Detecting patterns

Generated: 12/29/2019 12:10 pm
Detecting patterns

This section describes detecting patterns in your data. For a complete list of topics on detecting anomalies, finding and removing outliers, and time series forecasting see About advanced statistics, in this manual.

Detecting patterns in events

The cluster command is a powerful command for detecting patterns in your events. The command groups events based on how similar they are to each other. The cluster command groups events based on the contents of the _raw field, unless you specify another field.

When you use the `cluster` command, two new fields are appended to each event.

- The `cluster_count` is the number of events that are part of the cluster. This is the cluster size.
- The `cluster_label` specifies which cluster the event belongs to. For example, if the search returns 10 clusters, then the clusters are labeled from 1 to 10.

Anomalies come in small or large groups (or clusters) of events. A small group might consist of 1 or 2 login events from a user. An example of a large group of events might be a DDoS attack of thousands of similar events.

Use the cluster command parameters wisely

- Use the `labelonly=true` parameter to return all of the events. If you use `labelonly=false`, which is the default, then only one event from each cluster is returned.
- Use the `showcount=true` parameter so that a `cluster_count` field is added to all of the events. If `showcount=false`, which is the default, the event count is not added to the event.
- The threshold parameter $t$ adjusts the cluster sensitivity. The smaller the threshold value, the fewer the number of clusters.

Other commands to use with the cluster command

- Use the dedup command on the cluster_label column to see the most recent grouped events within each cluster.
To group the events and make the results more readable, use the sort command with the cluster columns. Sort the cluster_count column based on the number of clusters.

- For small groups of events, sort the `cluster_count` column in ascending order.
- For large groups of events sort the `cluster_count` column in descending order.
- Sort the `cluster_label` column in ascending order. Cluster labels are numeric. Sorting in ascending order organizes the events by label, in numerical order.

Return the 3 most recent events in each cluster

The following search uses the CustomerID in the `sales_entries.log` file. Setting `showcount=true` ensures that all events get a `cluster_count`. The cluster threshold is set to 0.7. Setting `labelonly=true` returns the incoming events. The `dedup` command is used to see the 3 most recent events within each cluster. The results are sorted in descending order to group the events.

```
source="/opt/log/ecommsv1/sales_entries.log" CustomerID | cluster showcount=true t=0.7 labelonly=true | table _time, cluster_count, cluster_label, _raw | dedup 3 cluster_label | sort -cluster_count, cluster_label, -_time
```

If you do not set `labelonly=true`, then only one event from each cluster is returned.

See also

Related information
  About advanced statistics

Commands
  `dedup`, `cluster`
  `sort`