Federator.ai for Datadog
User Guide
Contents

Overview ........................................................................................................................................... 3
Datadog Integration Workflows ........................................................................................................ 3

ProphetStor Dashboards in Datadog ................................................................................................. 4

Integrated Monitors .......................................................................................................................... 5

ProphetStor Federator.ai Cluster Overview Dashboard ............................................................... 6
Cluster Resource Usage Predications and Recommendations .................................................. 6
Node Resource Utilization History ................................................................................................. 7

ProphetStor Federator.ai Application Overview Dashboard ....................................................... 8
Application Workload Predictions/Resource Recommendations ........................................... 8
Application Resource Utilization History ..................................................................................... 9
Autoscaling/HPA Recommendations ............................................................................................ 9
  Current/Desired/Recommended Replicas ................................................................................ 10
  CPU Usage/Request/Limit vs. Recommended CPU Limit ....................................................... 10
  Memory Usage/Request/Limit vs. Recommended Memory Limit ........................................... 10
  CPU Usage/Limit Utilization .................................................................................................... 10

ProphetStor Federator.ai Kafka Overview Dashboard .............................................................. 11
  Recommended Replicas vs Current Desired Replicas .......................................................... 12
  Production vs Consumption vs Product Prediction ............................................................... 12
  Kafka Consumer Lag ............................................................................................................. 12
  Consumer Queue Latency ........................................................................................................ 12
  Deployment Memory Usage .................................................................................................... 12
  Deployment CPU Usage ......................................................................................................... 12

ProphetStor Federator.ai Cost Analysis Overview Dashboard .................................................. 13
  Multi-cloud Cost Analysis ...................................................................................................... 13
  Namespace Cost Allocation Charts ........................................................................................... 14

Metrics ............................................................................................................................................. 15
  Common Metrics .................................................................................................................... 15
    Resource recommendation for an application for autoscaling purposes ............................ 15
    Rolled-up CPU usage metrics by Federator.ai for a Kubernetes node .................................. 16
    Rolled-up memory metrics by Federator.ai for a Kubernetes node ...................................... 17
    Rolled-up CPU usage by Federator.ai for a Kubernetes controller ........................................ 18
    Rolled-up memory metrics by Federator.ai for a Kubernetes controller .............................. 19
Health status of Federator.ai integration

Generic Application Metrics for Predictions – Nodes
- Workload predictions for a node
- Average value of predictions for a node
- Minimum value of predictions for a node
- Maximum value of predictions for a node

Generic Application Metrics for Predictions – Controllers
- Workload predictions for a specific controller
- Average value of predictions for a specific controller
- Minimum value of predictions for a specific controller
- Maximum value of predictions for a specific controller

Generic Application Metrics for Resource Planning – Nodes
- Resource planning for a node
- Resource planning for a controller

Kafka Metrics
- Workload predictions for Kafka
- Kafka broker offset rate
- Kafka consumer offset rate

Cost Analysis Metrics
- Cost of a recommended cloud instance
- Cost analysis for a cloud instance
- Cost allocation for a namespace
- Cost prediction for a namespace
- Cost percentage prediction for a namespace

Datadog Federator.ai Metrics List and Updating Interval
Overview

ProphetStor Federator.ai is an AI-based solution that helps enterprises manage, optimize, and auto-scale resources for any applications on Kubernetes. Using advanced machine learning algorithms to predict application workloads, Federator.ai scales the appropriate amount of resources at the right time for optimized application performance. Federator.ai offers:

- AI-based workload prediction for Kafka and other applications
- Resource recommendations based on workload prediction, application, Kubernetes, and other related metrics
- Automatic scaling of application containers through Datadog Watermark Pod Autoscaler (WPA)

Datadog Integration Workflows

Datadog provides monitoring for servers, applications, and services. With Datadog, enterprise customers can monitor their application workloads and get visibility into Kubernetes clusters. Federator.ai integration with Datadog aggregates metrics and events and provides resource predictions/recommendation for Kubernetes deployments and application-aware acceleration and optimization for Kafka.

The following diagram shows how application metrics are used by Federator.ai to predict workloads and to automatically scale applications for better performance:

1. The Datadog Agent sends cluster and application metrics to Datadog Services.
2. The Federator.ai Data-Adapter queries cluster and application metrics from Datadog Services and forwards to the Federator.ai AI engine.
3. The Federator.ai Data-Adapter posts its predictions, recommendations, and plans to Datadog Services.
4. The Datadog Cluster Agent gets the predictions, recommendations, and plans from Datadog Services.
5. WPA applies the plans and auto-scales applications.
6. The Datadog dashboard displays cluster and application metrics as well as predictions, recommendations, and plans from Federator.ai.

**ProphetStor Dashboards in Datadog**

Note that Federator.ai must be installed and configured for Datadog before you can access the ProphetStor dashboards from Datadog. Refer to your *Federator.ai Installation Guide* for more details.

ProphetStor Federator.ai has four dashboards available in Datadog:

- **ProphetStor Federator.ai Cluster Overview** - Resource usage predictions and recommendations for clusters and nodes and historical utilization.
- **ProphetStor Federator.ai Application Overview** - Predicted CPU and memory usage and recommendations for each application.
- **ProphetStor Federator.ai Kafka Overview** - Usage information and recommendations about Kafka consumer replicas.
- **ProphetStor Federator.ai Cost Analysis Overview**

To access these dashboards from Datadog, select *Dashboards ➔ Dashboard List*. The ProphetStor dashboards are listed and can be easily located by the ProphetStor logo. Click the name of the dashboard you want to view to display that dashboard.

For easier access to the dashboards, click the yellow star next to each ProphetStor Federator.ai dashboard so that they appear on under *Dashboards* in the navigation pane.
Integrated Monitors

Federator.ai has two predefined monitors in Datadog to alert you when CPU load and memory usage is predicted to be high for a node over the next 24 hours. By default, these monitors are enabled and will trigger a warning when CPU/memory usage is above 80% and will trigger an alert when CPU/memory usage is above 90% is used.

You can modify these alert/warning thresholds and can set thresholds for warning and alert recovery. In addition, you need to set up email notification for your team members. When a warning/alert is triggered/recovered, an email will be sent to selected recipients.

In the Datadog navigation, select Monitors → New Monitor, click the Recommended Monitors tab, and select Federator.ai under Integrations. Click one of the monitors to set/change your threshold and specify email recipients. Click Save when done.

At the top of the monitor screen, you can see historical instances of when this monitor was triggered for nodes.

Federator.ai alerts and warnings that have occurred can be viewed on the Monitors → Manage Monitors page. Click on each for more details.
ProphetStor Federator.ai Cluster Overview Dashboard

The Cluster Overview dashboard has two sections. The tables at the top display future resource usage predictions and recommendations for clusters and nodes (hosts). The charts below display node resource utilization history.

By default, information in the tables is displayed for all clusters and all hosts for the upcoming 24 hours, but you can filter information for a specific cluster or node (host) and change the timeframe to 7 or 30 days. There are several cluster fields; use the appropriate field for your configuration based upon how your data is tagged.

Cluster Resource Usage Predications and Recommendations

In the Cluster Resource Usage Predications and Recommendations section, the tables display predicted average, minimum, and maximum CPU (in millicores) and memory usage, along with recommended allocations for each cluster and its nodes.
Node Resource Utilization History

In the Node Resource Utilization History section, current and previously predicted CPU and memory usage information is shown in charts. The solid line represents the observed actual usage, while the dotted line shows what was predicted. Click anywhere on a chart to see values for a specific point in time. By default, all nodes are listed, but you can select a specific node from the Host drop-down box at the top of the dashboard or you can select/deselect nodes at the bottom of a chart to show/hide them from the display.

Daily charts include data from the past two days, weekly charts include data from the past week, and monthly charts include data from the past month.

To view a full screen chart, click the zoom icon (top right of widget icon).
ProphetStor Federator.ai Application Overview Dashboard

The *Application Overview* dashboard has three sections. The tables at the top display future resource usage predictions and recommendations for all user-defined applications. Refer to your *Federator.ai Installation Guide* to configure applications that should be monitored by Federator.ai.

The charts below display application resource utilization history, and autoscaling/HPA recommendations. Information in the charts is combined for all selected applications, making them difficult to use. In order for the charts to be useful, you should specify a single application. To do this, select a cluster, a namespace, and a deployment (controller). You can also filter by deployment type. If you are deploying applications in a Kubernetes environment, filter with the fields $kube_deployment or $kube_stateful_set. If you are deploying applications in an OpenShift environment, filter with the fields $oshift_deployment_config or $oshift_deployment.

### Application Workload Predictions/Resource Recommendations

#### Workload Prediction for Next 24 Hours

<table>
<thead>
<tr>
<th>KUBE_NAME</th>
<th>KUBE_DEPLOYMENT</th>
<th>KUBE_STATEFUL</th>
<th>OSHOST_DEPLOYMENT</th>
<th>AVG_CPU (MICRO)</th>
<th>MIN_CPU (MICRO)</th>
<th>MAX_CPU (MICRO)</th>
<th>REC_CPU (MICRO)</th>
<th>AVG_MEM (BYTES)</th>
<th>MIN_MEM (BYTES)</th>
<th>MAX_MEM (BYTES)</th>
<th>REC_MEM (BYTES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>nginx</td>
<td>my-nginx</td>
<td>N/A</td>
<td>N/A</td>
<td>238.64</td>
<td>194.75</td>
<td>360.1</td>
<td>272</td>
<td>21.1M</td>
<td>25.07M</td>
<td>28.82M</td>
<td>38.3M</td>
</tr>
<tr>
<td>myproject</td>
<td>consumer</td>
<td>N/A</td>
<td>N/A</td>
<td>82.96</td>
<td>65.64</td>
<td>116.89</td>
<td>100</td>
<td>1.42G</td>
<td>1.22G</td>
<td>1.84G</td>
<td>1.95G</td>
</tr>
</tbody>
</table>

#### Workload Prediction for Next 7 Days

<table>
<thead>
<tr>
<th>KUBE_NAME</th>
<th>KUBE_DEPLOYMENT</th>
<th>KUBE_STATEFUL</th>
<th>OSHOST_DEPLOYMENT</th>
<th>AVG_CPU (MICRO)</th>
<th>MIN_CPU (MICRO)</th>
<th>MAX_CPU (MICRO)</th>
<th>REC_CPU (MICRO)</th>
<th>AVG_MEM (BYTES)</th>
<th>MIN_MEM (BYTES)</th>
<th>MAX_MEM (BYTES)</th>
<th>REC_MEM (BYTES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>nginx</td>
<td>my-nginx</td>
<td>N/A</td>
<td>N/A</td>
<td>263.98</td>
<td>219.62</td>
<td>284.33</td>
<td>277</td>
<td>35.77M</td>
<td>34.34M</td>
<td>37.18M</td>
<td>41.88M</td>
</tr>
<tr>
<td>myproject</td>
<td>consumer</td>
<td>N/A</td>
<td>N/A</td>
<td>94.41</td>
<td>90.72</td>
<td>98.1</td>
<td>97</td>
<td>1.56G</td>
<td>1.21G</td>
<td>2.05G</td>
<td>2.36G</td>
</tr>
</tbody>
</table>

#### Workload Prediction for Next 30 Days

<table>
<thead>
<tr>
<th>KUBE_NAME</th>
<th>KUBE_DEPLOYMENT</th>
<th>KUBE_STATEFUL</th>
<th>OSHOST_DEPLOYMENT</th>
<th>AVG_CPU (MICRO)</th>
<th>MIN_CPU (MICRO)</th>
<th>MAX_CPU (MICRO)</th>
<th>REC_CPU (MICRO)</th>
<th>AVG_MEM (BYTES)</th>
<th>MIN_MEM (BYTES)</th>
<th>MAX_MEM (BYTES)</th>
<th>REC_MEM (BYTES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>nginx</td>
<td>my-nginx</td>
<td>N/A</td>
<td>N/A</td>
<td>240.39</td>
<td>240.39</td>
<td>240.39</td>
<td>N/A</td>
<td>35.88G</td>
<td>35.88G</td>
<td>35.88G</td>
<td>N/A</td>
</tr>
<tr>
<td>myproject</td>
<td>consumer</td>
<td>N/A</td>
<td>N/A</td>
<td>80.43</td>
<td>80.43</td>
<td>80.43</td>
<td>N/A</td>
<td>1.46G</td>
<td>1.46G</td>
<td>1.46G</td>
<td>N/A</td>
</tr>
</tbody>
</table>

In the *Application Workload Prediction/Resource Recommendations* section, the tables display predicted average, minimum, and maximum CPU (in millicores) and memory usage, along with recommended allocations for each user-defined application in a cluster, namespace, deployment, and deployment type.

By default, information in the charts is displayed for all clusters, namespaces, and deployments, but you can filter information for a specific cluster, namespace, deployment, or deployment type. There are several cluster fields; use the appropriate field for your configuration based upon how your data is tagged.

Information is displayed for the next 24 hours, 7 days, and 30 days.
Application Resource Utilization History

In the Application Resource Utilization History section, current and previously predicted CPU and memory usage information is shown in charts. The solid line represents the observed actual usage, while the dotted line shows what was predicted. Click anywhere on a chart to see values for a specific point in time.

Daily charts include data from the past two days, weekly charts include data from the past week, and monthly charts include data from the past month.

To view a full screen chart, click the zoom icon (top right of widget icon).

Autoscaling/HPA Recommendations

When autoscaling is enabled, CPU and memory usage is monitored, and the number of pods is increased/decreased based on the workload. An auto-scaled pod is called a replica.

The charts display Horizontal Path Autoscaling (HPA) recommendations.
To view a full screen chart, click the zoom icon (top right of widget icon).

**Current/Desired/Recommended Replicas**

This chart displays the number of current, desired, and recommended replicas.

**CPU Usage/Request/Limit vs. Recommended CPU Limit**

This chart displays the CPU usage by the current replicas, along with requested CPU usage, and the CPU limit. Because of the amount of detail, it is recommended to view this chart full screen. Click on the key at the bottom of the chart to show/hide individual metrics.

The solid line/area (blue, `kubernetes.cpu.usage.total`) represents the current CPU usage. The middle dotted line (light gray, `kubernetes.cpu.requests`) represents the requested CPU usage. The top dotted line (dark gray, `kubernetes.cpu.limits`) represents the CPU limit. The bottom dotted line (red, `federatorai.recommendation`) represents the recommended CPU usage.

If actual CPU usage is much lower than what is requested, it indicates that the amount of CPU is being over-provisioned.

**Memory Usage/Request/Limit vs. Recommended Memory Limit**

This chart displays the memory usage by the current replicas, along with requested memory usage, and the memory limit. Because of the amount of detail, it is recommended to view this chart full screen. Click on the key at the bottom of the chart to show/hide individual metrics.

The solid line/area (blue, `kubernetes.memory.usage`) represents the current memory usage. The middle dotted (dark gray, `kubernetes.memory.requests`) line represents the requested memory usage. The top dotted line (light gray, `kubernetes.memory.limits`) represents the memory limit. The bottom line (red, `federatorai.recommendation`) represents the recommended memory usage.

If actual memory usage is much lower than what is requested, it indicates that the amount of memory is being over-provisioned.

**CPU Usage/Limit Utilization**

This chart displays actual CPU utilization, which is determined by dividing actual CPU usage (represented by the blue line) by the CPU limitation.

Since autoscaling is based on usage, replicas are added/removed to keep your system within the CPU limits.
If you are using Kafka software, the *Kafka Overview* dashboard displays usage information and recommendations about Kafka consumer replicas for clusters, namespaces, monitored topics, and monitored consumer groups.

With Kafka, the Datadog Agent monitors and collects many types of Kafka metrics, including resource (CPU and memory) utilization and Kafka workload and performance metrics (such as message queue latency and consumer lag). With the Federator.ai integration, the Kafka message production and consumption rates are continuously collected from Datadog and analyzed by Federator.ai to predict the Kafka message production rate. Based on these predictions, Federator.ai works with the Datadog Watermark Pod Autoscaler (WPA) to automatically scale Kafka consumer replicas to handle the increased or decreased workload.

By default, information in the charts is displayed for all clusters, namespaces, deployments, topics, and consumer groups for the past hour but you can filter information for a specific cluster, namespace, deployment, topic, or consumer group and select a different time frame. There are several cluster fields; use the appropriate field for your configuration based upon how your data is tagged.

Click anywhere on a chart to see values for a specific point in time.
Recommended Replicas vs Current Desired Replicas

This chart displays the recommended and current consumer replicas for the specified time range.

The yellow line represents recommended replicas every five minutes and the green line represents the actual number of replicas. The numbers align closely because once a recommendation is made to increase/decrease the number of replicas and autoscaling occurs, the lines match up.

Production vs Consumption vs Product Prediction

This chart displays the actual observed message production rate (blue line) and consumption/processed rate (purple line) for the specified time range and the predicted rate (yellow line).

With Kafka, autoscaling is based on message production (arrival) vs. consumption (processed); it is not based on CPU/memory usage.

Kafka Consumer Lag

This chart displays the number of messages in the Kafka broker that are yet to be processed by the Kafka consumer for the selected group and time range. This represents messages in the queue waiting to be processed.

Consumer Queue Latency

This chart displays how long it takes for each message to be processed (in milliseconds) for the selected time range.

Deployment Memory Usage

This chart displays observed actual memory usage for the selected time range for the specified consumer.

Deployment CPU Usage

This chart displays observed actual CPU usage for the selected time range for the specified consumer.
ProphetStor Federator.ai Cost Analysis Overview Dashboard

The *Cost Analysis Overview* dashboard has two sections. The Multi-cloud Cost Analysis displays your current cluster costs and outlines your cost for doing business with a variety of cloud providers. The Namespace charts below display the actual cost for each namespace.

**Multi-cloud Cost Analysis**

The Multi-cloud Cost Analysis displays the amount you are spending with your current cluster costs and outlines your cost for doing business with a variety of cloud providers based on fees charged by each provider and recommended changes to your current resource configuration. Select which cluster to display from the appropriate Cluster drop-down box. You can filter the display by pricing options and country.

The Current Cluster Cost box displays what you are currently paying to your cloud provider with your existing resource configuration along with your current cluster configuration, including hosts, roles, availability zone, instance type being used, CPU capacity, memory capacity, and storage size.

The Recommended Cluster boxes display potential savings with different providers. For each provider, the estimated cost/savings are displayed. The table on the right shows the recommended cluster configuration to realize these savings. This includes the recommended instances, regions, and instance types, as well as the number of master and worker servers with each pricing model (on-demand, reserved), and country. The savings typically come from lower provider fees as well as from reducing idle resources and using more cost-effective instances with a provider. The information continually refreshes itself as new data becomes available.
While it is difficult to move between cloud providers, this information can help you reduce costs by changing instance types, reducing idle resources, and selecting a different region from your current provider. Even if you are not currently using a cloud provider, this information can show you what your options are if you move to the cloud.

**Namespace Cost Allocation Charts**

Select which cluster to display from the appropriate Cluster drop-down box.

The **Namespace Cost Allocation for Last 24 hours** chart displays the namespaces with the highest cost for the last day.

The **Namespace Cost Allocation Prediction for Next 30 Days** chart displays a cost projection for the namespaces with the highest predicted costs for the next month.
Metrics

The following is a list of Federator.ai metrics that are used with Datadog. These queries can be used to build custom dashboards in Datadog. The syntax for metrics is:

Metric (Metric name), Value (Metric data), Tags

Common Metrics
Recommendation metrics for generic applications and Kafka

Resource recommendation for an application for autoscaling purposes

Metric
federatorai.recommendation

Used for WPA/HPA to auto-scale Kubernetes controllers for:

- Generic applications - The metric is updated to Datadog every three minutes and provides recommended CPU, memory, number of replicas.
- Kafka - The metric is updated to Datadog every five minutes and provides the recommended number of replicas.

Value
Recommended resource usage.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resource</td>
<td>Type of resource being recommended for auto-scaling the application. Options: replicas, cpu, memory. For Kafka, the resource is replicas. For generic applications, the resource is cpu, memory, or replicas.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from Datadog, identifying the source cluster of the application being monitored.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from Datadog, identifying the source namespace of the application being monitored.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes deployment from Datadog, identifying the source deployment of the application being monitored.</td>
</tr>
</tbody>
</table>
Rolled-up CPU usage metrics by Federator.ai for a Kubernetes node

**Metric**

defedatorai.kubernetes.cpu.usage.total.node

The rolled-up (3600sec, 21600sec, 86400sec) metrics of `kubernetes.cpu.usage.total`, which is the number of cores (in millicores) used by the Kubernetes node.

**Value**

CPU usage in millicores.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which <code>kubernetes.cpu.usage.total</code> is collected. Note: This tag is copied from <code>kubernetes.cpu.usage.total</code>, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which <code>kubernetes.cpu.usage.total</code> is collected. Note: This tag is copied from <code>kubernetes.cpu.usage.total</code>, if available.</td>
</tr>
<tr>
<td>rollup_interval</td>
<td>Rolled-up interval (in second) of the metric. Possible values: 3600, 21600, 86400.</td>
</tr>
</tbody>
</table>
Rolled-up memory metrics by Federator.ai for a Kubernetes node

Metric
federatorai.kubernetes.memory.usage.node

The rolled-up (3600sec, 21600sec, 86400sec) metrics of \textit{kubernetes.memory.usage}, which is the memory usage (in bytes) of the Kubernetes node.

Value
Memory usage in bytes.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which \textit{kubernetes.memory.usage} is collected. Note: This tag is copied from \textit{kubernetes.memory.usage}, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which \textit{kubernetes.memory.usage} is collected. Note: This tag is copied from \textit{kubernetes.memory.usage}, if available.</td>
</tr>
<tr>
<td>rollup_interval</td>
<td>Rolled-up interval (in second) of the metric. Possible values: 3600, 21600, 86400.</td>
</tr>
</tbody>
</table>
Rolled-up CPU usage by Federator.ai for a Kubernetes controller

**Metric**

federatorai.kubernetes.cpu.usage.total.controller

The rolled-up (3600sec, 21600sec, 86400sec) metrics of `kubernetes.cpu.usage.total`, which is the number of cores (in millicores) used by the Kubernetes controller.

**Value**

CPU usage in millicores.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which <code>kubernetes.cpu.usage.total</code> is collected. Note: This tag is copied from <code>kubernetes.cpu.usage.total</code>, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which <code>kubernetes.cpu.usage.total</code> is collected. Note: This tag is copied from <code>kubernetes.cpu.usage.total</code>, if available.</td>
</tr>
<tr>
<td>rollup_interval</td>
<td>Rolled-up interval (in second) of the metric. Possible values: 3600, 21600, 86400.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes Deployment from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_stateful_set</td>
<td>Kubernetes StatefulSet from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>oshift_deployment_config</td>
<td>OpenShift DeploymentConfig from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
</tbody>
</table>
Rolled-up memory metrics by Federator.ai for a Kubernetes controller

**Metric**

federatorai.kubernetes.memory.usage.controller

The rolled-up (3600sec, 21600sec, 86400sec) metrics of `kubernetes.memory.usage`, which is the memory usage (in bytes) of the Kubernetes node.

**Value**

Memory usage in bytes.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| kube_cluster     | Kubernetes cluster name from which `kubernetes.memory.usage` is collected.  
|                  | Note: This tag is copied from `kubernetes.memory.usage`, if available.                                                                     |
| host             | Hostname from which `kubernetes.memory.usage` is collected.                                                                                  
|                  | Note: This tag is copied from `kubernetes.memory.usage`, if available.                                                                     |
| rollup_interval  | Rolled-up interval (in second) of the metric. Possible values: 3600, 21600, 86400.                                                             |
| kube_namespace   | Kubernetes namespace from which the source metric is collected.  
|                  | Note: This tag is copied from the `source_metric`, if available.                                                                            |
| kube_deployment  | Kubernetes Deployment from which the source metric is collected.                                                                           
|                  | Note: This tag is copied from `source_metric`, if available.                                                                                |
| kube_stateful_set| Kubernetes StatefulSet from which the source metric is collected.                                                                          
|                  | Note: This tag is copied from `source_metric`, if available.                                                                                |
| oshift_deployment_config | OpenShift DeploymentConfig from which the source metric is collected.  
|                  | Note: This tag is copied from `source_metric`, if available.                                                                              |
Health status of Federator.ai integration

Metric
federatorai.integration.status

Used for sending Federator.ai health status to Datadog. This metric is always emitted and is used to show on the Datadog integration tile that Federator.ai integration is functioning correctly.

Value
0 – Status ok.
1 – Status error.
Generic Application Metrics for Predictions – Nodes

Workload predictions for a node

Metric
federatorai.prediction.node

Workload prediction for a Kubernetes node. The type of prediction is defined by the source_metric tag.

Value
Predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, kubernetes.cpu.usage.total or kubernetes.memory.usage.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are 24_hours, 7_days, or 30_days.</td>
</tr>
<tr>
<td>prediction_granularity</td>
<td>Time interval between each prediction point. Possible values are 1_hour, 6_hours, or 24_hours.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
</tbody>
</table>
Average value of predictions for a node

Metric
federatorai.prediction.node.avg

The average value of workload predictions for a Kubernetes node for a prediction window. The type of workload predicted is defined by the `source_metric` tag.

Value
Average predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, <code>kubernetes.cpu.usage.toal</code> or <code>kubernetes.memory.usage</code>.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are <code>24_hours</code>, <code>7_days</code>, or <code>30_days</code>.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
</tbody>
</table>
Minimum value of predictions for a node

Metric
federatorai.prediction.node.min

The minimum value of workload predictions for a Kubernetes node for a prediction window. The type of prediction is defined by the source_metric tag.

Value
Minimum predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, kubernetes.cpu.usage.toal or kubernetes.memory.usage.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are 24_hours, 7_days, or 30_days.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected.</td>
</tr>
<tr>
<td></td>
<td>Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which the source metric is collected.</td>
</tr>
<tr>
<td></td>
<td>Note: This tag is copied from source_metric, if available.</td>
</tr>
</tbody>
</table>
Maximum value of predictions for a node

Metric
federator.ai.prediction.node.max

The maximum value of workload predictions for a Kubernetes node for a prediction window. The type of prediction is defined by the `source_metric` tag.

Value
Maximum predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, <code>kubernetes.cpu.usage.toal</code> or <code>kubernetes.memory.usage</code>.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are <code>24_hours</code>, <code>7_days</code>, or <code>30_days</code>.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
</tbody>
</table>
Generic Application Metrics for Predictions – Controllers

Workload predictions for a specific controller

Metric
federatorai.prediction.controller

Workload prediction for a specific controller. The type of prediction is defined by the `source_metric` tag.

Value
Predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, <code>kubernetes.cpu.usage.total</code> or <code>kubernetes.memory.usage</code>.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are <code>24_hours</code>, <code>7_days</code>, or <code>30_days</code>.</td>
</tr>
<tr>
<td>prediction_granularity</td>
<td>Time interval between each prediction point. Possible values are <code>1_hour</code>, <code>6_hours</code>, or <code>24_hours</code>.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes Deployment from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_stateful_set</td>
<td>Kubernetes StatefulSet from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>oshift_deployment_config</td>
<td>OpenShift DeploymentConfig from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
</tbody>
</table>
Average value of predictions for a specific controller

**Metric**

federatorai.prediction.controller.avg

The average value of workload predictions for a specific controller for a prediction window. The type of prediction is defined by the `source_metric` tag.

**Value**

Average predicted workload value.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, <code>kubernetes.cpu.usage.toal</code> or <code>kubernetes.memory.usage</code>.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>The time interval for which the prediction value is calculated. Possible values are <code>24_hours</code>, <code>7_days</code>, or <code>30_days</code>.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes Deployment from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>kube_stateful_set</td>
<td>Kubernetes StatefulSet from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
<tr>
<td>oshift_deployment_config</td>
<td>OpenShift DeploymentConfig from which the source metric is collected. Note: This tag is copied from <code>source_metric</code>, if available.</td>
</tr>
</tbody>
</table>
Minimum value of predictions for a specific controller

Metric
federatorai.prediction.controller.min

The minimum value of workload predictions for a specific controller for a prediction window. The type of prediction is defined by the source_metric tag.

Value
Minimum predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, kubernetes.cpu.usage.total or kubernetes.memory.usage.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are 24_hours, 7_days, or 30_days.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes Deployment from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_stateful_set</td>
<td>Kubernetes StatefulSet from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>oshift_deployment_config</td>
<td>OpenShift DeploymentConfig from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
</tbody>
</table>
Maximum value of predictions for a specific controller

Metric
federatorai.prediction.controller.max

The maximum value of workload predictions for a specific controller for a prediction window. The type of prediction is defined by the source_metric tag.

Value
Maximum predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, kubernetes.cpu.usage.toal or kubernetes.memory.usage.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are 24_hours, 7_days, or 30_days.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes Deployment from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_stateful_set</td>
<td>Kubernetes StatefulSet from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>oshift_deployment_config</td>
<td>OpenShift DeploymentConfig from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
</tbody>
</table>
Generic Application Metrics for Resource Planning – Nodes

Resource planning for a node

Metric
federatorai.resource_planning.node

Resource planning for a specific node.

Value
Planned value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resource</td>
<td>Type of resource being recommended for a node. Possible values are cpu or memory.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are 24_hours, 7_days, or 30_days.</td>
</tr>
<tr>
<td>prediction_granularity</td>
<td>Time interval between each prediction point. Possible values are 1_hour, 6_hours, or 24_hours.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
</tbody>
</table>
Resource planning for a controller

Metric
federatorai.resource_planning.controller

Resource planning for a specific Kubernetes controller.

Value
Planned value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resource</td>
<td>Type of resource being recommended for a controller. Possible values are <em>cpu</em> or <em>memory</em>.</td>
</tr>
<tr>
<td>prediction_window</td>
<td>Time interval for which the prediction value is calculated. Possible values are <em>24_hours</em>, <em>7_days</em>, or <em>30_days</em>.</td>
</tr>
<tr>
<td>prediction_granularity</td>
<td>Time interval between each prediction point. Possible values are <em>1_hour</em>, <em>6_hours</em>, or <em>24_hours</em>.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from <em>source_metric</em>, if available.</td>
</tr>
<tr>
<td>Kube_namespace</td>
<td>Kubernetes namespace from which the source metric is collected. Note: This tag is copied from <em>source_metric</em>, if available.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes <em>Deployment</em> from which the source metric is collected. Note: This tag is copied from <em>source_metric</em>, if available.</td>
</tr>
<tr>
<td>kube_stateful_set</td>
<td>Kubernetes <em>StatefulSet</em> from which the source metric is collected. Note: This tag is copied from <em>source_metric</em>, if available.</td>
</tr>
<tr>
<td>oshift_deployment_config</td>
<td>OpenShift <em>DeploymentConfig</em> from which the source metric is collected. Note: This tag is copied from <em>source_metric</em>, if available.</td>
</tr>
</tbody>
</table>
Kafka Metrics

Workload predictions for Kafka

Metric

federatorai.prediction.kafka

Workload prediction for a Kafka.

Value

Predicted workload value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_metric</td>
<td>Source metric from Datadog used for workload prediction. For example, <code>kubernetes.cpu.usage.total</code> or <code>kubernetes.memory.usage</code>.</td>
</tr>
<tr>
<td>prediction_granularity</td>
<td>Time interval between each prediction point. The value is always 1_minute.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>kube_deployment</td>
<td>Kubernetes Deployment from which the source metric is collected. Note: This tag is copied from source_metric, if available.</td>
</tr>
<tr>
<td>topic</td>
<td>Kafka topic name of the Kafka workload being monitored.</td>
</tr>
<tr>
<td>consumer_group</td>
<td>Kafka consumer group of the Kafka workload being monitored.</td>
</tr>
</tbody>
</table>
Kafka broker offset rate

**Metric**

federatorai.kafka.broker_offset_rate

The number of broker offsets increased in the last 60 seconds.

**Value**

Kafka broker offset rate.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from Datadog, which identifies the source cluster of the Kafka broker being monitored.</td>
</tr>
<tr>
<td>topic</td>
<td>Kafka topic name of the Kafka workload being monitored.</td>
</tr>
</tbody>
</table>

Kafka consumer offset rate

**Metric**

federatorai.kafka.consumer_offset_rate

The number of consumer offsets increased in the last 60 seconds.

**Value**

Kafka consumer offset rate.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from Datadog, which identifies the source cluster of the Kafka consumer being monitored.</td>
</tr>
<tr>
<td>topic</td>
<td>Kafka topic name of the Kafka workload being monitored.</td>
</tr>
<tr>
<td>consumer_group</td>
<td>Kafka consumer group of the Kafka workload being monitored.</td>
</tr>
</tbody>
</table>
**Cost Analysis Metrics**
Recommendation for cloud providers and instances.

**Cost of a recommended cloud instance**

**Metric**

federatorai.recommendation.instance

**Cost of a recommended cloud instance.**

**Value**

Cost value.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name.</td>
</tr>
<tr>
<td>time_interval</td>
<td>Time interval over which the cost of the instance is calculated. The value is 30_days.</td>
</tr>
<tr>
<td>country</td>
<td>Possible values are All, Australia, Bahrain, Belgium, Brazil, Canada, China, Finland, France, Germany, India, Ireland, Japan, Milan, Netherlands, Norway, Republic of South Africa, Singapore, South Korea, Sweden, Switzerland, Taiwan, UK, USA, United Arab Emirates</td>
</tr>
<tr>
<td>display_name</td>
<td>Instance display name, instance name, and brief information.</td>
</tr>
<tr>
<td>instance_type</td>
<td>Instance type from a cloud provider.</td>
</tr>
<tr>
<td>ondemand_instance_master_num</td>
<td>Number of on-demand instances for the master node.</td>
</tr>
<tr>
<td>reserved_instance_master_num</td>
<td>Number of reserved instances for the master node.</td>
</tr>
<tr>
<td>provider</td>
<td>Cloud provider.</td>
</tr>
<tr>
<td>region</td>
<td>Region where the instance is hosted.</td>
</tr>
<tr>
<td>ondemand_instance_worker_num</td>
<td>Number of on-demand instances for worker nodes.</td>
</tr>
<tr>
<td>reserved_instance_worker_num</td>
<td>Number of reserved instances for worker nodes.</td>
</tr>
<tr>
<td>spot_instance_master_num</td>
<td>Number of AWS spot instances for the master node.</td>
</tr>
<tr>
<td>spot_instance_worker_num</td>
<td>Number of AWS spot instance for the worker node.</td>
</tr>
<tr>
<td>pricing_option</td>
<td>Calculated price of an instance based on instance type. Possible values are on-demand-instance, reserved-instance, and spot-instance.</td>
</tr>
</tbody>
</table>
Cost analysis for a cloud instance

**Metric**
federatorai.cost_analysis.instance.cost

Cost analysis for a cloud instance.

**Value**
Cost value.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name.</td>
</tr>
<tr>
<td>host</td>
<td>Kubernetes node name.</td>
</tr>
<tr>
<td>kube_node_role</td>
<td>Kubernetes node role. Possible values are master or worker.</td>
</tr>
<tr>
<td>time_interval</td>
<td>Time interval over which the cost of the instance is calculated. The value is 30_days.</td>
</tr>
<tr>
<td>instance_type</td>
<td>Node instance type provided by the cloud provider.</td>
</tr>
<tr>
<td>provider</td>
<td>Cloud provider. Possible values are aws, gcp, or azure.</td>
</tr>
<tr>
<td>region</td>
<td>Region where the instance is located.</td>
</tr>
</tbody>
</table>
Cost allocation for a namespace

Metric
federatorai.cost_analysis.namespace.cost

Cost of a namespace in a Kubernetes cluster.

Value
Cost value.

Tags

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_interval</td>
<td>Time interval over which the cost of the namespace is calculated. The value is 24_hours.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from Datadog, identifying the source cluster of the namespace being monitored.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from Datadog, identifying the source namespace of the application being monitored.</td>
</tr>
<tr>
<td>provider</td>
<td>Cloud provider. Possible values are aws, gcp, or azure.</td>
</tr>
</tbody>
</table>
Cost prediction for a namespace

**Metric**

federatorai.prediction.namespace.cost

Cost prediction for a namespace in a Kubernetes cluster.

**Value**

Cost value.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>prediction_window</td>
<td>Time interval for which the cost prediction of a namespace is calculated. Possible values are 24_hours, 7_days, or 30_days.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from Datadog, identifying the source cluster of the namespace being monitored.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from Datadog, identifying the source namespace being monitored.</td>
</tr>
<tr>
<td>provider</td>
<td>Cloud provider. Possible values are aws, gcp, or azure.</td>
</tr>
</tbody>
</table>

Cost percentage prediction for a namespace

**Metric**

federatorai.prediction.namespace.cost_pct

Cost percentage prediction of a namespace in a Kubernetes cluster.

**Value**

Cost percentage.

**Tags**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>prediction_window</td>
<td>Time interval for which the cost prediction of a namespace is calculated. Possible values are 24_hours, 7_days, or 30_days.</td>
</tr>
<tr>
<td>kube_cluster</td>
<td>Kubernetes cluster name from Datadog, identifying the source cluster of the namespace being monitored.</td>
</tr>
<tr>
<td>kube_namespace</td>
<td>Kubernetes namespace from Datadog, identifying the source namespace being monitored.</td>
</tr>
<tr>
<td>provider</td>
<td>Cloud provider. Possible values are aws, gcp, or azure.</td>
</tr>
<tr>
<td>Metric_name</td>
<td>Metric_type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>federatorai.integration.status</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.recommendation</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.kafka</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.kafka.broker_offset_rate</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.kafka.consumer_offset_rate</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.node</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.node.avg</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.node.min</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.node.max</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.controller</td>
<td>gauge</td>
</tr>
<tr>
<td>Metric_name</td>
<td>Metric_type</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>federatorai.prediction.controller.avg</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.controller.min</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.controller.max</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.resource_planning.node</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.resource_planning.controller</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.recommendation.instance</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.cost_analysis.instance.cost</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.cost_analysis.namespace.cost</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.prediction.namespace.cost</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.kubernetes.cpu.usage.total.controller</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.kubernetes.memory.usage.controller</td>
<td>gauge</td>
</tr>
<tr>
<td>Metric_name</td>
<td>Metric_type</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>federatorai.kubernetes.cpu.usage.total.node</td>
<td>gauge</td>
</tr>
<tr>
<td>federatorai.kubernetes.memory.usage.node</td>
<td>gauge</td>
</tr>
</tbody>
</table>