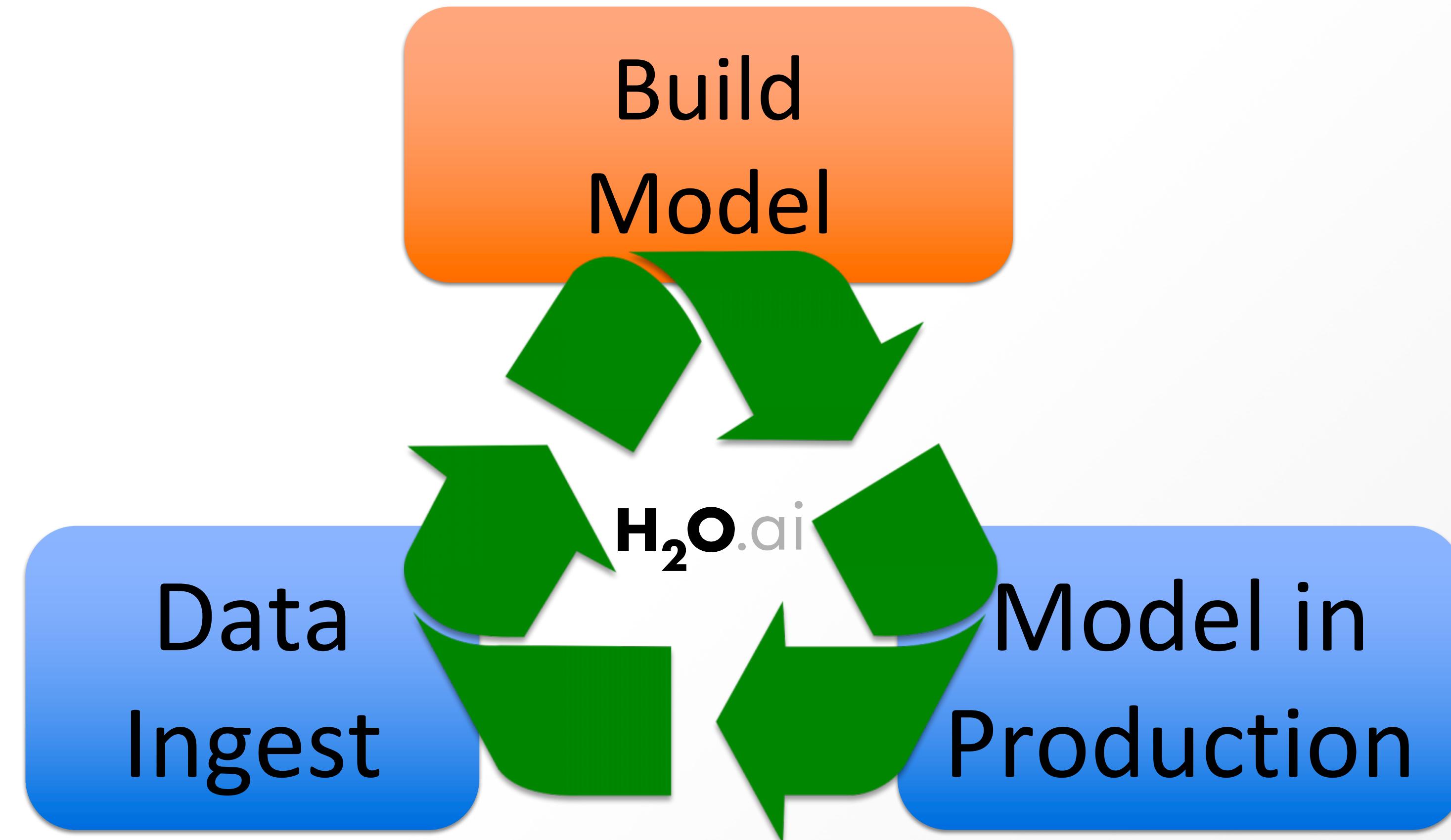


Machine Learning with H₂O

H₂O.ai

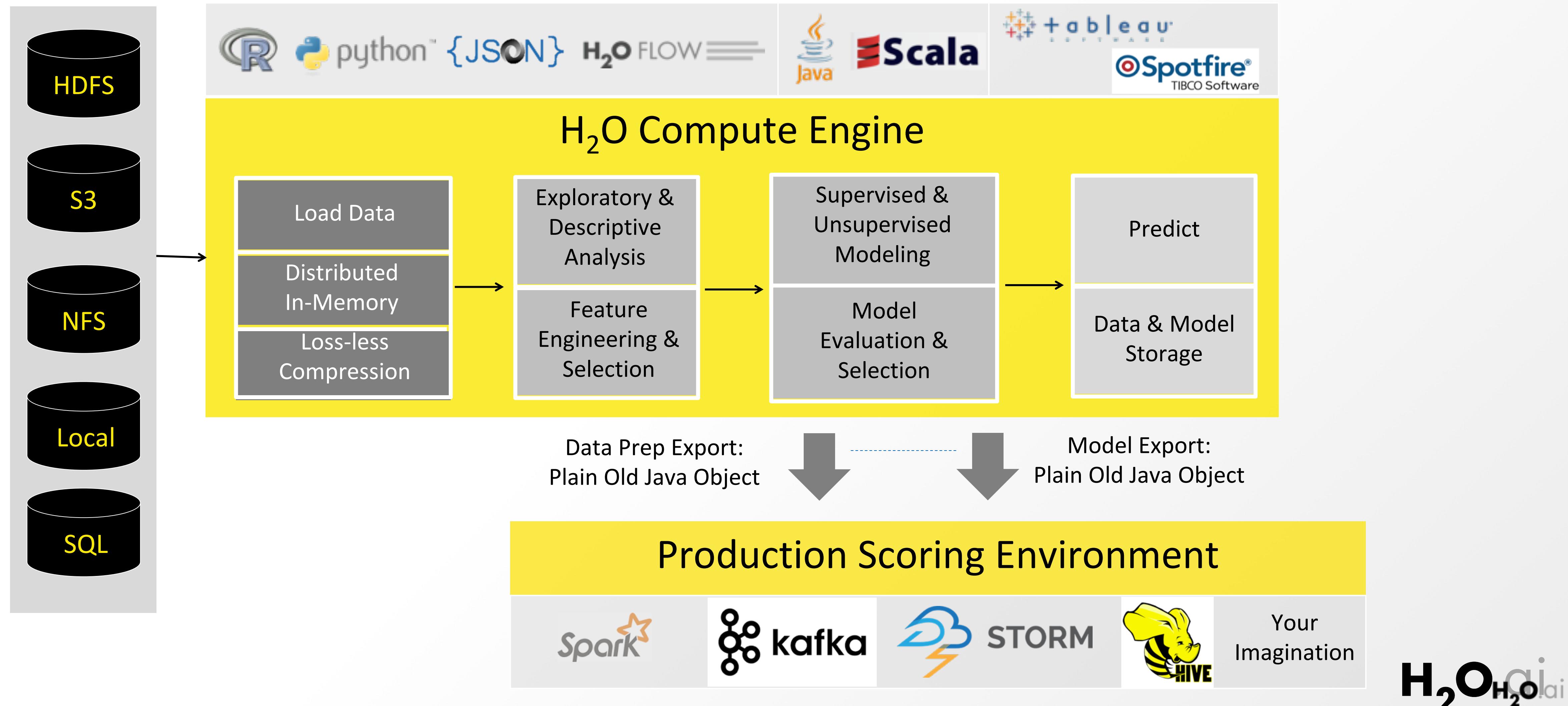
Machine Learning Process



H₂O MACHINE LEARNING PLATFORM

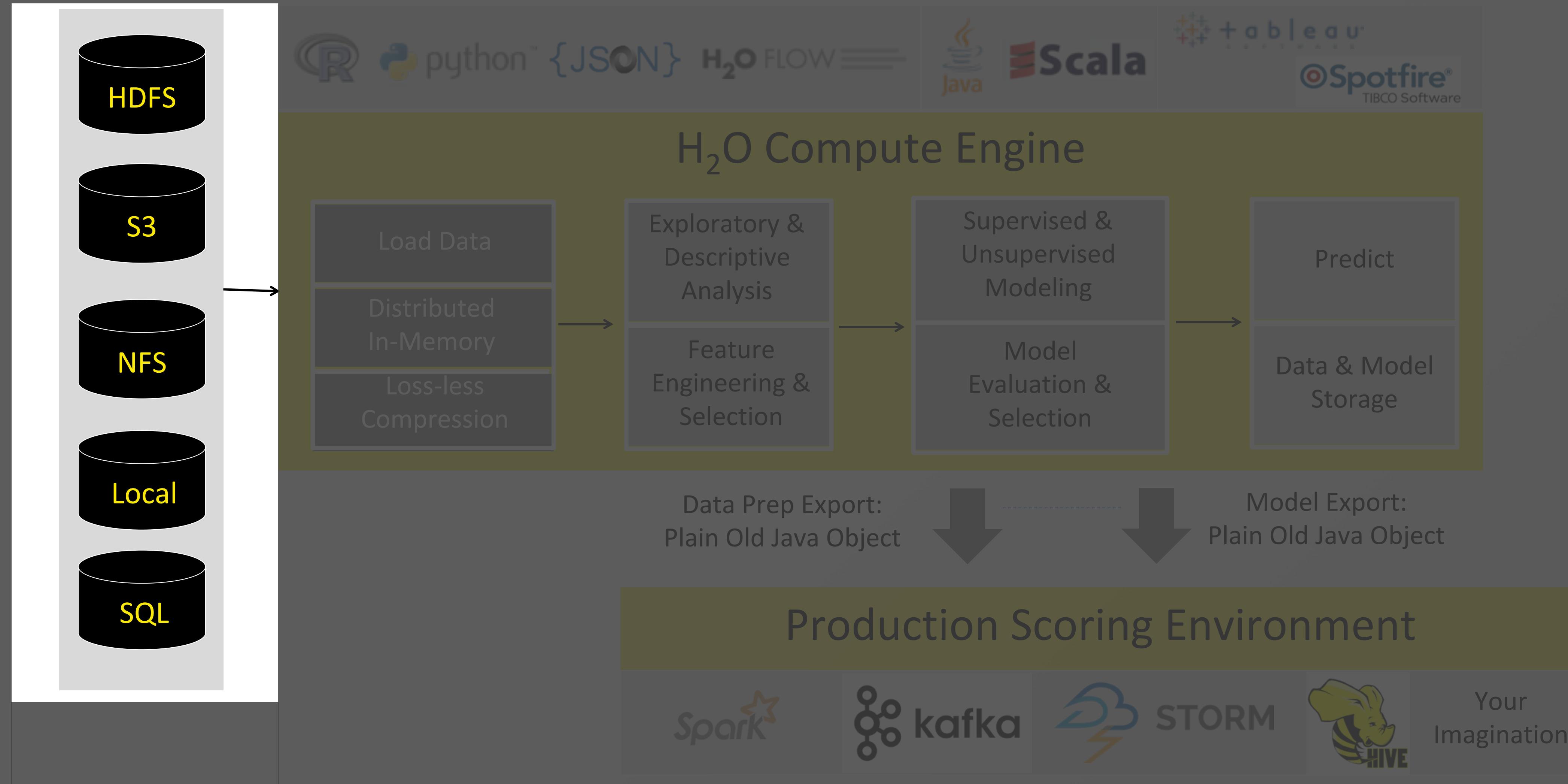
H₂O.ai

High Level Architecture



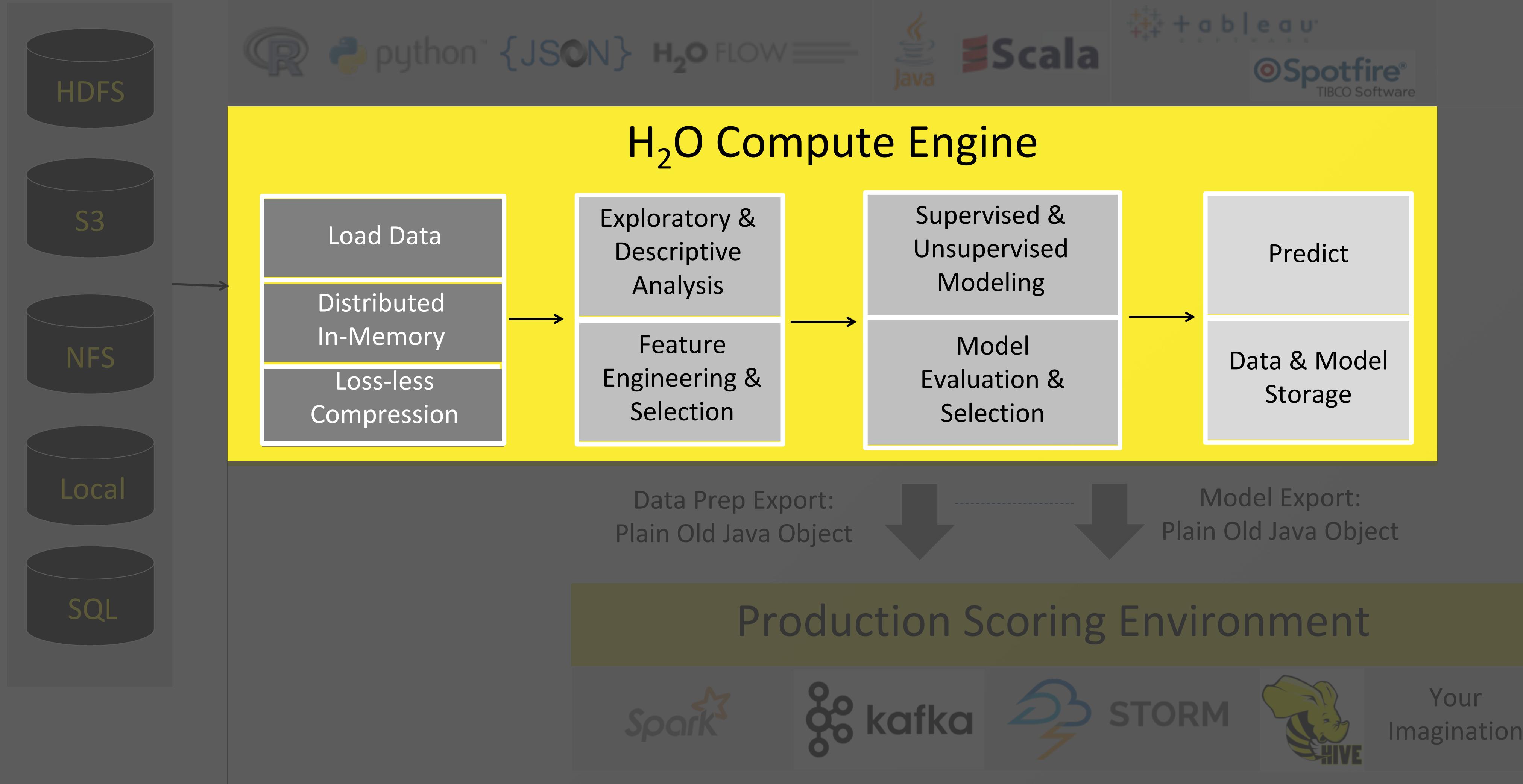
Import Data from Multiple Sources

High Level Architecture



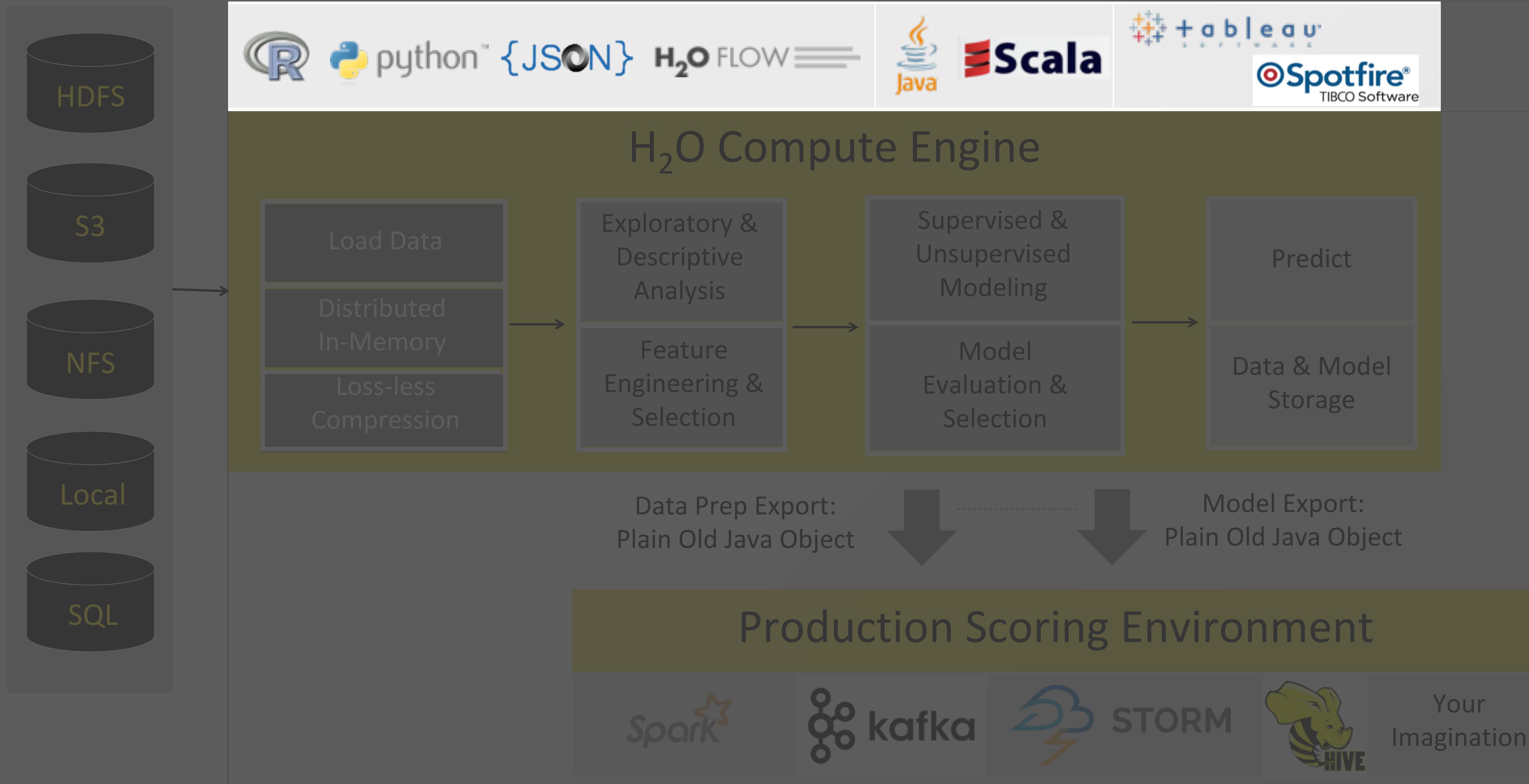
High Level Archi

Fast, Scalable & Distributed
Compute Engine Written in
Java



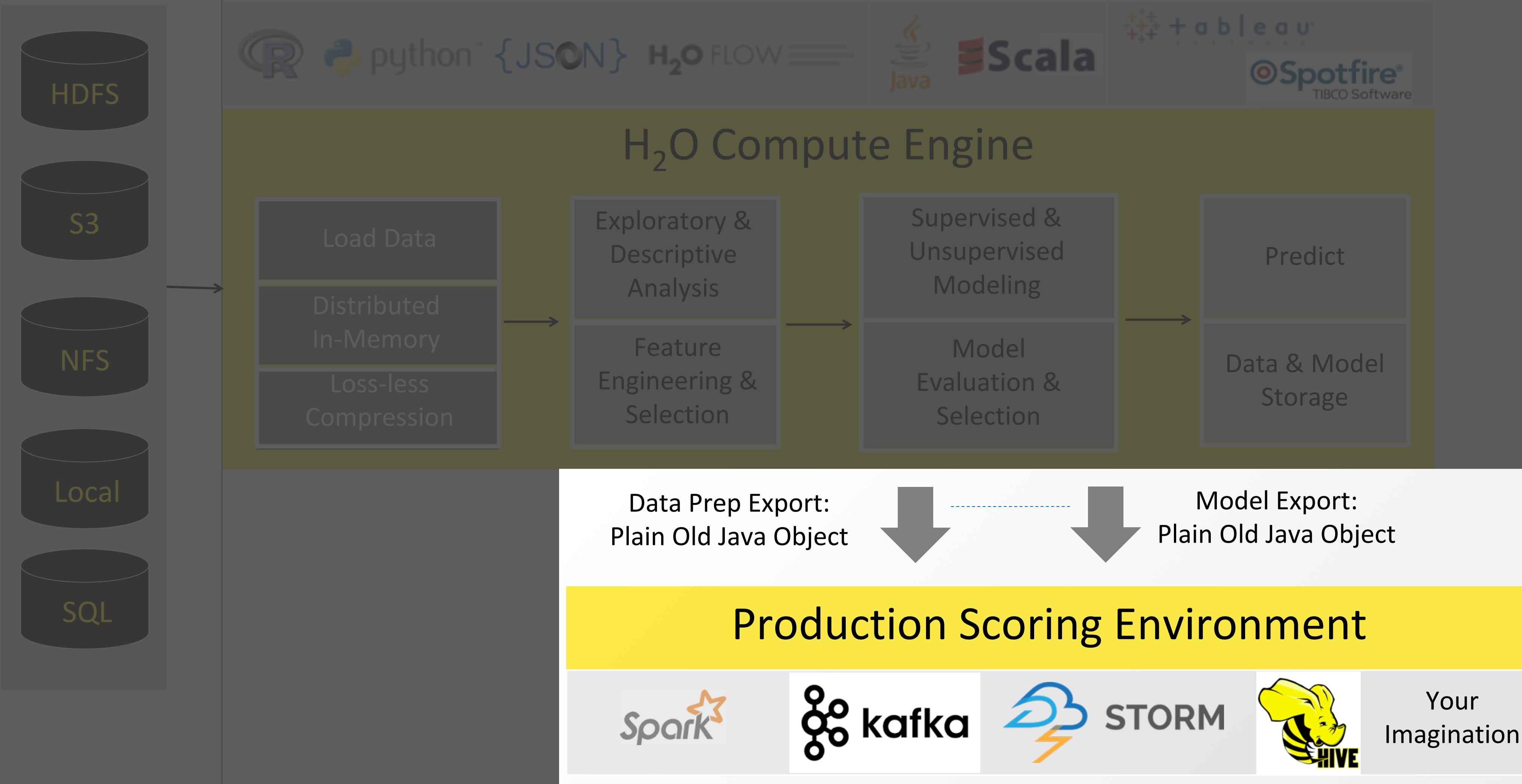
Multiple Interfaces

High Level Architecture



Export Standalone Models for Production

High Level Archi



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

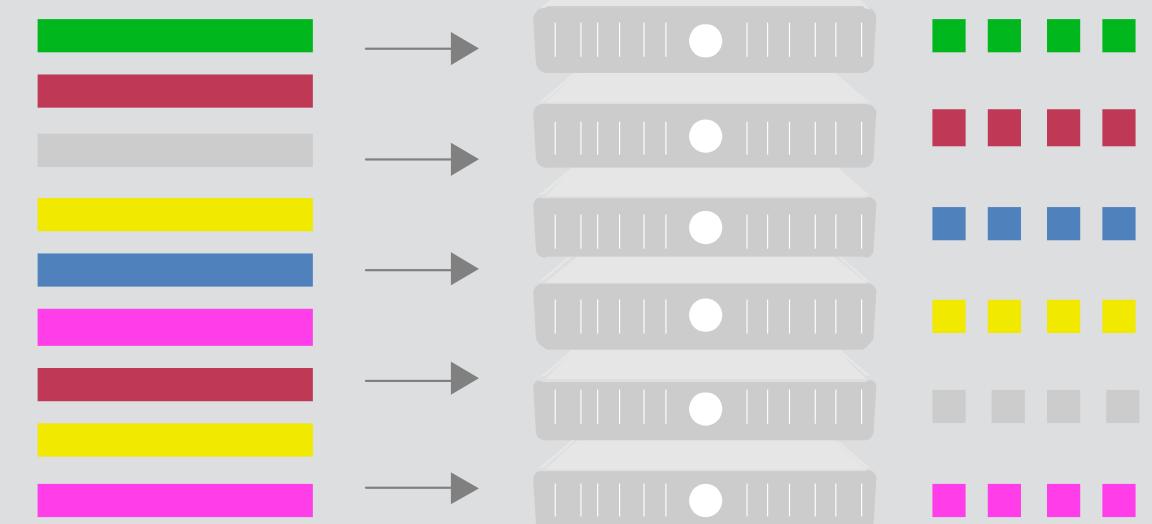
- **Principal Component Analysis:** Linearly transforms correlated variables to independent components
- **Generalized Low Rank Models:** extend the idea of PCA to handle arbitrary data consisting of numerical, Boolean, categorical, and missing data

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 H₂O: 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**

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

Getting Started & User Guides

H₂O

[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 H₂O](#)

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

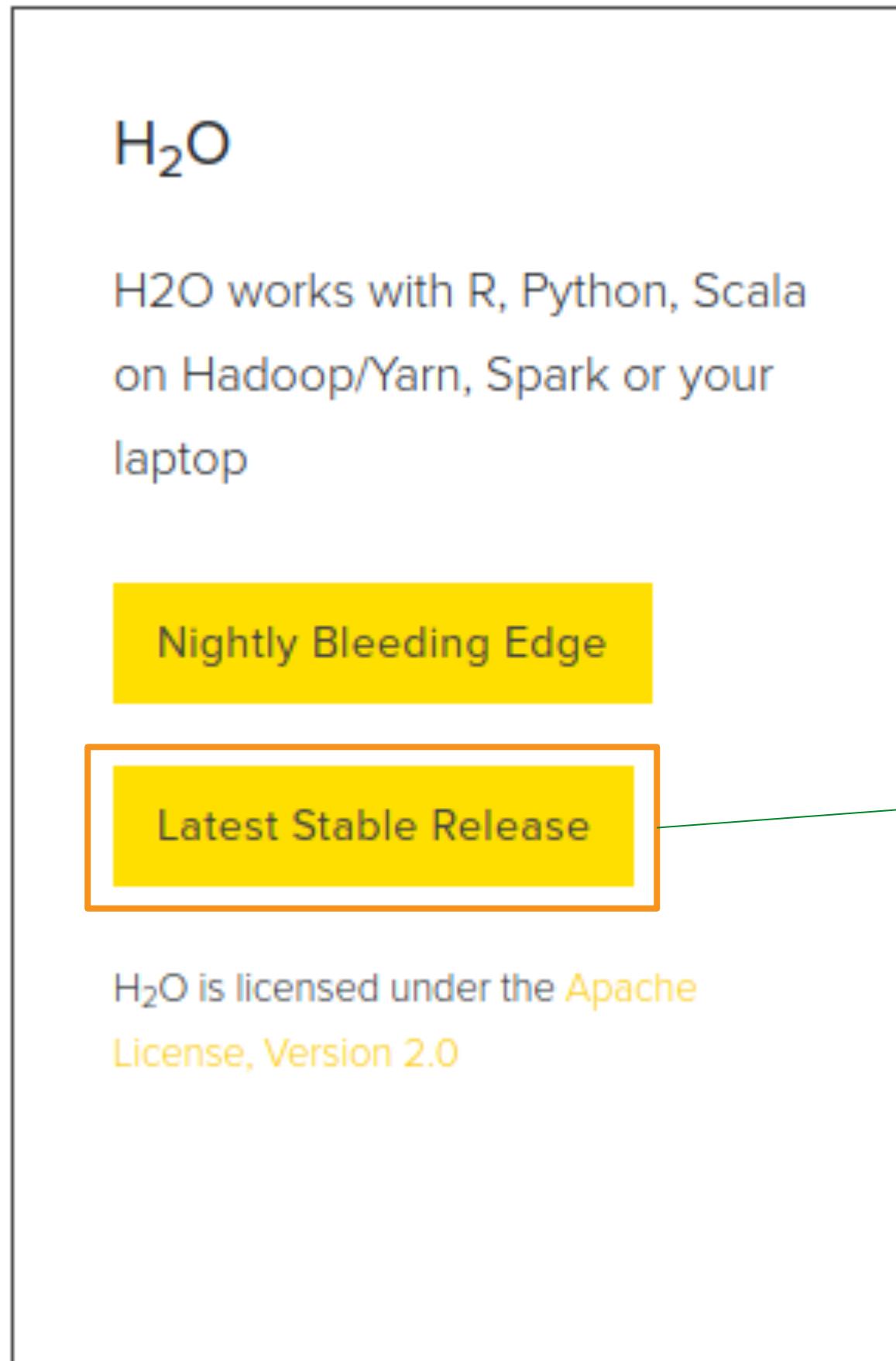
Miscellaneous

Word2vec	Tutorial	Reference
----------	--------------------------	---------------------------

TOP

H₂O_{ai}

Prerequisite: Java version 1.7+



H₂O
Version 3.14.0.2

Fast Scalable Machine Learning API
For Smarter Applications

DOWNLOAD AND RUN INSTALL IN R INSTALL IN PYTHON INSTALL ON HADOOP USE FROM MAVEN

DOWNLOAD H₂O

Get started with H₂O in 3 easy steps

1. Download H₂O. This is a zip file that contains everything you need to get started.
2. From your terminal, run:

```
cd ~/Downloads
unzip h2o-3.14.0.2.zip
cd h2o-3.14.0.2
java -jar h2o.jar
```
3. Point your browser to <http://localhost:54321>

Install and Start H₂O in R / Python

Left Screenshot (R):

Right Screenshot (Python):

Use H₂O directly from R

Copy and paste these commands into R one line at a time:

```
# The following two commands remove any previously installed H2O packages for R.
if ("package:h2o" %in% search()) { detach("package:h2o", unload=TRUE) }
if ("h2o" %in% rownames(installed.packages())) { remove.packages("h2o") }

# Next, we download packages that H2O depends on.
pkgs <- c("statmod", "RCurl", "jsonlite")
for (pkg in pkgs) {
  if (! (pkg %in% rownames(installed.packages()))) { install.packages(pkg) }
}

# Now we download, install and initialize the H2O package for R.
install.packages("h2o", type="source", repos="https://h2o-release.s3.amazonaws.com/h2o/rel-weierstrass/2/R")

# Finally, let's load H2O and start up an H2O cluster
library(h2o)
h2o.init()
```

Use H₂O directly from Python

1. Prerequisite: Python 2.7 or 3.5+
2. Install dependencies (prepending with `sudo` if needed):

```
pip install requests
pip install tabulate
pip install scikit-learn
pip install colorama
pip install future
```

At the command line, copy and paste these commands one line at a time:

```
# The following command removes the H2O module for Python.
pip uninstall h2o

# Next, use pip to install this version of the H2O Python module.
pip install https://h2o-release.s3.amazonaws.com/h2o/rel-weierstrass/2/Python/h2o-3.14.0.2-py2.py3-none-any.whl
```

Install and Start H₂O Flow (Web Interface)

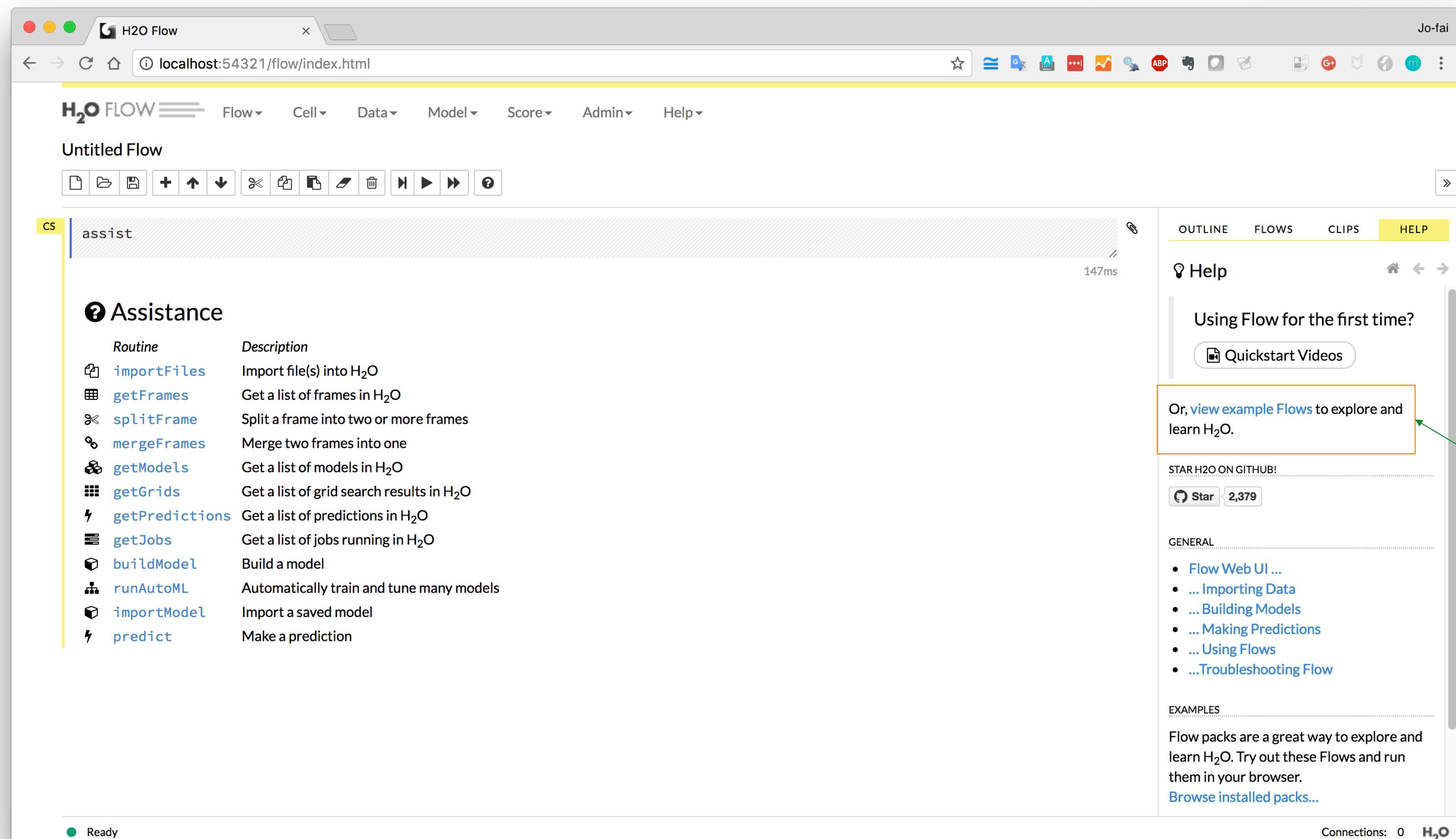
The diagram illustrates the process of installing and starting the H₂O Flow web interface. It consists of four main components:

- ZIP File:** A file named "h2o-3.14.0.2.zip" is shown, representing the downloaded software package.
- File Explorer:** A Mac OS X Finder window displays the contents of the extracted directory "h2o-3.14.0.2". Inside, there are several sub-folders: "bindings", "h2o.jar" (which is highlighted with a red box), "python", and "R".
- Terminal Window:** A terminal window titled "h2o-3.14.0.2 — bash — 64x26" shows the command "java -jar h2o.jar" being entered.
- Log Window:** A terminal window titled "h2o-3.14.0.2 — java -jar h2o.jar — 175x73" shows the application's startup logs. Key messages include:
 - "INFO: Found XGBoost backend with library: xgboost4j"
 - "INFO: Your system supports only minimal version of XGBoost (no GPUs, no multithreading)!"
 - "INFO: --- H2O started ---"
 - "INFO: Build git branch: rel-weierstrass"
 - "INFO: Build git hash: 03e84dc5c8fb7cad9372bb4a73c4aab4f52d9"
 - "INFO: Build project version: 3.14.0.2 (latest version: 3.14.0.2)"
 - "INFO: Build age: 14 days, 2 hours and 13 minutes"
 - "INFO: Built by: 'jenkins'"
 - "INFO: Built on: '2017-08-21 22:18:30'"
 - "INFO: Watchdog Build git branch: (unknown)"
 - "INFO: Watchdog Build git hash: (unknown)"
 - "INFO: Watchdog Build git describe: (unknown)"
 - "INFO: Watchdog Build project version: (unknown)"
 - "INFO: Watchdog Build by: (unknown)"
 - "INFO: Watchdog Built on: (unknown)"
 - "INFO: XGBoost Build git branch: (unknown)"
 - "INFO: XGBoost Build git hash: (unknown)"
 - "INFO: XGBoost Build git describe: (unknown)"
 - "INFO: XGBoost Build project version: (unknown)"
 - "INFO: XGBoost Built by: (unknown)"
 - "INFO: XGBoost Built on: (unknown)"
 - "INFO: KrbStandalone Build git branch: (unknown)"
 - "INFO: KrbStandalone Build git hash: (unknown)"
 - "INFO: KrbStandalone Build git describe: (unknown)"
 - "INFO: KrbStandalone Build project version: (unknown)"
 - "INFO: KrbStandalone Built by: (unknown)"
 - "INFO: KrbStandalone Built on: (unknown)"
 - "INFO: Processed H2O arguments: []"
 - "INFO: Java available processors: 8"
 - "INFO: Java heap totalMemory: 145.5 MB"
 - "INFO: Java heap maxMemory: 145.5 GB"
 - "INFO: Java version: Java 1.8.0_21 (from Oracle Corporation)"
 - "INFO: JVM launch parameters: []"
 - "INFO: OS version: Mac OS X 10.11.6 (x86_64)"
 - "INFO: Possible IP Address: en0 (en0), fe80:0:0:0:f65c:89ff:fe80:94b1%en0"
 - "INFO: Machine physical memory: 16.00 GB"
 - "INFO: X-h2o-cluster-id: 1504567939177"
 - "INFO: User name: 'jofaichow'"
 - "INFO: IPv6 stack selected: false"
 - "INFO: Network address/interface is not reachable in 150ms: /fe80:0:0:0:70ff:fe3a:c6b3%awd10 (awd10)"
 - "INFO: Possible IP Address: en0 (en0), fe80:0:0:0:f65c:89ff:fe80:94b1%en0"
 - "INFO: Possible IP Address: en0 (en0), 192.168.1.70"
 - "INFO: Possible IP Address: lo0 (lo0), 0.0.0.0:0:1"
 - "INFO: Possible IP Address: lo0 (lo0), 127.0.0.1:100"
 - "INFO: H2O node running in unencrypted mode."
 - "INFO: Internal communication uses port: 54322"
 - "INFO: Listening for HTTP and REST traffic on http://192.168.1.70:54321/
 - "INFO: H2O cloud name: 'jofaichow' on /192.168.1.70:54321, discovery address /226.48.84.54:57904"
 - "INFO: If you have trouble connecting, try SSH tunneling from your local machine (e.g., via port 55555): 1. Open a terminal and run 'ssh -L 55555:localhost:54321 jofaichow@192.168.1.70'
 - "INFO: 2. Point your browser to http://localhost:55555"
 - "INFO: Log dir: '/tmp/h2o-jofaichow/h2o-logs'
 - "INFO: Cur dir: '/Users/jofaichow/Desktop/H2O-3 Zips/h2o-3.14.0.2'
 - "INFO: HDFS subsystem successfully initialized"
 - "INFO: HDFS subsystem successfully initialized"
 - "INFO: Flow dir: '/Users/jofaichow/h2oflow'
 - "INFO: Cloud of size 1 formed [/192.168.1.70:54321]
 - "INFO: Registered parsers: [GUESS, ARFF, XLS, SVMLight, AVRO, PARQUET, CSV]
 - "INFO: Watchdog extension initialized"
 - "INFO: XGBoost extension initialized"
 - "INFO: KrbStandalone extension initialized"
 - "INFO: Registered 3 core extensions in: 56ms"
 - "INFO: Registered H2O core extensions: [Watchdog, XGBoost, KrbStandalone]
 - "INFO: Registered: 160 REST APIs in: 200ms"
 - "INFO: Registered: 230 schemas in 140ms"
 - "INFO: H2O started in 164ms"
 - "INFO: Open H2O Flow in your web browser: http://192.168.1.70:54321"
 - "INFO:"

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>

H₂O Flow (Web Interface)



More
Examples

H₂O Flow (Web)

The screenshot shows the H2O Flow (Web) interface running in a web browser. The title bar reads "H2O Flow". The main menu bar includes "Flow", "Cell", "Data", "Model" (which is currently selected), "Score", "Admin", and "Help". Below the menu is a toolbar with various icons for file operations like import, export, and model building. The main workspace is titled "Untitled Flow" and contains a search bar with the text "assist". A sidebar on the left lists "Assistance" routines with their descriptions:

Routine	Description
importFiles	Import file(s) into H ₂ O
getFrames	Get a list of frames in H ₂ O
splitFrame	Split a frame into two or more
mergeFrames	Merge two frames into one
getModels	Get a list of models in H ₂ O
getGrids	Get a list of grid search results
getPredictions	Get a list of predictions in H ₂ O
getJobs	Get a list of jobs running in H ₂ O
buildModel	Build a model
runAutoML	Automatically train and tune a model
importModel	Import a saved model
predict	Make a prediction

A dropdown menu under "Model" lists various machine learning models and utilities:

- Aggregator...
- Deep Learning...
- Distributed Random Forest...
- Gradient Boosting Machine...
- Generalized Linear Modeling...
- Generalized Low Rank Modeling...
- K-means...
- Naive Bayes...
- Principal Components Analysis...
- Stacked Ensemble...
- Word2Vec...
- XGBoost...

The right side of the interface features a "Help" section with links to "Quickstart Videos", "example Flows", and "STAR H2O ON GITHUB!". It also includes sections for "GENERAL" (Flow Web UI, Importing Data, Building Models, Making Predictions, Using Flows, Troubleshooting Flow) and "EXAMPLES" (Flow packs for exploring H₂O). The bottom status bar shows the URL "localhost:54321/flow/index.html#" and "Connections: 0".

H₂O + 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)
```

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