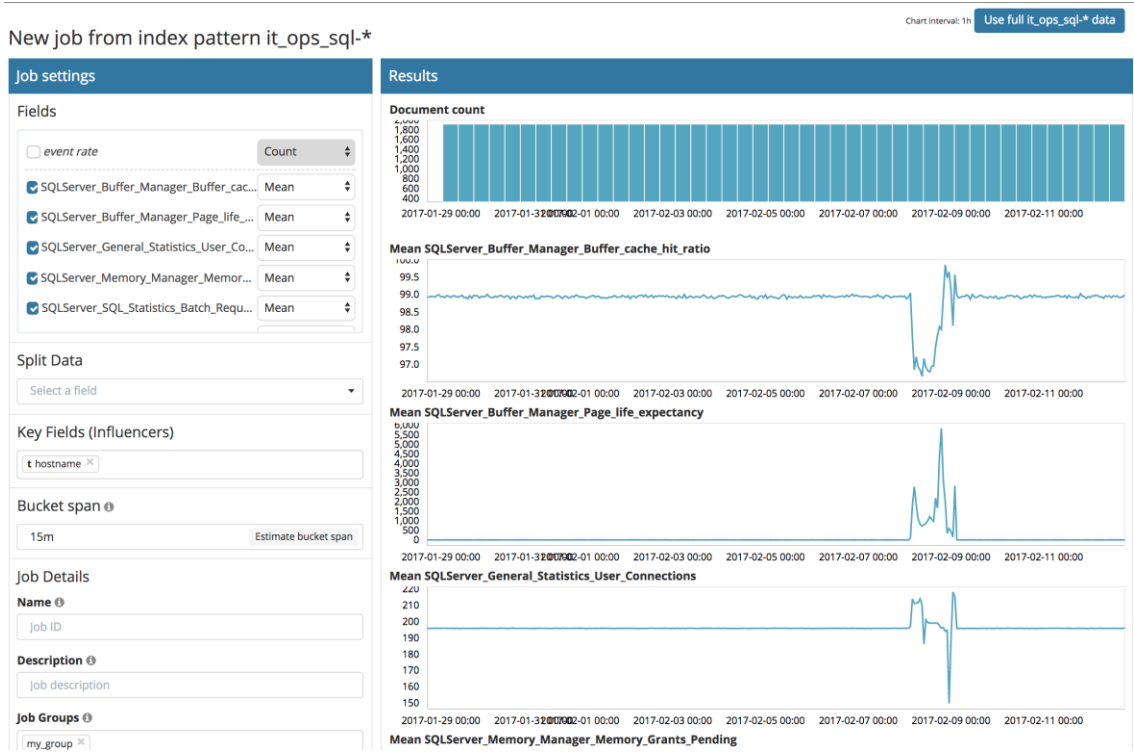
- For index:it_ops_app_logs
  - create an advanced job
  - put job in a group
  - make sure you choose “message” for categorization_field_name
  - detector is: count with by_field_name of “mlcategory”





Lab 4: Track 1 - Solution

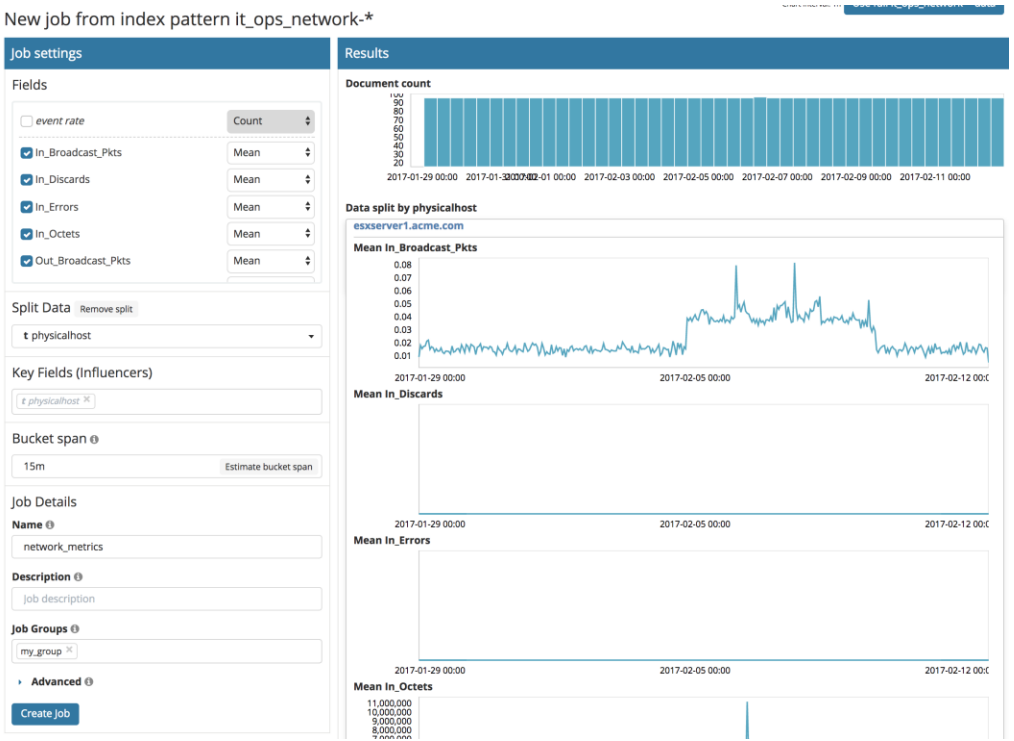
- For index:it_ops_sql-*
  - create multi-metric job
  - “mean” for all SQL metrics
- Also put job in same Job Group as the app logs job



Lab 4: Track 1 - Solution

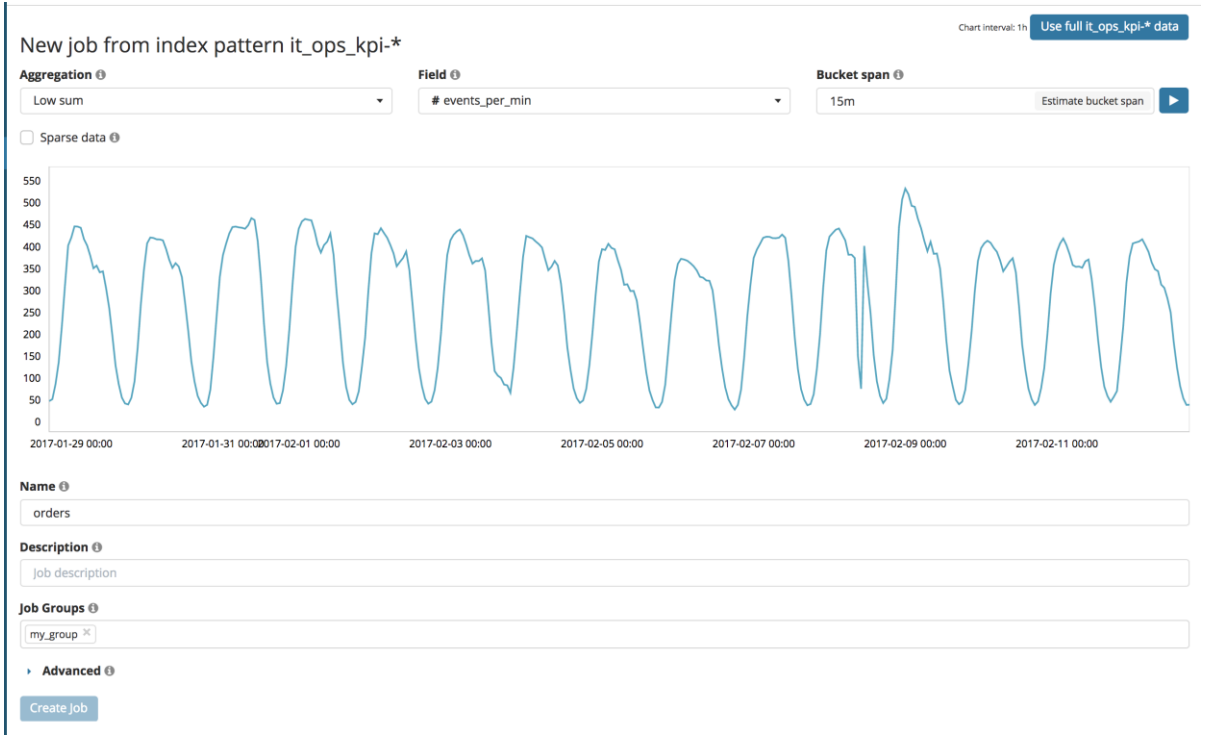
- For index:it_ops_network-*
  - create multi-metric job
  - “mean” for all metrics

- Also put job in same Job Group as the app logs job



Lab 4: Track 1 - Solution

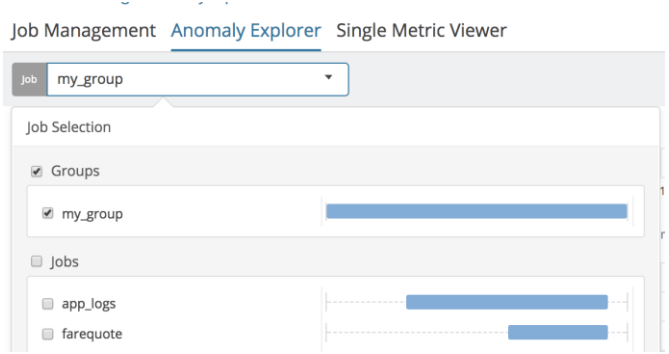
- For index:it_ops_kpi-*
  - create single-metric job
  - “low_sum” for field “events_per_min”
- Also put job in same Job Group as the app logs job



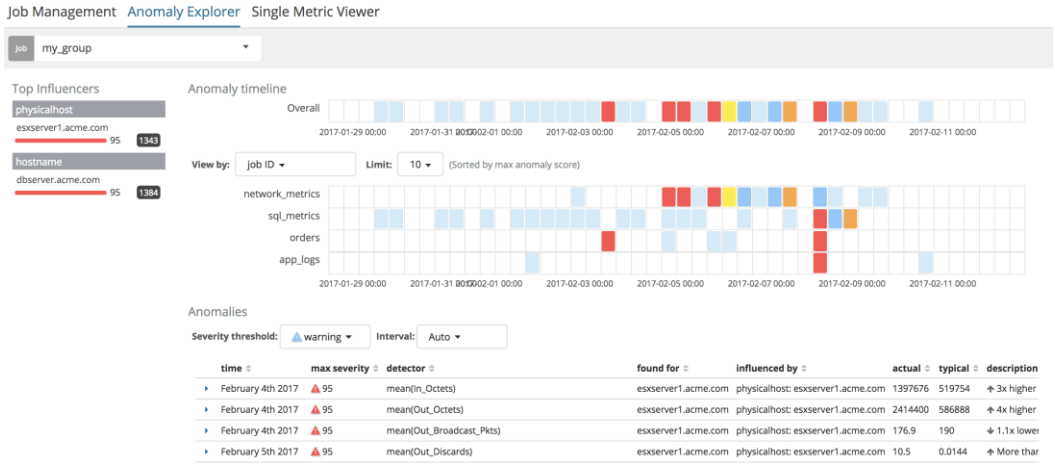
elastic

Lab 4: Track 1 - Solution

- View all jobs in Group



- See correlated anomalies



elastic

Lab 4, Track 2: Analyze Web Logs

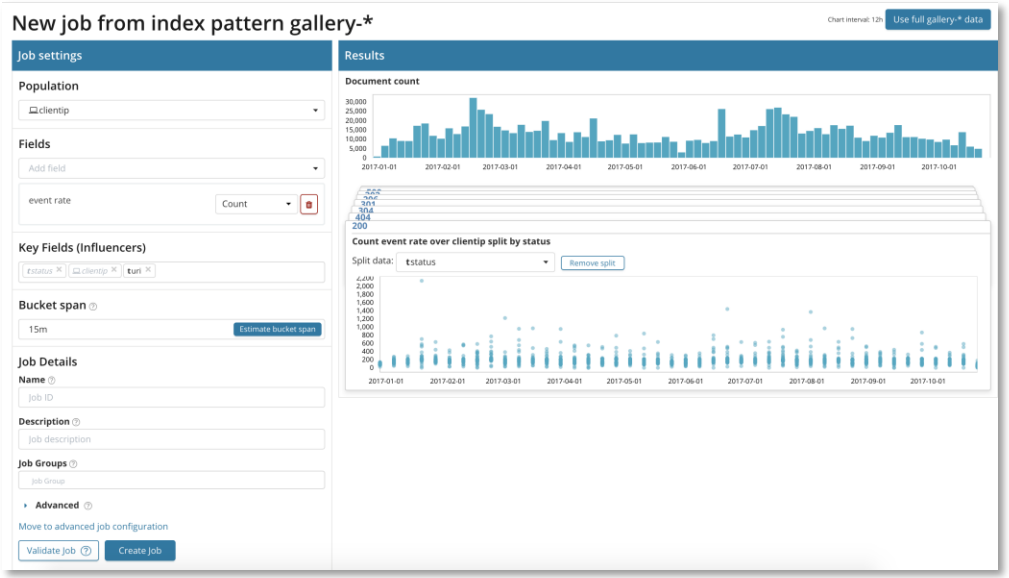
elastic

Lab 4: Track 2 - Problem

- There’s been suspicion that automated bots are slamming the web site from time to time.
- Use a population analysis job to find any rogue IPs and see what URIs were being requested
- Data is in gallery-* index

Lab 4: Track 2 - Solution

- Create and population job for:
  - “gallery-*”
  - count by status over clientIP
  - uri as additional Influencer



Lab 4: Track 2 - Solution

- See rogue IP 173.203.78.60
- Thousands of requests for URI: wp-login.php
- This is a low-intelligence brute force attack
- Fortunately, that web page doesn't exists on webserver, hence 404 status code

elastic

ML Recipes

Complex Detector Example

- **high_count by ErrorCode over Host partition_field=App**
  - Partition: App
    - Split the data into separate data sets for each App. Create a model (and population) for each.
  - Over: Host
    - For each App, create a population model out of all the Hosts and detect when a Host stands out.
  - By: ErrorCode
    - When modeling and evaluating Hosts, baseline each ErrorCode separately.
  - Function: High_Count
    - Measure the number of times an ErrorCode is observed in the time bucket.

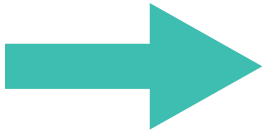
https://www.elastic.co/kr/what-is/elasticsearch-machine-learning/recipes

2. Entity based ML : Data Frame Analytics Concepts

Elastic Data frames transforms

1st Transform: Pivot

```
Original Data:
{
    index: reviews,
    user: rich,
    vendor: acme,
    review: 4,
    timestamp: 2019-04-26T10:12:01
},
{
    index: reviews,
    user: rich,
    vendor: transpop,
    review: 3,
    timestamp: 2019-04-28T10:17:42
},
…
```



```
Output stream:
{
    index: reviews,
    user: rich,
    avg_review: 3.5,
    num_reviews: 2
}
```

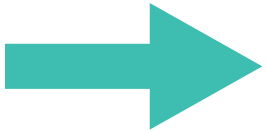
Entity-centric indexing/summarization from a data stream

elastic

Elastic Data frames transforms

1st Transform: Pivot

```
Original Data:
{
    index: reviews,
    user: rich,
    vendor: acme,
    review: 4,
    timestamp: 2019-04-26T10:12:01
},
{
    index: reviews,
    user: rich,
    vendor: transpop,
    review: 3,
    timestamp: 2019-04-28T10:17:42
},
{
    index: reviews,
    user: rich,
    vendor: vcentrix,
    review: 5,
    timestamp: 2019-04-29T09:27:12
},
…
```



```
Output stream:
{
    index: reviews,
    user: rich,
    avg_review: 4,
    num_reviews: 3
}
```

Entity-centric indexing/summarization from a data stream

elastic

“DataFrames”

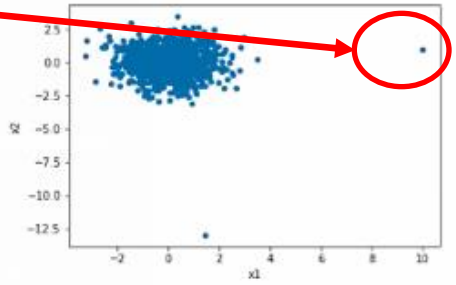
What do people do with it?

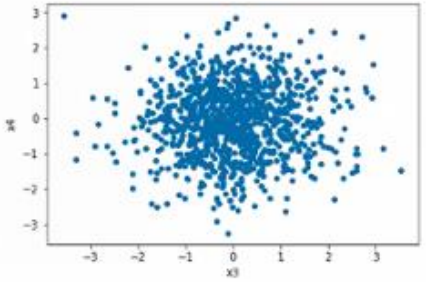
- Analyze multiple variables together
- Model real-world situations
- Find outliers
- Make predictions

```
In [5]: import pandas as pd
        from matplotlib import pyplot as plt

In [15]: dfx = pd.read_csv('/Users/thomasveasey/data/x')
         dfx.head()

Out[15]:
```

	id	x1	x2	x3	x4	x5	x6
0	1	10.000000	0.988253	-1.279285	0.373475	-0.829858	-1.530520
1	2	1.452591	-13.000000	0.266071	-0.067993	0.109710	0.780813
2	3	-1.387421	-1.380247	-0.982101	-1.150864	-0.425088	0.999819
3	4	-0.310192	-0.638912	-0.939306	1.544084	-0.933099	0.167036
4	5	-0.305223	-1.793177	2.122106	0.522571	0.455539	-0.312145

```
In [11]: dfx = pd.read_csv('/Users/thomasveasey/data/x')
         dfx.plot.scatter(x = 'x1', y = 'x2')
         dfx.plot.scatter(x = 'x3', y = 'x4')

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x121f42eb8>
```





Use Cases

- Sales Analysis (House prices, etc.)
- Unusual Transactions (Fraud, etc.)
- User Behavior (Customer churn, etc.)

elastic

“DataFrames”

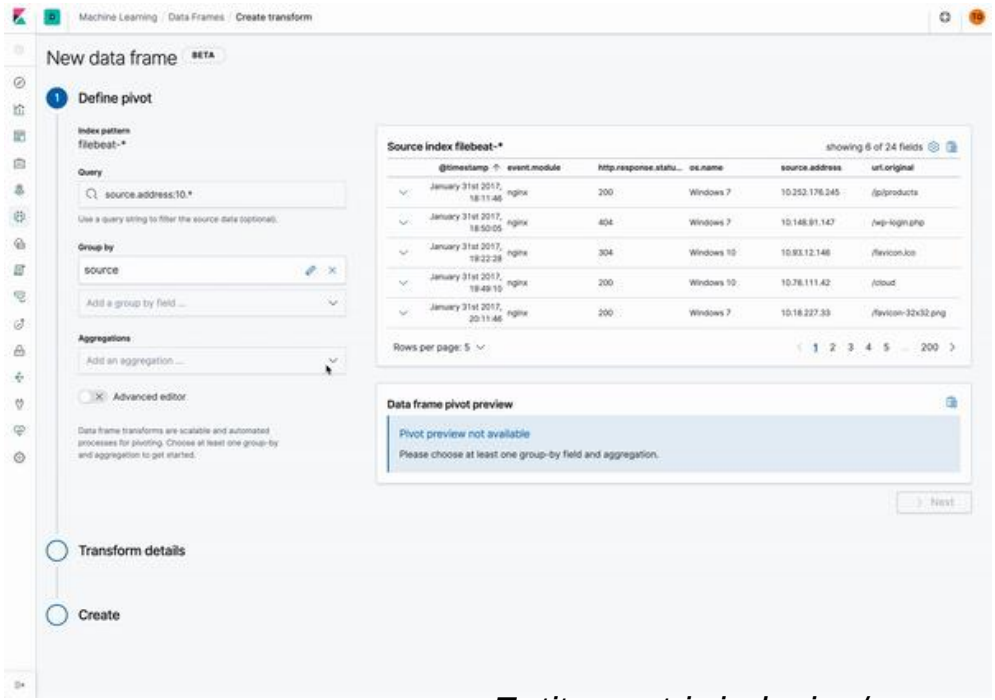
Common Challenges?

- Analysis often limited by RAM of single machine
- Data Scientists cannot/don’t want to design and manage a scalable architecture for analysis
- Data may already be collected and centralized in a ”Big Data” tool, but cannot “use it there”
- Data may exist in a time-series format and would need to be transformed

Lab 5, Transforms

Elastic Data frames transforms

1st Transform: Pivot



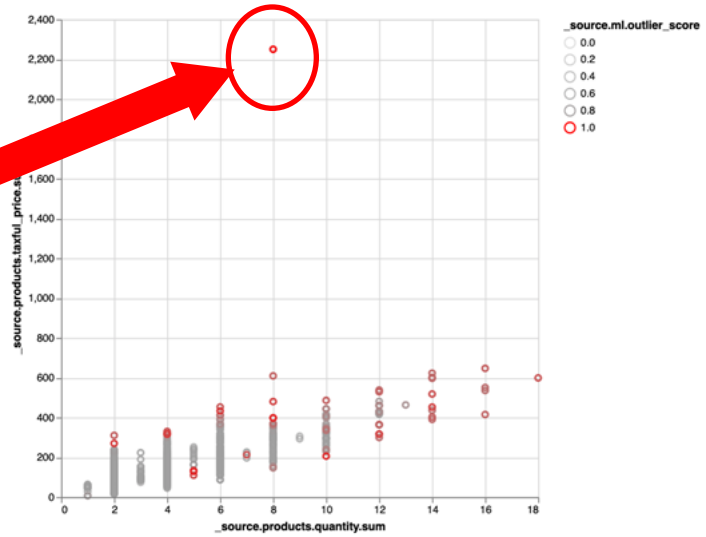
Entity-centric indexing/summarization from a data stream

elastic

Elastic “Data frames analytics”

2st Analysis – Multi-dimensional outlier detection, Regression, Classification…


```
"customer_full_name" : {
  "keyword" : "Wagdi Shaw"
},
"ml__id_copy" : "Vyu9e08pKNasT-9TLV9p3k0AAAAAAAAA",
"products" : {
  "taxful_price" : {
    "sum" : 2250.0
  },
  "quantity" : {
    "sum" : 8.0
  }
},
"ml" : {
  "outlier_score" : 0.9848338961601257,
  "feature_influence.products.quantity.sum" : 0.007586637046188116,
  "feature_influence.products.taxful_price.sum" : 0.992413341999054
}
```

elastic

Topics in Production

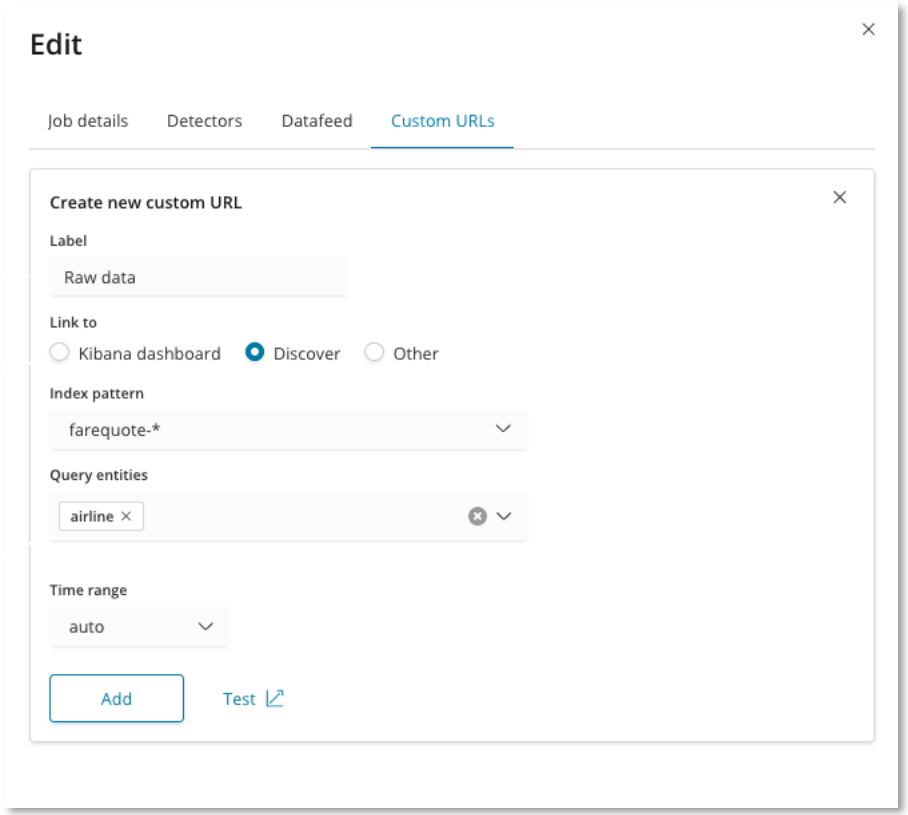
Custom URLs: Linking to other places, in context

Custom URLs in Job config

Name the label

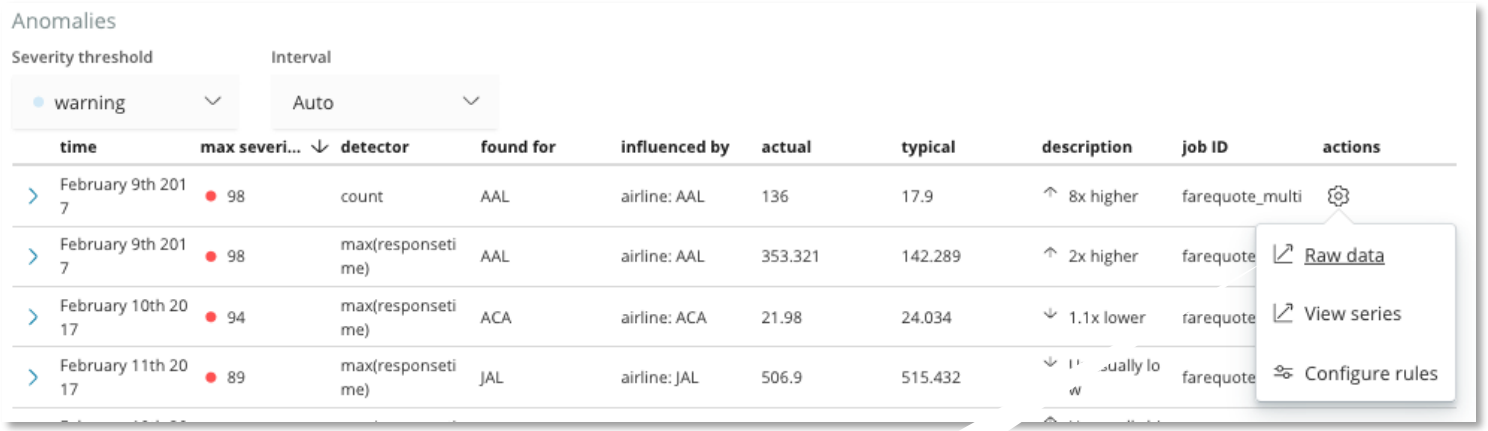
where to link

fields for filtering



elastic

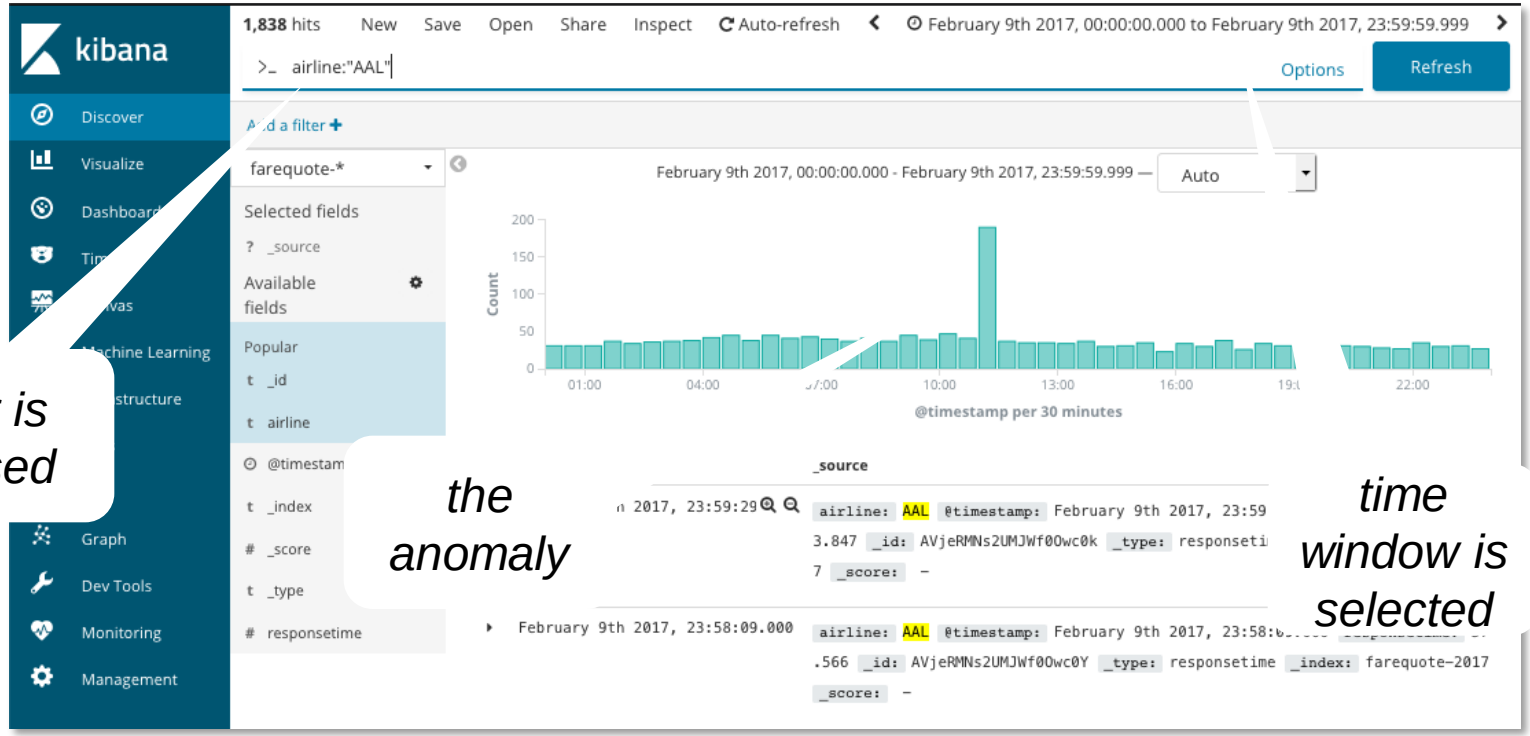
In Anomaly Explorer



Link now available

elastic

Clicking on Custom URL, passing context



elastic

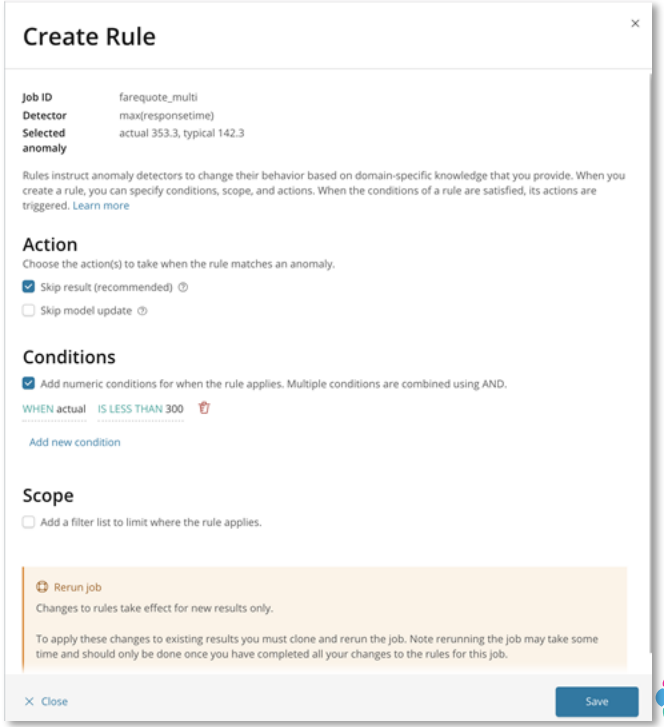
Custom Rules

Adding Custom Rules to ML
(v6.4+)

Define rules that:
- Ignore the creation of anomalies
- Disqualify data from being modeled

Meet certain conditions:
- on “actual”, ”typical” or difference

Optional scoping
- limit when rule is applied



elastic

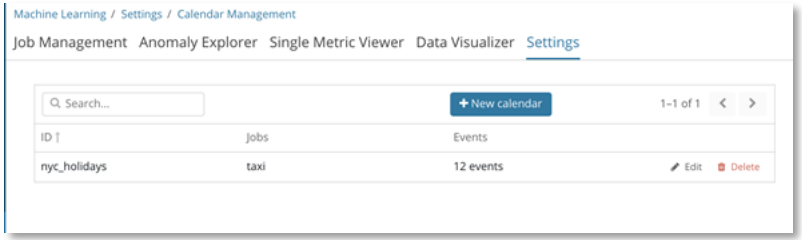
Calendars

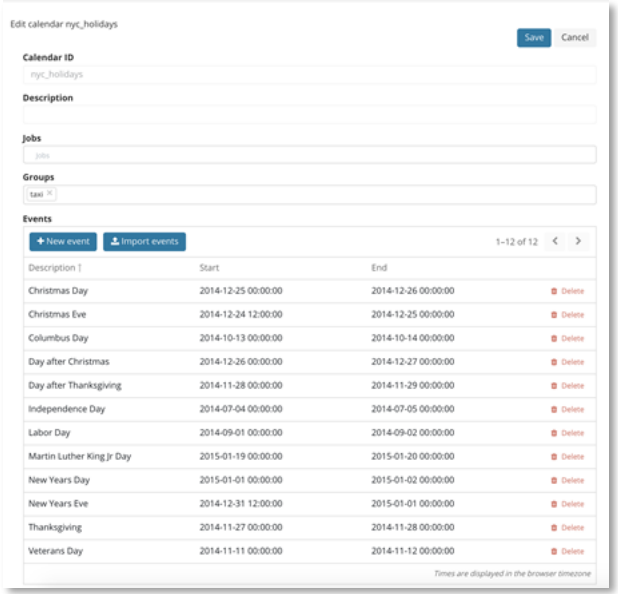
Ignoring Timeframes with Calendars

Define timeframes for ML to
- not create anomalies
- not allow model to see new data

Good for known, upcoming times

Calendars can apply to individual jobs or job groups

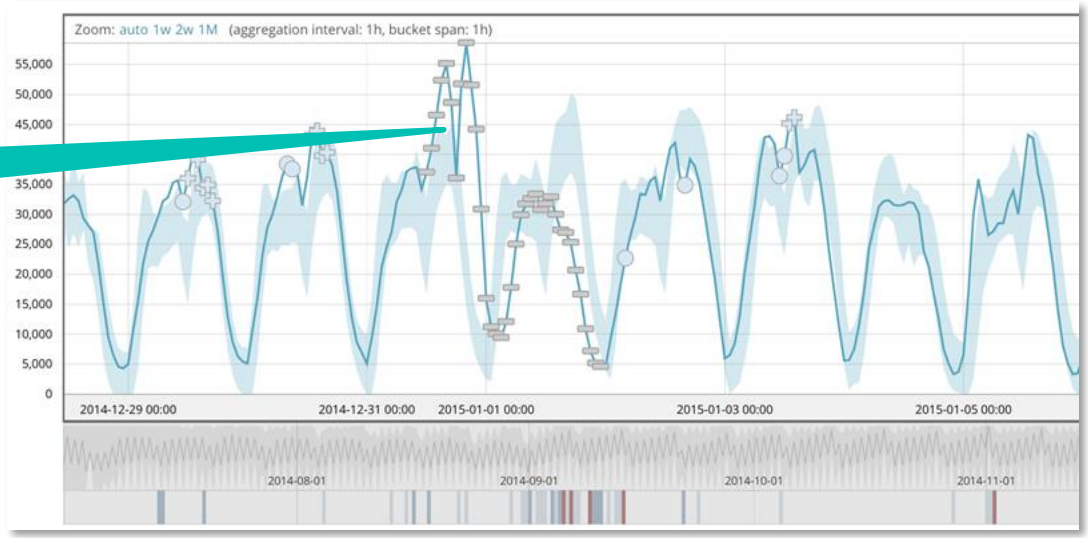




elastic

Ignoring Timeframes with Calendars

Ignored times marked with "—" symbol



elastic

Let’s apply ML to your DATA!!!

elastic

Thank you

https://www.elastic.co/kr/what-is/elasticsearch-machine-learning

elastic