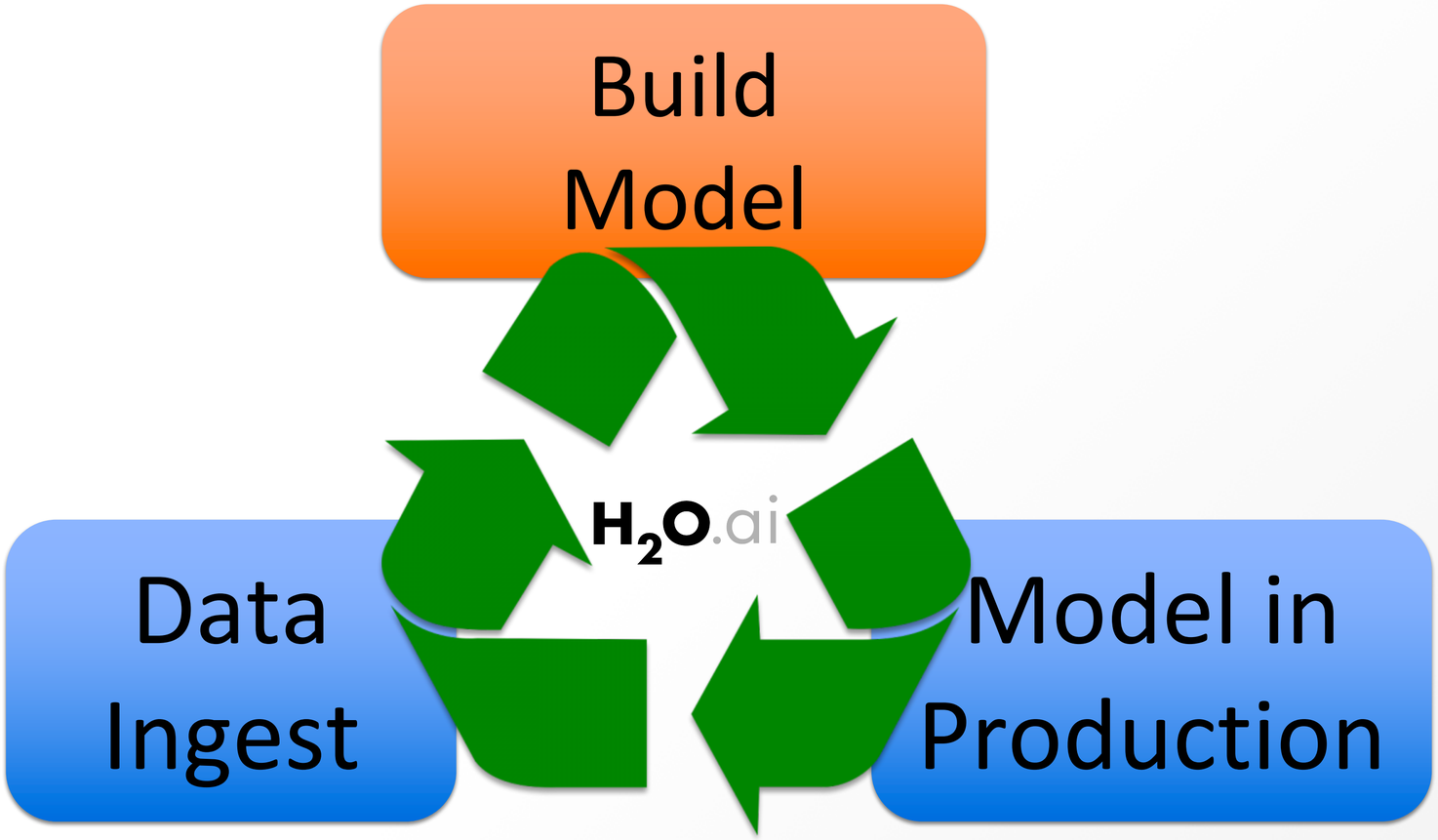
Machine Learning with H_2O

H_2O.ai

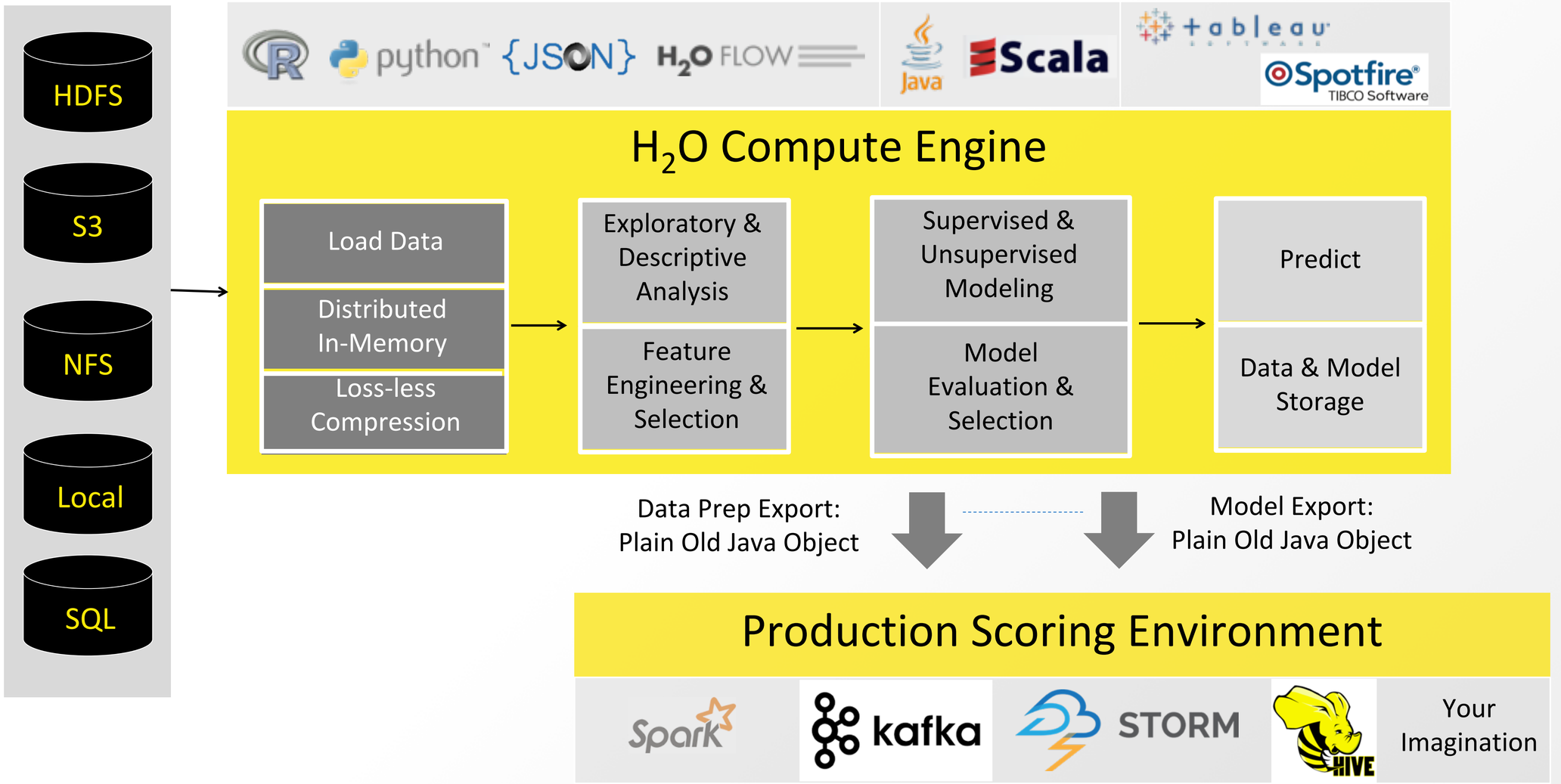
Machine Learning Process



H_2O MACHINE LEARNING PLATFORM

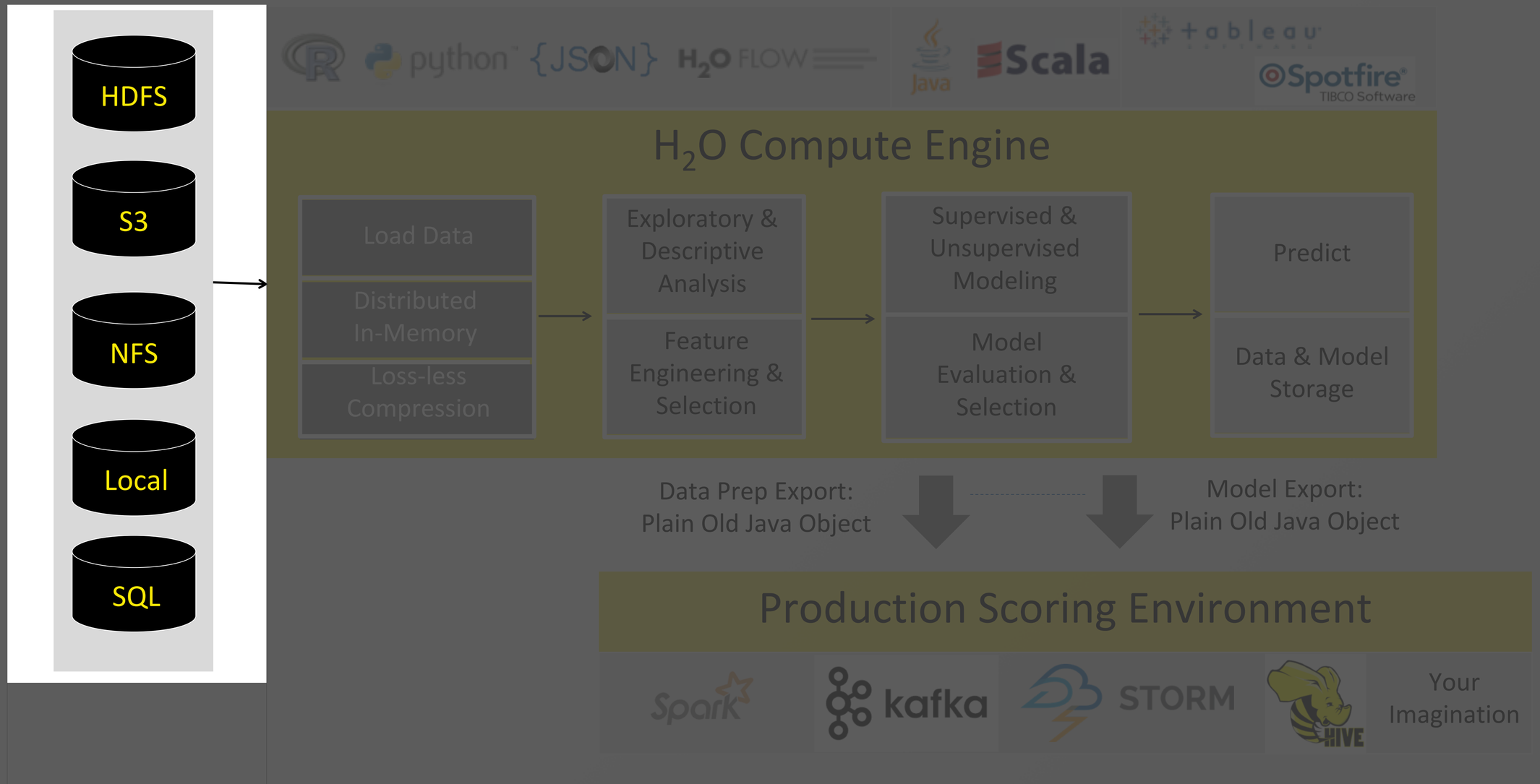
H_2O.ai

High Level Architecture



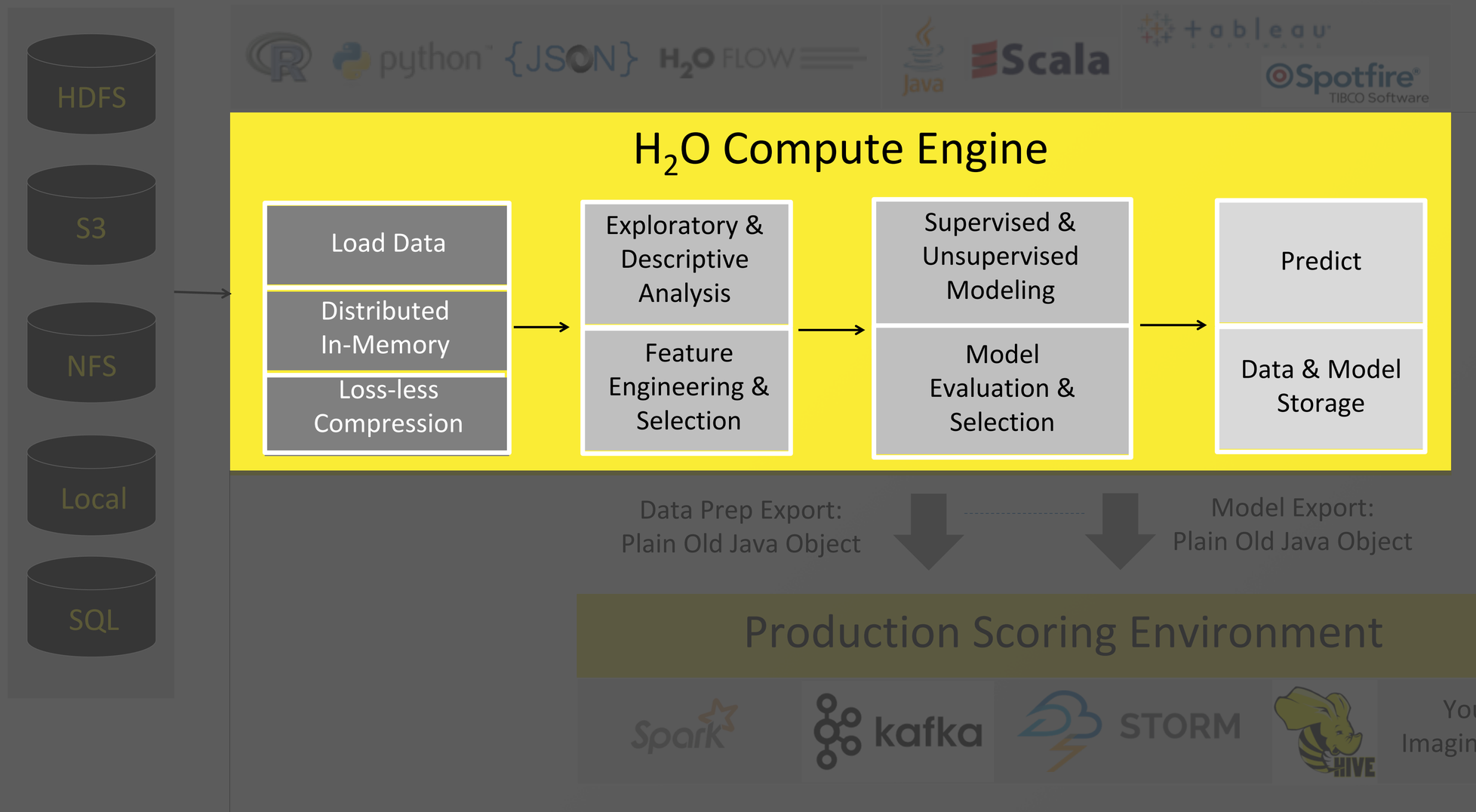
High Level Architecture

Import Data from Multiple Sources



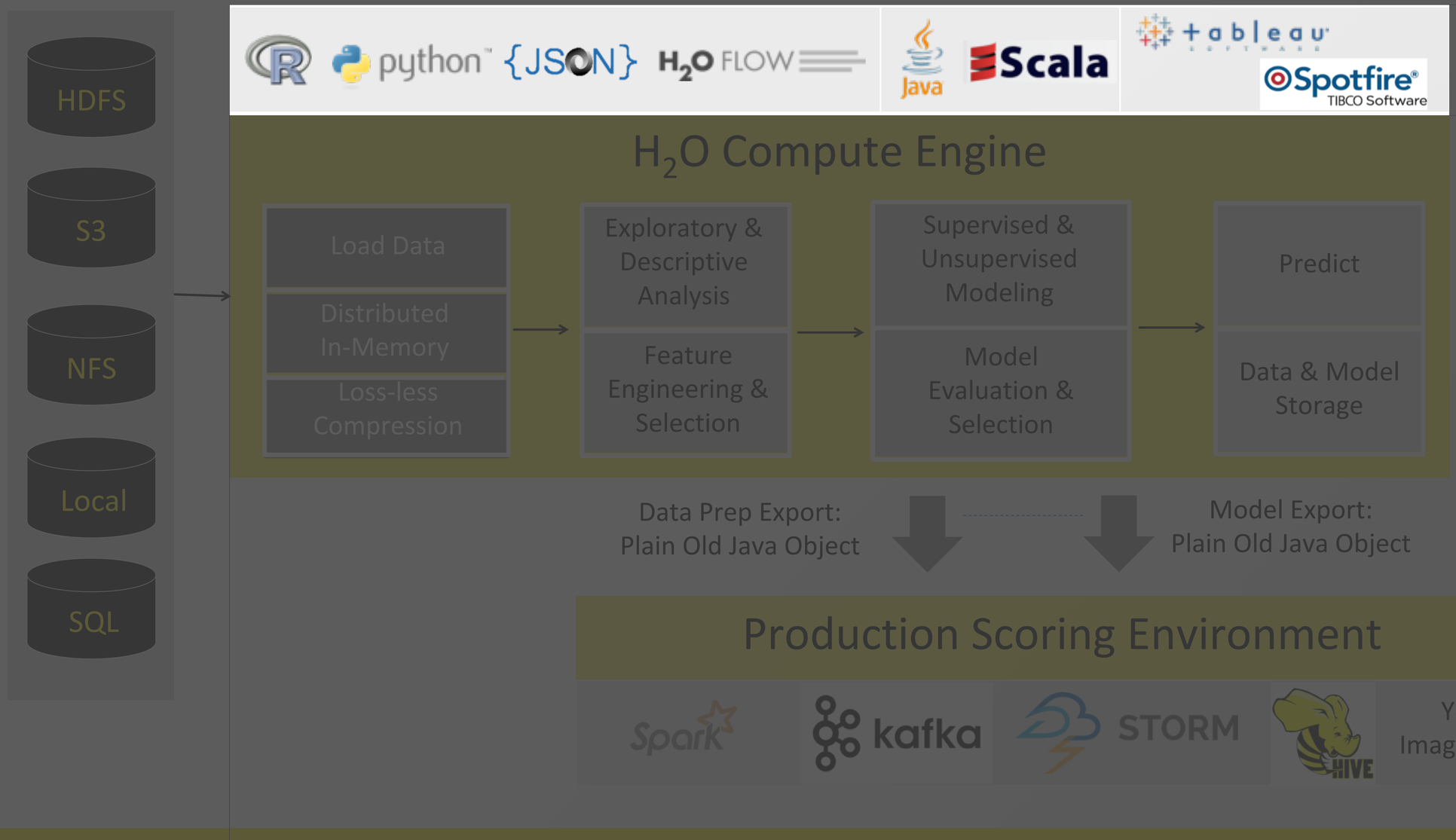
High Level Archi

Fast, Scalable & Distributed Compute Engine Written in Java

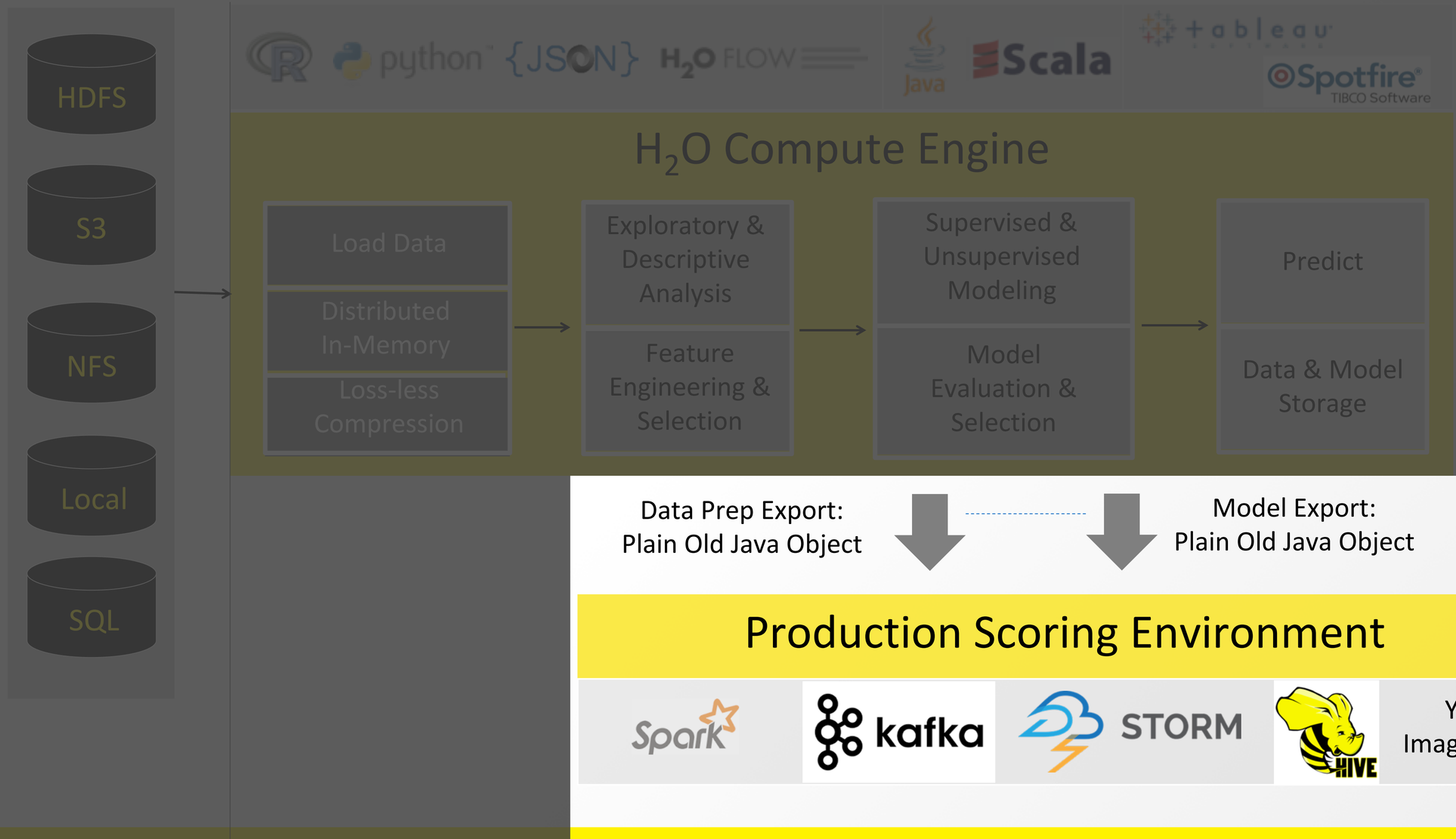


Multiple Interfaces

High Level Architecture



Export Standalone Models for Production



Algorithms Overview

Supervised Learning

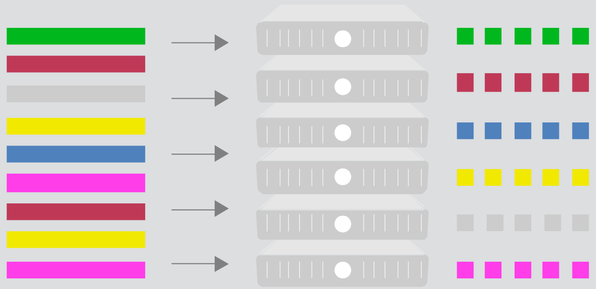
Statistical Analysis	• **Generalized Linear Models**: Binomial, Gaussian, Gamma, Poisson and Tweedie • **Naïve Bayes**
Ensembles	• **Distributed Random Forest**: Classification or regression models • **Gradient Boosting Machine**: Produces an ensemble of decision trees with increasing refined approximations
Deep Neural Networks	• **Deep learning**: Create multi-layer feed forward neural networks starting with an input layer followed by multiple layers of nonlinear transformations

Unsupervised Learning

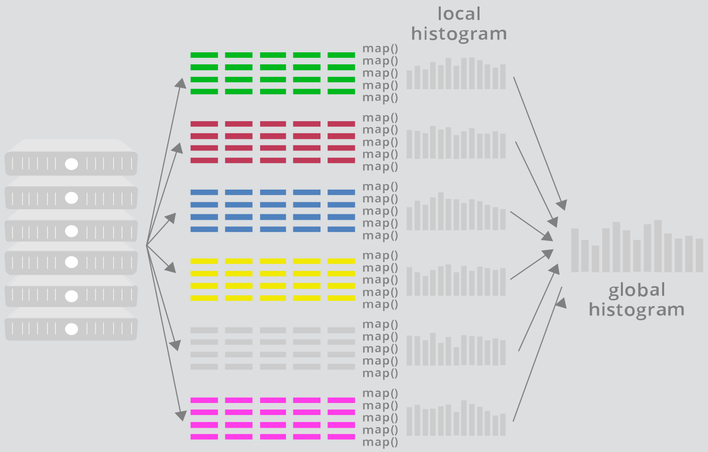
Clustering	• **K-means**: Partitions observations into k clusters/groups of the same spatial size. Automatically detect optimal k
Dimensionality Reduction	• **Principal Component Analysis**: Linearly transforms correlated variables to independent components • **Generalized Low Rank Models:** extend the idea of PCA to handle arbitrary data consisting of numerical, Boolean, categorical, and missing data
Anomaly Detection	• **Autoencoders**: Find outliers using a nonlinear dimensionality reduction using deep learning

Distributed Algorithms

Foundation for Distributed Algorithms



Parallel Parse into **Distributed Rows**



Fine Grain Map Reduce Illustration: Scalable Distributed Histogram Calculation for GBM

Advantageous Foundation

- Foundation for In-Memory Distributed Algorithm Calculation - **Distributed Data Frames** and **columnar compression**
- All algorithms are distributed in H_2O: GBM, GLM, DRF, Deep Learning and more. Fine-grained map-reduce iterations.
- **Only enterprise-grade, open-source distributed algorithms in the market**

User Benefits

- "Out-of-box" functionalities for all algorithms **(NO MORE SCRIPTING)** and uniform interface across all languages: R, Python, Java
- **Designed for all sizes of data sets, especially large data**
- **Highly optimized Java code for model exports**
- **In-house expertise for all algorithms**

H2O.ai

H2O.ai

Key features

- Open Source (Apache 2.0)
- All supported ML algorithms are coded by our engineers
- Designed for speed, scalability and for super large data-sets
- Same distribution for open source community & enterprise
- Very active production, every other week release
- Vibrant open source community
  - https://community.h2o.ai
- Enterprise Support portal
  - https://support.h2o.ai
- We have 70,000 users, 8,000 organizations and growing daily

Usage: Simple Solution

- Single Deployable compiled Java code (jar)
- Ready to use point and click FLOW Interface
- Connection from R and Python after specific packages are installed
- Use Java, Scala natively and any other language through RESTful API
- Deployable models - Binary & Java (POJO & MOJO)
- One click prediction/scoring engine

Usage: Complex Solution

- Multi-node Deployment
- Spark and Hadoop distributed environment
  - Sparkling Water (Spark + H2O)
- Data ingested from various inputs
  - S3, HDFS, NFS, JDBC, Object store etc.
  - Streaming support in Spark (through Sparking Water)
- Distributed machine learning for every algorithm in platform
- Prediction service deployment on several machines

docs.h2o.ai

Getting Started & User Guides

H_2O

- What is H2O?
- **H2O User Guide** (Main docs)
- H2O Book (O'Reilly)
- Recent Changes
- Open Source License (Apache V2)

- Quick Start Video - Flow Web UI
- Quick Start Video - R
- Quick Start Video - Python

Download H_2O

Sparkling Water

- What is Sparkling Water?
- Sparkling Water Booklet
- PySparkling Readme 2.0 | 2.1 | 2.2
- RSparkling Readme
- Open Source License (Apache V2)

- Quick Start Video - Scala

Download Sparkling Water

Steam

- What is Steam?
- Steam User Guide
- Recent Changes
- Open Source License (AGPL)

Download Steam

Deep Water (preview)

- Deep Water Readme
- Deep Water Booklet
- Deep Water AMI Guide
- Deep Water Docker Image
- Open Source License (Apache V2)

Launch Deep Water AMI
(choose p2.xlarge)

Q & A

- FAQ
- Issue Tracking (JIRA)
- Stack Overflow
- h2ostream Google Group
- Gitter
- Cross Validated

For Supported Enterprise Customers

Enterprise Support Web | Email

Algorithms

Supervised Learning

Generalized Linear Modeling (GLM)	Tutorial	Booklet	Reference	Tuning
Gradient Boosting Machine (GBM)	Tutorial	Booklet	Reference	Tuning
Deep Learning	Tutorial	Booklet	Reference	Tuning
Distributed Random Forest	Tutorial	Booklet	Reference	Tuning
Naive Bayes	Tutorial	Booklet	Reference	Tuning
Stacked Ensembles	Tutorial	Booklet	Reference	Tuning
XGBoost	Tutorial	Booklet	Reference	Tuning

Unsupervised Learning

Generalized Low Rank Models (GLRM)	Tutorial	Reference
K-Means Clustering	Tutorial	Reference
Principal Components Analysis (PCA)	Tutorial	Reference

Miscellaneous

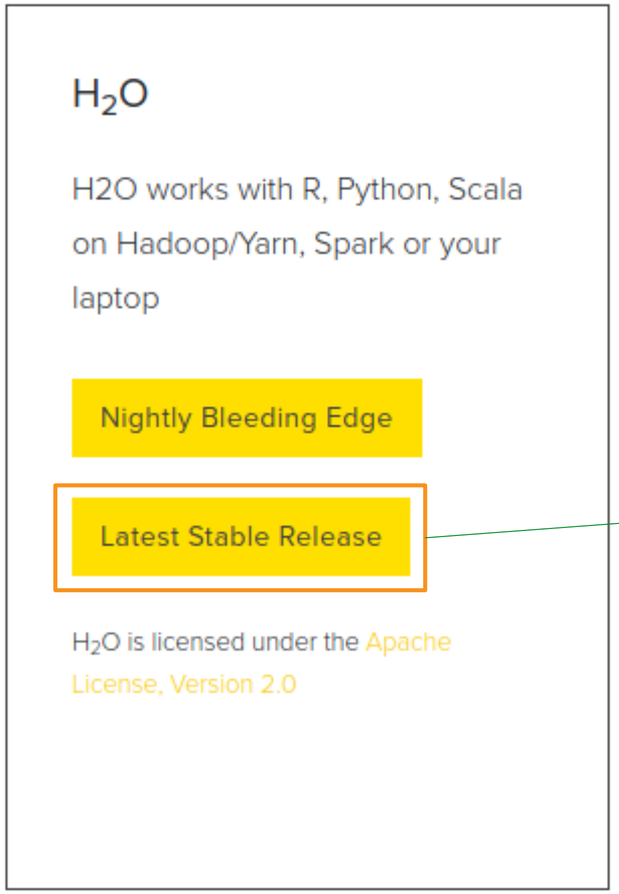
Word2vec	Tutorial	Reference

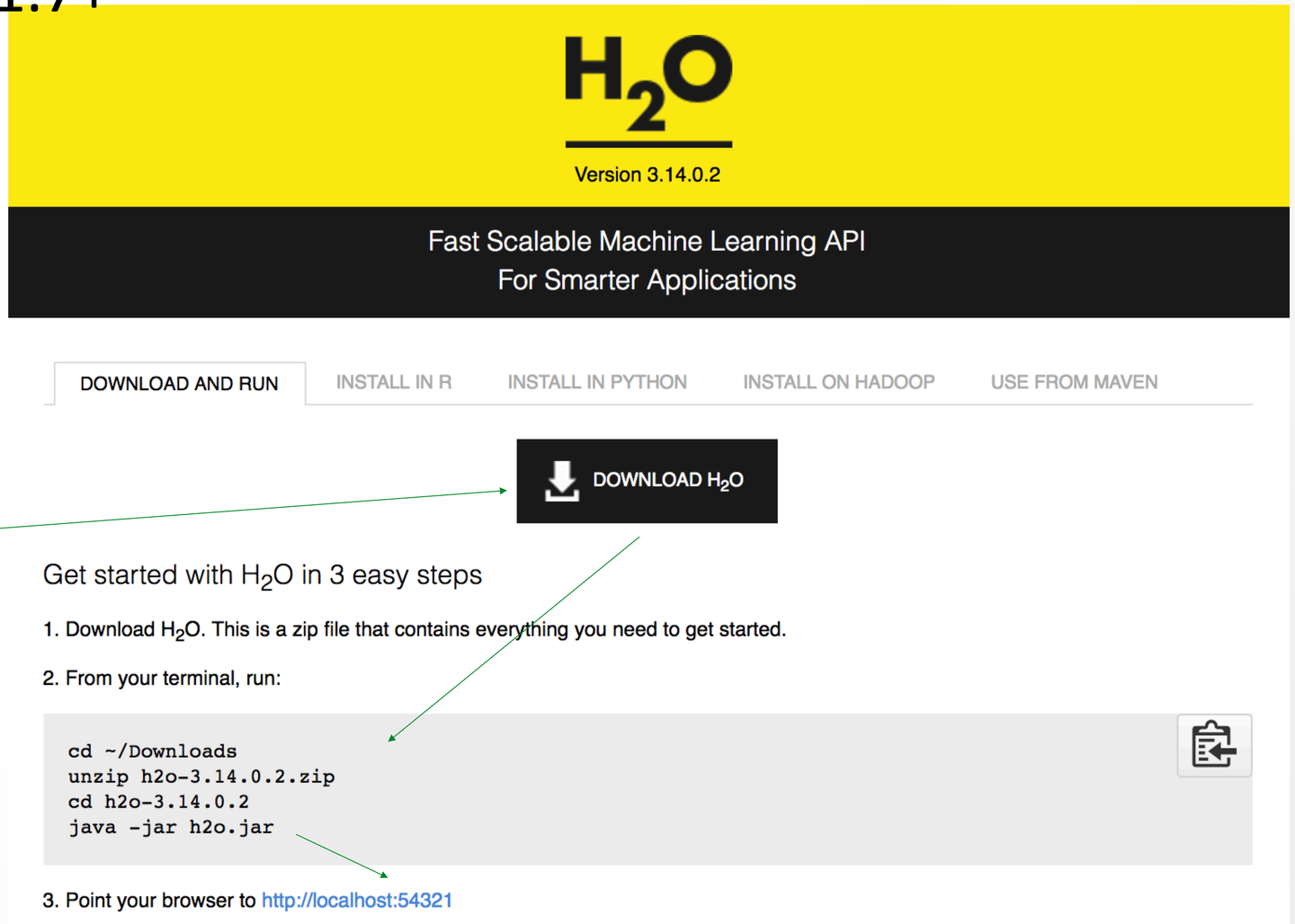
TOP

H_2O.ai

www.h2o.ai/download

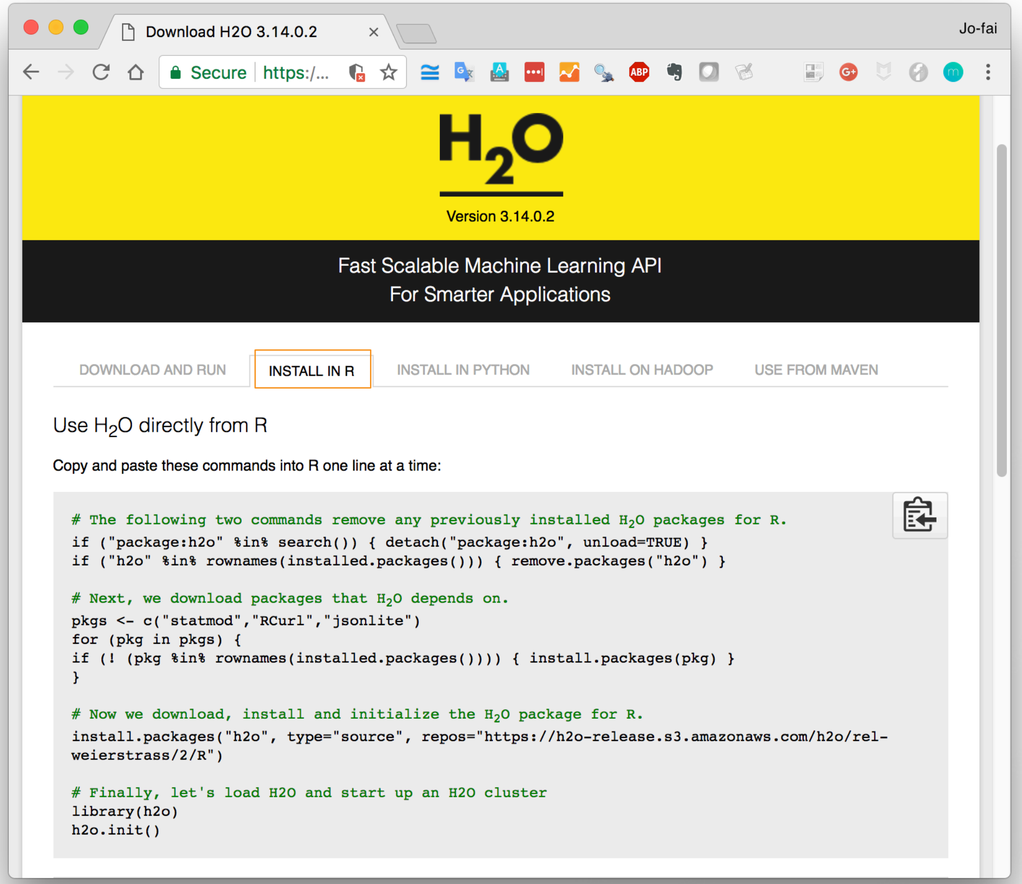
Prerequisite: Java version 1.7+

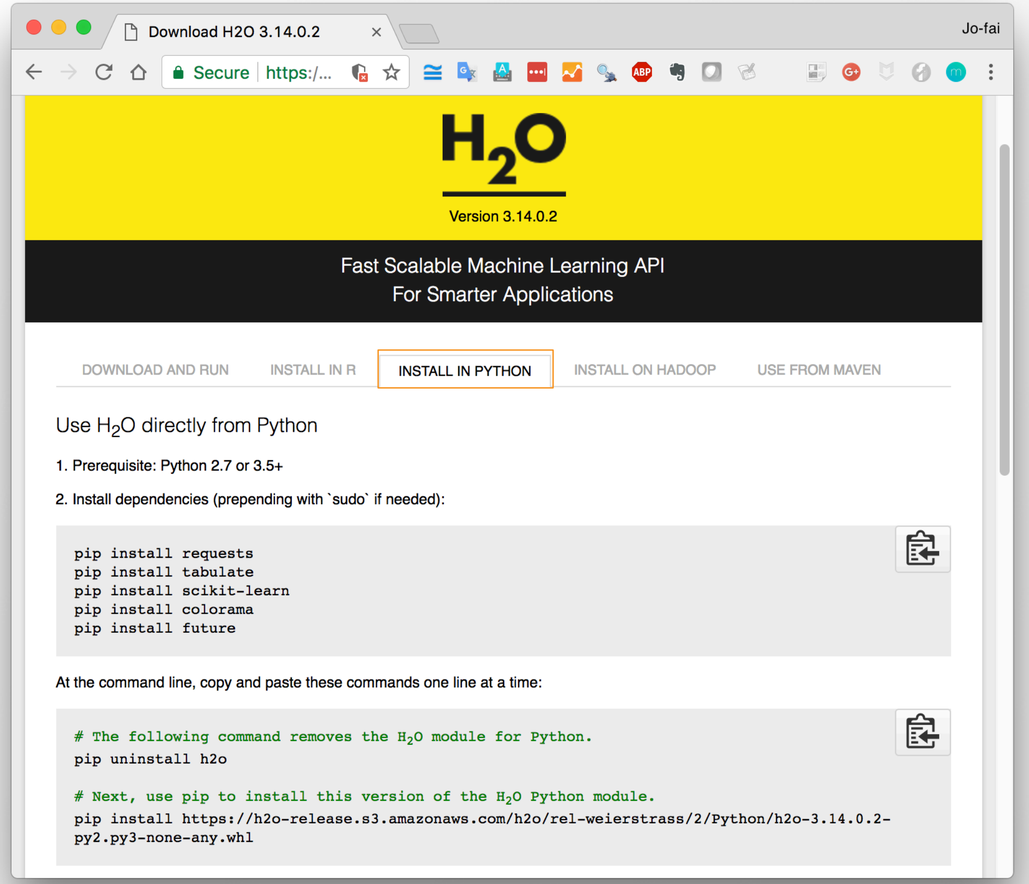




H2O.ai

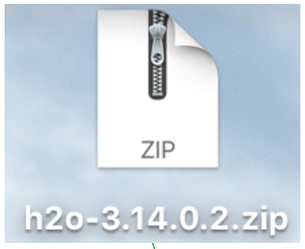
Install and Start H_2O in R / Python

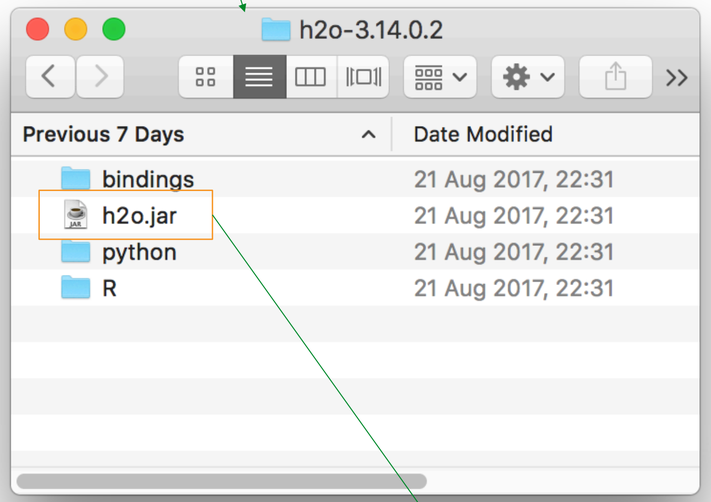


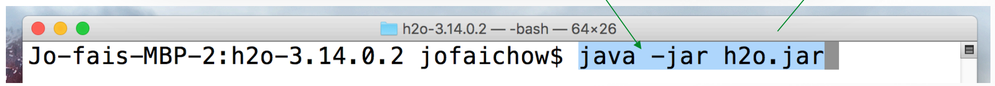


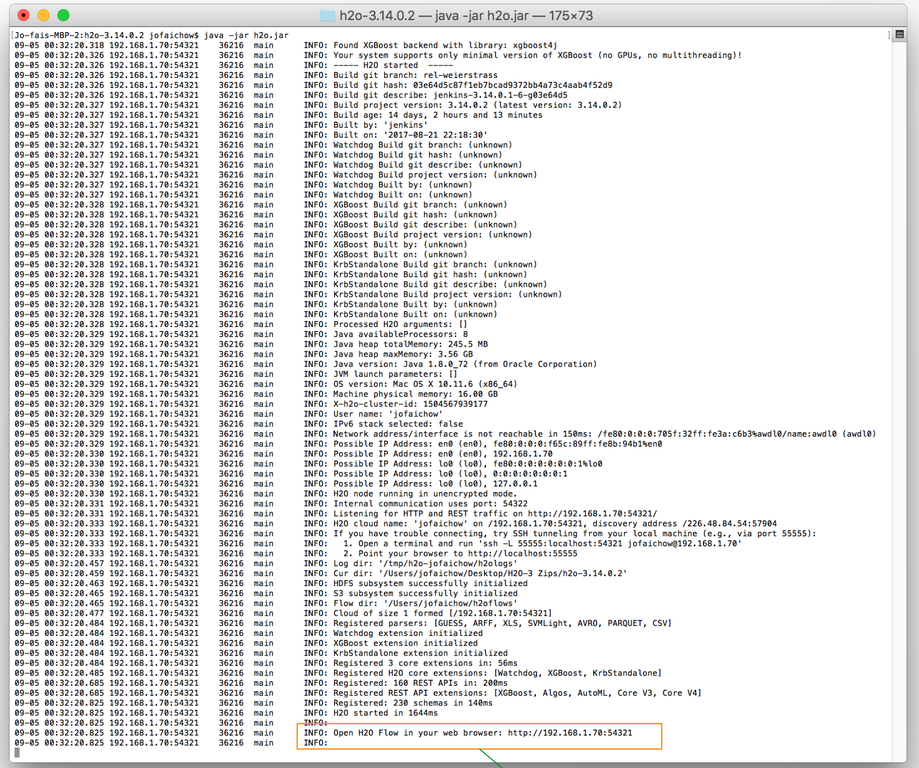
H2O.ai

Install and Start H_2O Flow (Web Interface)









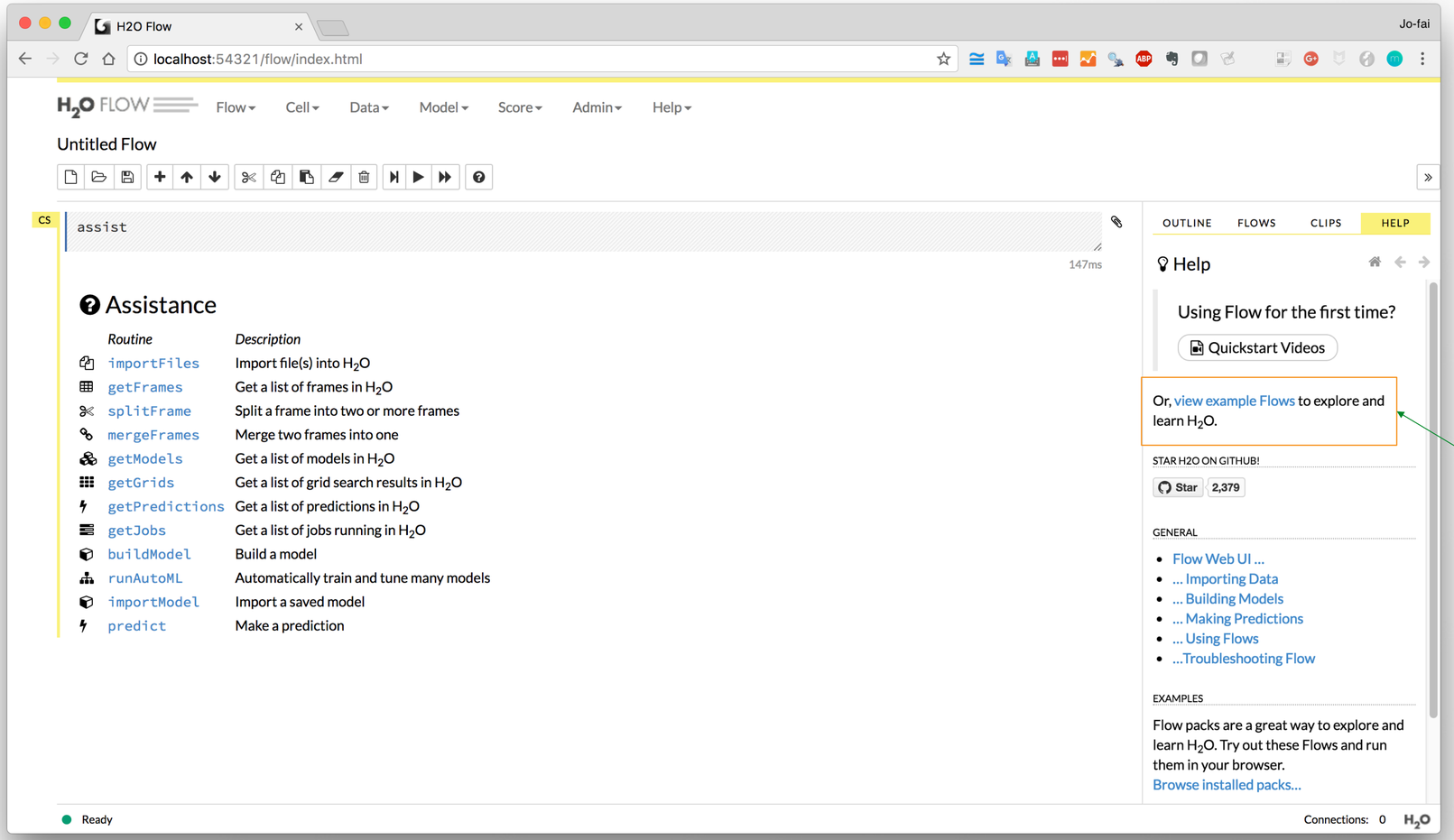
INFO: Open H2O Flow in your web browser: http://192.168.1.70:54321

H_2O.ai

How to launch H2O locally

- Use standalone version
  - java –jar h2o.jar
- Python Client
  - pip install h2o # installs from PyPi
  - import h2o
- R Client
  - install.packages("h2o") # installs from CRAN
  - library(h2o)
- When in doubt use http://h2o.ai/download

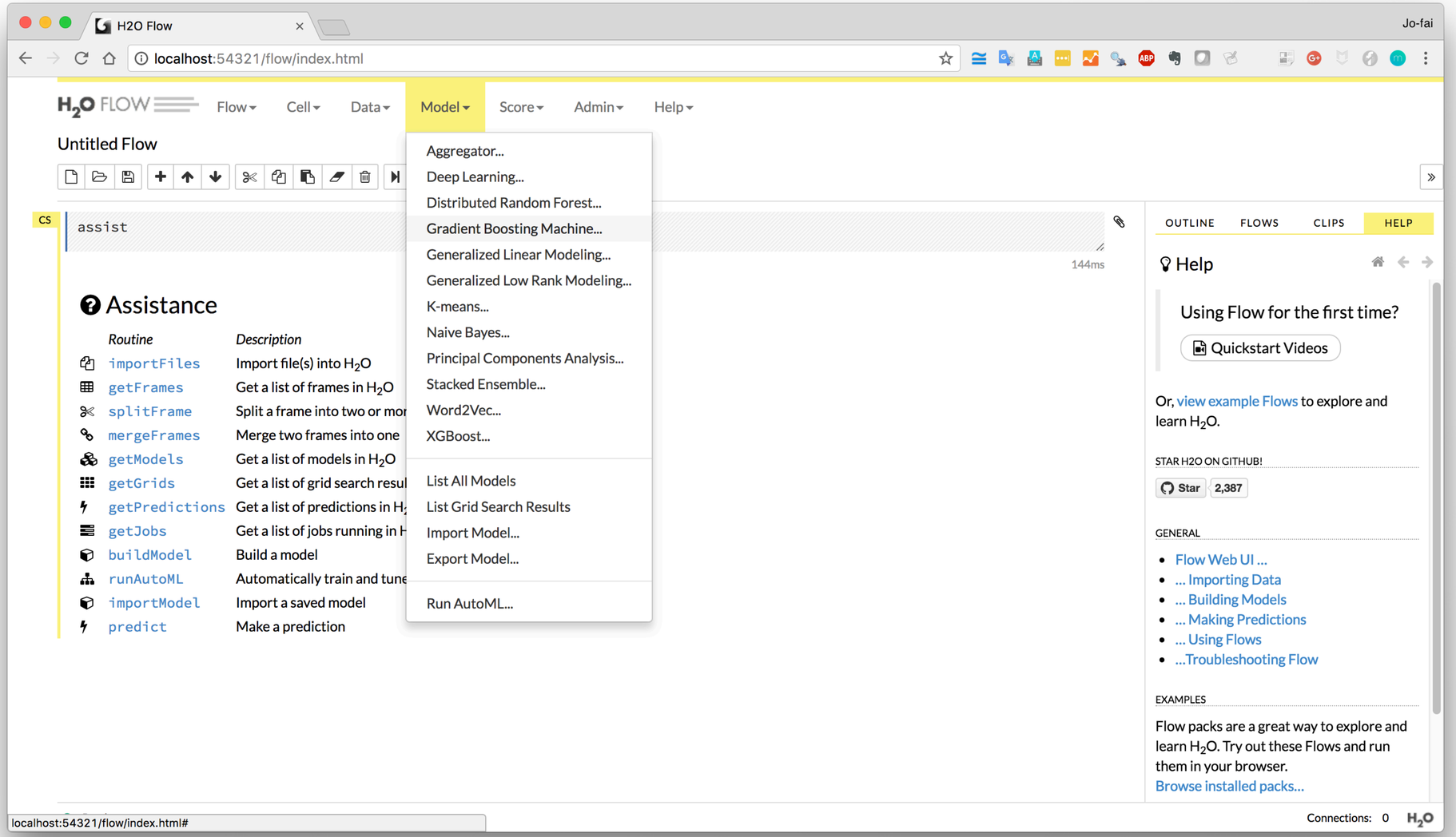
H_2O Flow (Web Interface)



More Examples

H_2O.ai

H_2O Flow (Web)



H_2O.ai

H_2O + R

```
# ------------------------------------------------------------
# Train a H2O Model
# ------------------------------------------------------------

# Train three basic H2O models
model_drf <- h2o.randomForest(x = features,
                              y = target,
                              model_id = "iris_random_forest",
                              training_frame = d_iris)

model_gbm <- h2o.gbm(x = features,
                     y = target,
                     model_id = "iris_gbm",
                     training_frame = d_iris)

model_dnn <- h2o.deeplearning(x = features,
                              y = target,
                              model_id = "iris_deep_learning",
                              training_frame = d_iris)
```

H_2O + Python

Gradient Boosting Machines

```
# Build a Gradient Boosting Machines (GBM) model with default settings

# Import the function for GBM
from h2o.estimators.gbm import H2OGradientBoostingEstimator

# Set up GBM for regression
# Add a seed for reproducibility
gbm_default = H2OGradientBoostingEstimator(model_id = 'gbm_default', seed = 1234)

# Use .train() to build the model
gbm_default.train(x = features,
                  y = 'quality',
                  training_frame = wine_train)
```

```
gbm Model Build progress: |████████████████████████████████████████████████| 100%
```

avkash@h2o.ai