Enterprise Steam User Guide

Release 0.9.2.14

H2O.ai

May 31, 2017

CONTENTS

Enterprise Steam is an "instant on" platform that streamlines the entire process of building and deploying applications. It is the industry's first data science hub that lets data scientists and developers collaboratively build, deploy, and refine predictive applications across large scale datasets. Data scientists can publish Python and R code as REST APIs and easily integrate with production applications.

This document describes how to start and use Enterprise Steam. Note that this document assumes that an admin has successfully installed and started Enterprise Steam on a YARN edge node using the instructions provided in the Enterprise Steam Installation and Setup steps.

Note: Before you begin using Enterprise Steam, be sure that your minimum version of H2O is 3.10.4.1. If necessary, follow the instructions on the H2O Download page for your platform to upgrade H2O. For current customers with enterprise support, earlier versions can be supported. Contact H2O.ai if you require support for an earlier version.

CHAPTER

ONE

LOGGING IN TO ENTERPRISE STEAM

In a Chrome web browser, navigate to the Enterprise Steam web server using the login credentials provided by your Admin and/or Enterprise Steam Admin. This Enterprise Steam web server is the server on which an admin has installed Enterprise Steam (for example, http://192.16.2.182:9000). Contact your Admin for the IP address and for login credentials.

1.1 The Enterprise Steam UI

The first time you log in to Enterprise Steam, an empty Enterprise Steam page will display.

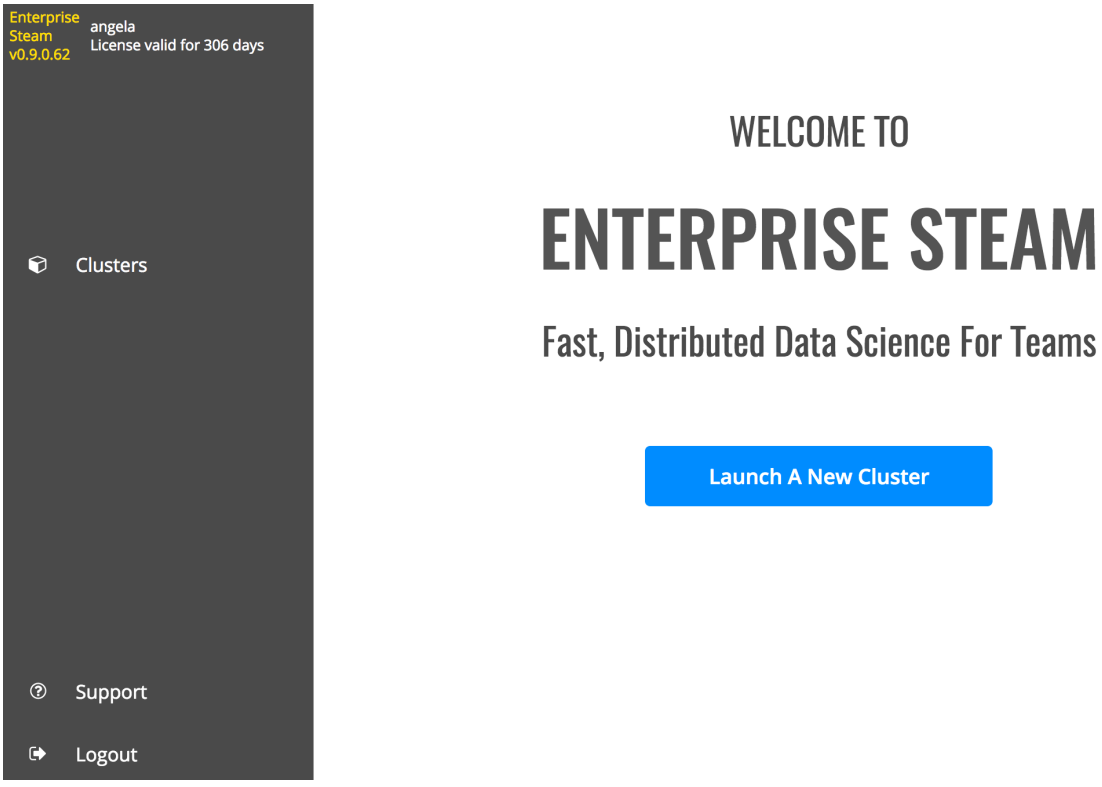


Fig. 1.1: Welcome page

The left navigation provides quick links for all the following:

- Cluster details

- An e-mail link to Enterprise Steam support at H2O
- A logout button

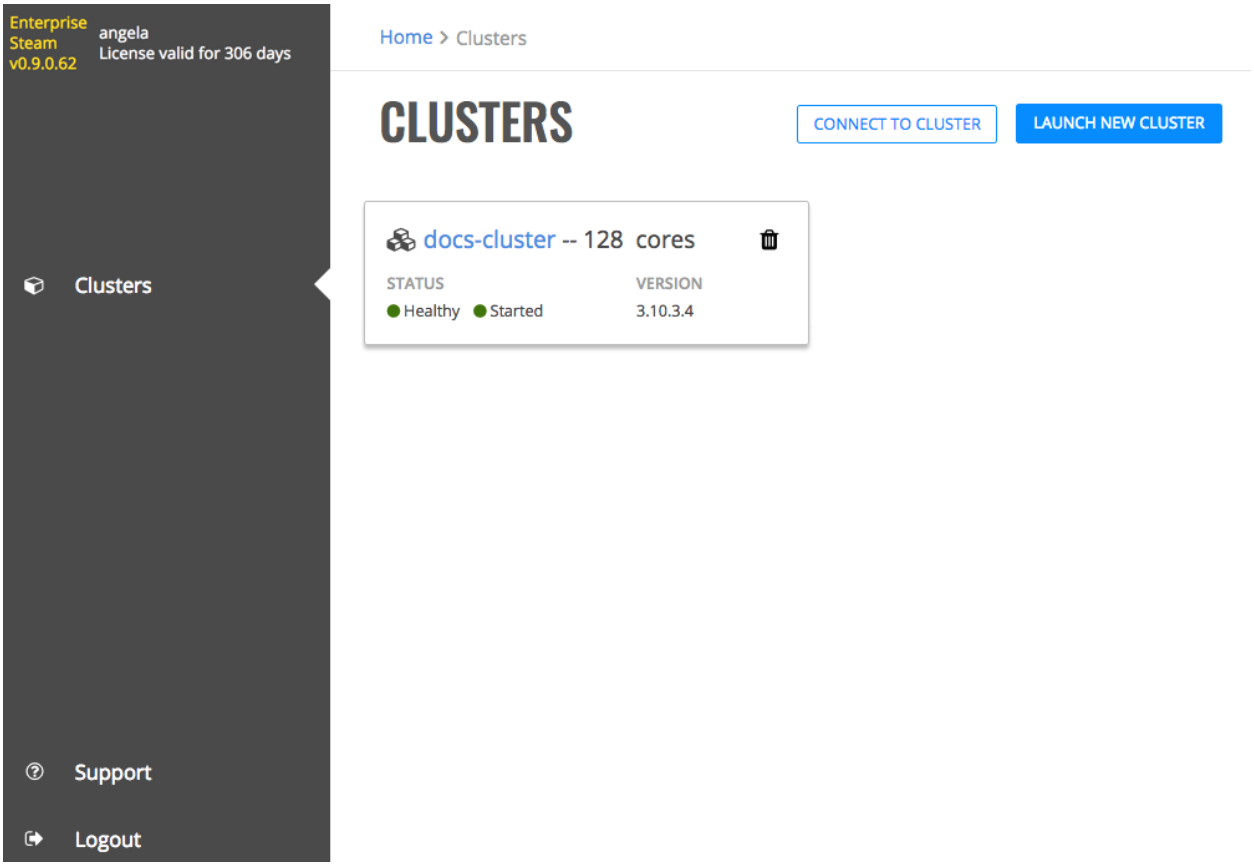
Note: When Enterprise Steam is started for the first time, no clusters will appear in the UI.

CHAPTER

TWO

CLUSTERS

The **Clusters** page shows all H2O clusters that Enterprise Steam is connected to along with the status of the cluster, the number of nodes available on the cluster, and the version of H2O currently running on the cluster. From this page, you can click the cluster name to access H2O Flow (see Using Enterprise Steam with H2O Flow), launch or connect to a new cluster, or delete a cluster.

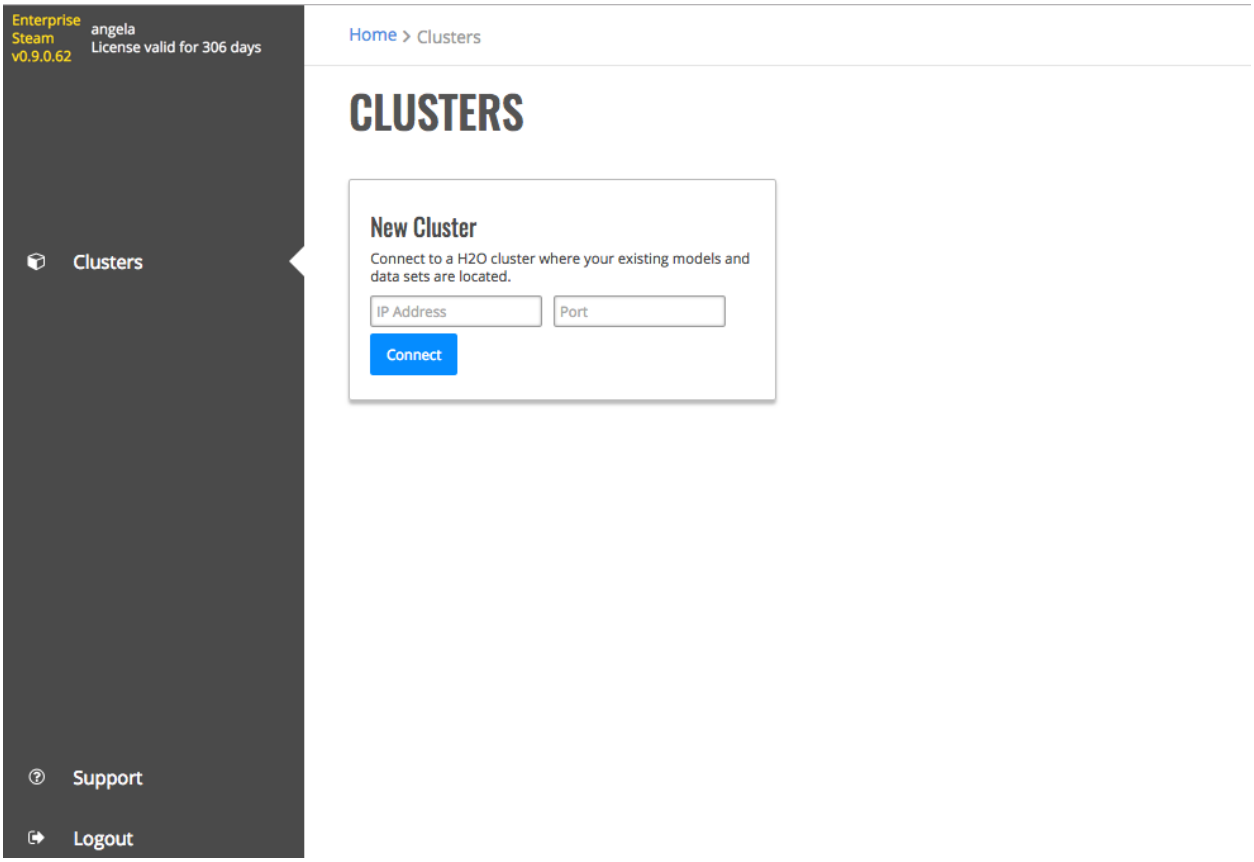


2.1 Connect to a Cluster

If your YARN environment already has H2O installed on one or more clusters, you can use Enterprise Steam to securely connect to that cluster.

1. On the Clusters page, click the **Connect to Cluster** button.

2. Enter the IP address and port for the cluster that is currently running H2O.
3. Click **Connect** to connect to the cluster.



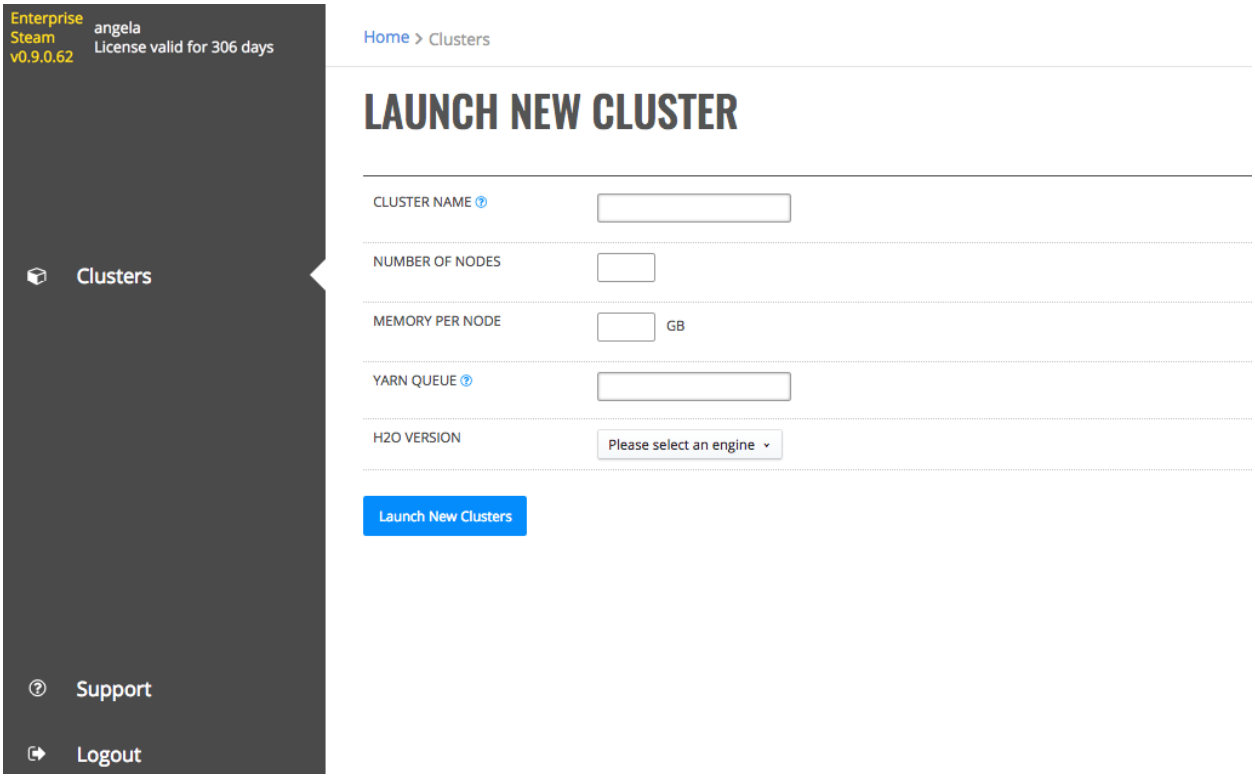
2.2 Launch a New Cluster

You can create a new cluster and start H2O on that cluster by clicking the **Launch New Cluster** button.

1. On the Launch New Cluster form, enter the following information:

- **Cluster Name**: Specify a name for this cluster.
- **Number of Nodes**: Specify the number of nodes for the cluster.
- **Memory per Node (in GB)**: Specify the amount of memory that should be available on each node.
- **YARN Queue**: If your cluster contains queues for allocating cluster resources, specify the queue for this cluster. Note that the YARN Queue cannot contain spaces.
- **H2O Version**: Select a jar file from the dropdown. Note that the Enterprise Steam Admin is responsible for adding engines to Enterprise Steam.

2. Click **Launch New Clusters** when you are done.


Enterprise Steam v0.9.0.62
angela
License valid for 306 days
Clusters
Support
Logout
Home > Clusters
LAUNCH NEW CLUSTER
CLUSTER NAME
NUMBER OF NODES
MEMORY PER NODE
GB
YARN QUEUE
H2O VERSION
Please select an engine
Launch New Clusters

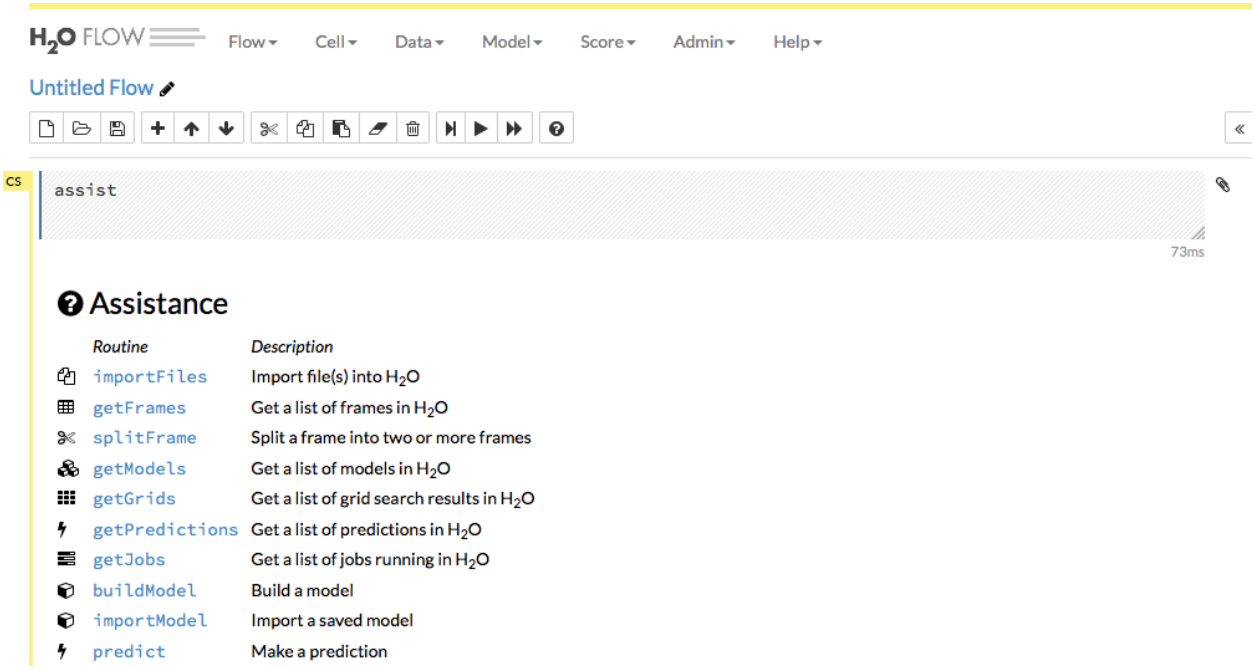
CHAPTER

THREE

USING ENTERPRISE STEAM WITH H2O FLOW

As with other H2O products, Flow can be used alongside Enterprise Steam when performing machine learning tasks. On the **Clusters** page, click the cluster name of the H2O cluster that you want to open.

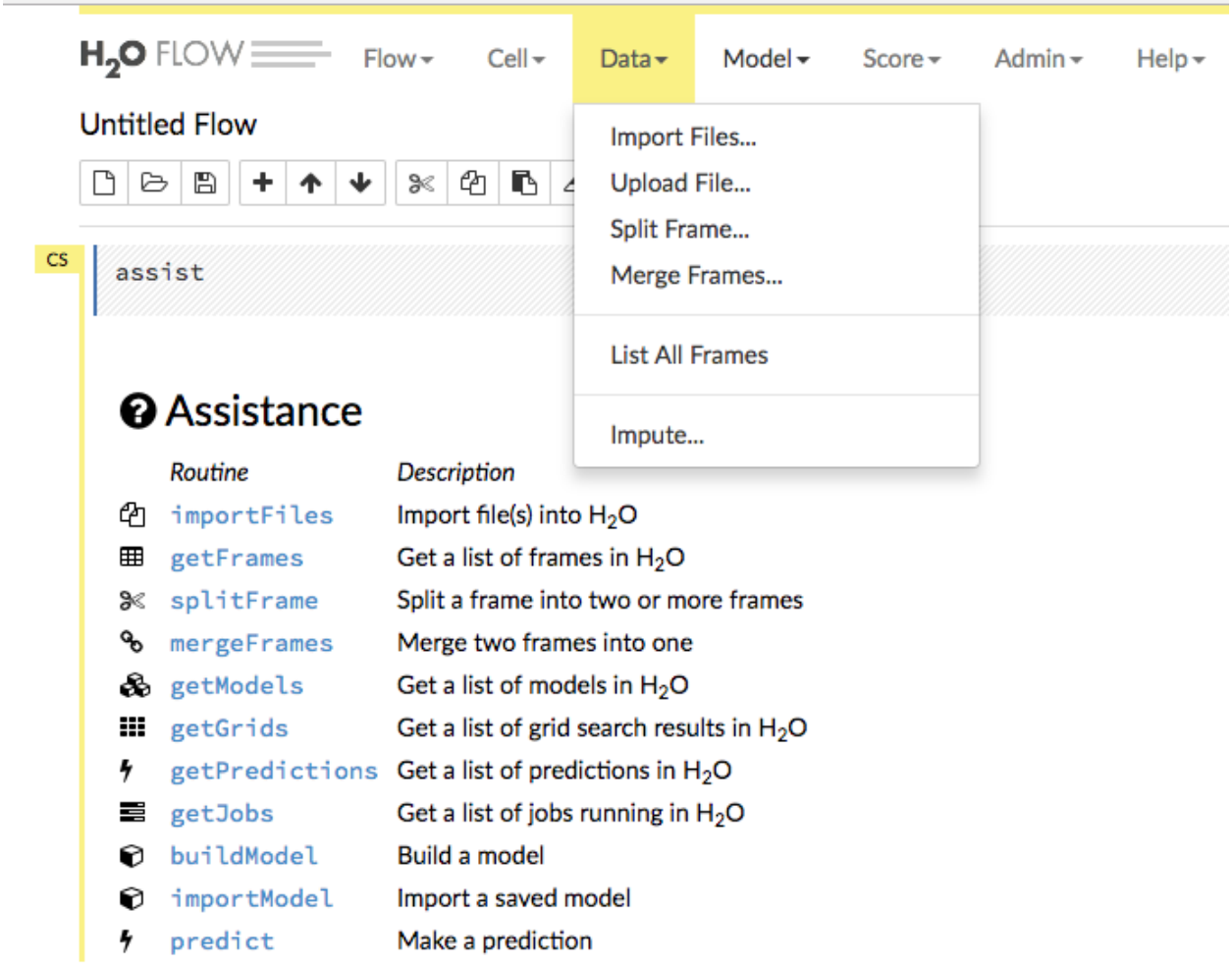
This opens H2O Flow in a new tab.



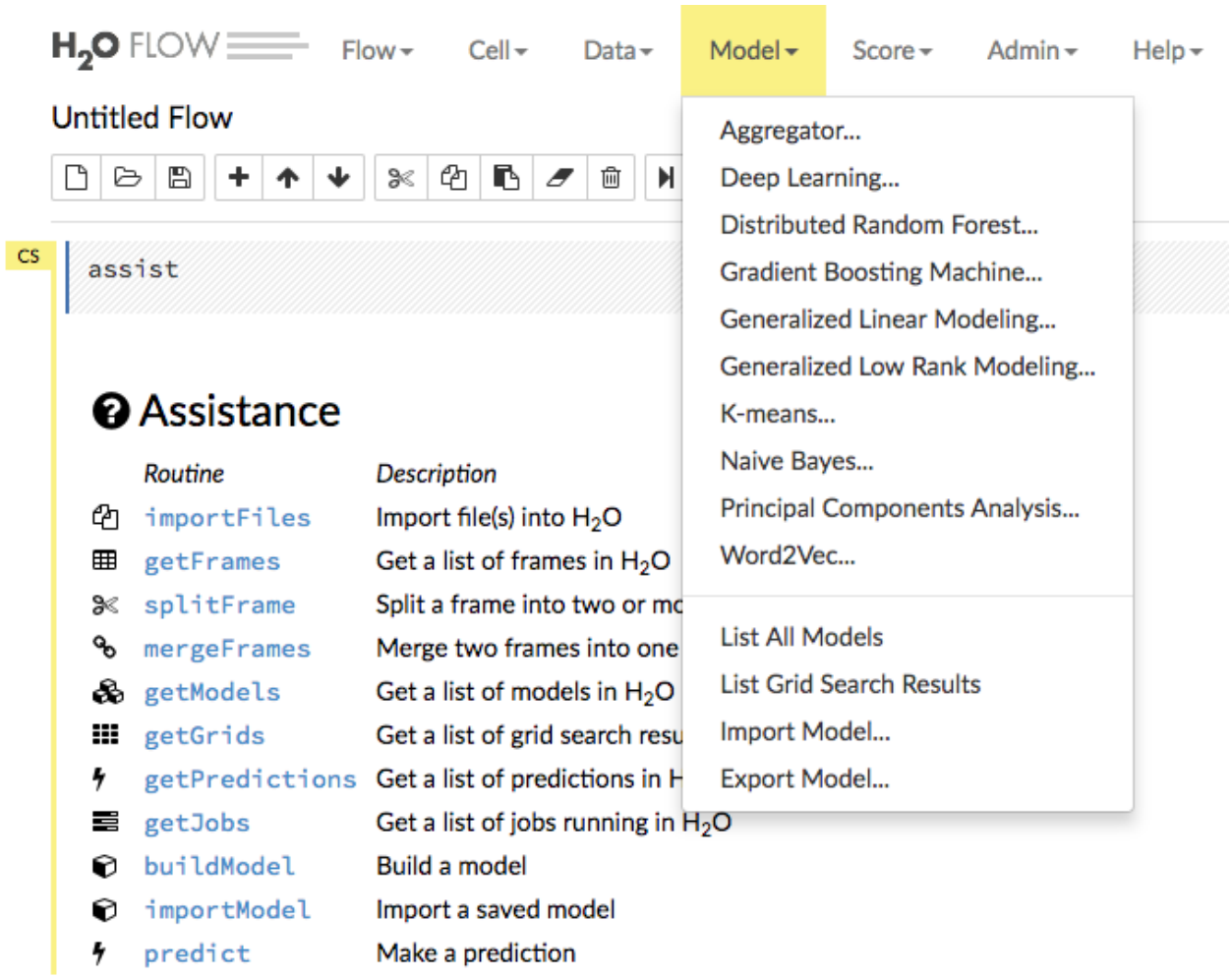
3.1 The H2O Flow UI

Use the menu items at the top to import/upload your data into Flow and to build and score models.

- The **Data** dropdown allows you import or upload a dataset, to split or merge frames, and to impute data.



- Use the **Model** dropdown to select an algorithm and begin building models or to import/export models.



Refer to the H2O Flow documentation for more information on how to use Flow.

CHAPTER

FOUR

USING ENTERPRISE STEAM WITH PYTHON/R

Enterprise Steam provides several Python and R functions that can be used for logging in, creating new clusters, ane connecting to existing clusters. Select one of the topics below to view an end-to-end example.

4.1 Using Enterprise Steam with Python

This section describes how to use the Enterprise Steam for Python. Note that each Python request will result in a warning message. These warnings can be ignored.

4.1.1 Downloading and Installing

1. Go to https://www.h2o.ai/download-enterprise-steam/.
2. Select the Python version that you want to download and install.
3. Open a Terminal window, and navigate to the location where the Python .whl file was downloaded. For example:

```
cd ~/Downloads
```

4. Install Enterprise Steam for Python using `pip install <file_name>`. For example:

```
pip install h2osteam-0.9.1.6-py2.py3-none-any.whl
```

4.1.2 `login`

In Python, use the `login` function to log in to your Enterprise Steam web server. Note that you must already have a username and a password. The web server and your username and password are provided to you by your Enterprise Steam Admin.

```
$ python
>>> import h2osteam
>>> conn = h2osteam.login(url = "https://steam.0xdata.loc",
                          verify_ssl = False,
                          username="jsmith",
                          password="jsmith")
```

4.1.3 start_h2o_cluster

Use the start_h2o_cluster function to create a new cluster. This function takes the following parameters:

- cluster_name: Specify a name for this cluster.
- num_nodes: Specify the number of nodes for the cluster.
- node_memory: Specify the amount of memory that should be available on each node.
- yarn_cueue: If your cluster contains queues for allocating cluster resources, specify the queue for this cluster. Note that the YARN Queue cannot contain spaces.
- callback_ip: Optionally specify the IP address for callback messages from the mapper to the driver (driverif).
- h2o_version: The H2O engine version that this cluster will use. Note that the Enterprise Steam Admin is responsible for adding engines to Enterprise Steam.

```
>>> cluster_config = conn.start_h2o_cluster(cluster_name = 'first-cluster-from-Python
↪',
                                            num_nodes = 2,
                                            node_memory = '30g',
                                            h2o_version = "3.10.4.1")

# Call the cluster to retrieve its ID and configuration params.
>>> cluster_config
{'id': 107, 'connect_params': {'cookies': [u'first-cluster-from-
↪Python=YW5nZWxhOmdrZm53aGJsdWY='], 'ip': 'steam.0xdata.loc', 'context_path': u
↪'jsmit_first-cluster-from-Python', 'verify_ssl_certificates': False, 'https': True,
↪'port': 9999}}
```

Note that after you create a cluster, you can immediately connect to that cluster and begin using H2O. Refer to the following for a complete Python example.

```
>>> import h2o
>>> from h2o.estimators.gbm import H2OGradientBoostingEstimator
>>> h2o.connect(config = cluster_config)

# import the cars dataset
# this dataset is used to classify whether or not a car is economical based on
# the car's displacement, power, weight, and acceleration, and the year it was made
>>> cars = h2o.import_file("https://s3.amazonaws.com/h2o-public-test-data/smalldata/
↪junit/cars_20mpg.csv")

# convert response column to a factor
>>> cars["economy_20mpg"] = cars["economy_20mpg"].asfactor()

# set the predictor names and the response column name
>>> predictors = ["displacement","power","weight","acceleration","year"]
>>> response = "economy_20mpg"

# split into train and validation sets
>>> train, valid = cars.split_frame(ratios = [.8], seed = 1234)

# initialize your estimator
>>> cars_gbm = H2OGradientBoostingEstimator(seed = 1234)

# train your model, specifying your 'x' predictors,
# your 'y' the response column, training_frame, and validation_frame
```

```
>>> cars_gbm.train(x = predictors, y = response, training_frame = train, validation_
↪frame = valid)

# print the auc for the validation data
>>> cars_gbm.auc(valid=True)
```

4.1.4 get_h2o_cluster

Use the `get_h2o_cluster` to retrieve information about a specific cluster using the cluster name.

```
>>> conn.get_h2o_cluster('first-cluster-from-Python')
{'id': 108, 'connect_params': {'cookies': [u'first-cluster-from-
↪Python=YW5nZWxhOnA1bHRreHN5amo='], 'ip': 'steam.0xdata.loc', 'context_path': u
↪'jsmith_first-cluster-from-Python', 'verify_ssl_certificates': False, 'https': True,
↪ 'port': 9999}}
```

4.1.5 stop_h2o_cluster

Use the `stop_h2o_cluster` function to stop a cluster.

```
>>> conn.stop_h2o_cluster(cluster_config)
```

4.2 Using Enterprise Steam with R

This section describes how to use the Enterprise Steam for R. Note that this requires "urltools". Refer to https://github.com/Ironholds/urltools/ for more information.

4.2.1 Downloading and Installing

1. Go to https://www.h2o.ai/download-enterprise-steam/.
2. Select the R version that you want to download and install.
3. Open a Terminal window, and navigate to the location where the Enterprse Steam file was downloaded. For example:

   ```
   cd ~/Downloads
   ```

4. Install Enterprise Steam for R using `R CMD INSTALL <file_name>`. For example:

   ```
   R CMD INSTALL h2osteam_0.9.1.6.tar.gz
   ```

4.2.2 login

Use the `login` function to log in to your Enterprise Steam web server. Note that you must already have a username and a password. The web server and your username and password are provided to you by your Enterprise Steam Admin.

```
$ r
> library(h2osteam)
> conn <- h2osteam.login(url = "https://steam.0xdata.loc",
                         verify_ssl = F,
                         username="jsmith",
                         password="jsmith")
```

4.2.3 start_h2o_cluster

Use the start_h2o_cluster function to create a new cluster. This function takes the following parameters:

- cluster_name: Specify a name for this cluster.
- num_nodes: Specify the number of nodes for the cluster.
- node_memory: Specify the amount of memory that should be available on each node.
- yarn_cueue: If your cluster contains queues for allocating cluster resources, specify the queue for this cluster. Note that the YARN Queue cannot contain spaces.
- callback_ip: Optionally specify the IP address for callback messages from the mapper to the driver (driverif).
- h2o_version: The H2O engine version that this cluster will use. Note that the Enterprise Steam Admin is responsible for adding engines to Enterprise Steam.

```
> cluster_config <- h2osteam.start_h2o_cluster(conn = conn,
                                               cluster_name = "first-cluster-from-R",
                                               num_nodes = 2,
                                               node_memory = "30g",
                                               h2o_version = "3.10.4.1")

# Call the cluster to retrieve its ID and configuration params.
> cluster_config
$id
[1] 109

$connect_params
$connect_params$ip
[1] "steam.0xdata.loc"

$connect_params$port
[1] 9999

$connect_params$cookies
[1] "first-cluster-from-R=YW5nZWxhOnVoYzdyeTNtM3g="

$connect_params$context_path
[1] "jsmith_first-cluster-from-R"

$connect_params$https
[1] TRUE

$connect_params$insecure
[1] TRUE
```

Note that after you create a cluster, you can immediately connect to that cluster and begin using H2O. Refer to the following for a complete R example.

```
> library(h2o)
> h2o.connect(config = cluster_config)

# import the cars dataset
# this dataset is used to classify whether or not a car is economical based on
# the car's displacement, power, weight, and acceleration, and the year it was made
> cars <- h2o.importFile("https://s3.amazonaws.com/h2o-public-test-data/smalldata/
↪junit/cars_20mpg.csv")

# convert response column to a factor
> cars["economy_20mpg"] <- as.factor(cars["economy_20mpg"])

# set the predictor names and the response column name
> predictors <- c("displacement","power","weight","acceleration","year")
> response <- "economy_20mpg"

# split into train and validation sets
> cars.split <- h2o.splitFrame(data = cars,ratios = 0.8, seed = 1234)
> train <- cars.split[[1]]
> valid <- cars.split[[2]]

# train your model, specifying your 'x' predictors,
# your 'y' the response column, training_frame, and validation_frame
> cars_gbm <- h2o.gbm(x = predictors,
                      y = response,
                      training_frame = train,
                      validation_frame = valid,
                      seed = 1234)

# print the auc for your model
> print(h2o.auc(cars_gbm, valid = TRUE))
```

4.2.4 get_h2o_cluster

Use the `get_h2o_cluster` to retrieve information about a specific cluster using the cluster name.

```
> h2osteam.get_h2o_cluster(conn, 'first-cluster-from-R')
$id
[1] 109

$connect_params
$connect_params$ip
[1] "steam.0xdata.loc"

$connect_params$port
[1] 9999

$connect_params$cookies
[1] "first-cluster-from-R=YW5nZWxhOnVoYzdyeTNtM3g="

$connect_params$context_path
[1] "jsmith_first-cluster-from-R"

$connect_params$https
[1] TRUE
```

```
$connect_params$insecure
[1] TRUE
```

4.2.5 stop_h2o_cluster

Use the `stop_h2o_cluster` function to stop a cluster.

```
> h2osteam.stop_h2o_cluster(conn, cluster_config)
```