

R과 H2O로 시작하는 머신러닝

목차

- H2O는 무엇인가?
 - 소개
 - 대상
 - 구조
 - 기능
 - 장/단점
- 공지사항
- 실습
- 자주 나오는 질문들

H₂O는 무엇인가?

**머신러닝, 딥러닝 등 관련된 이야기는 많이
들어보셨지요?**

H₂O는 무엇인가?

패키지만 한번 나열해볼까요?

H2O는 무엇인가?



DEEPLARNING4J



H₂O는 무엇인가?

위 패키지들의 특징은?

H2O는 무엇인가?

- **Backpropagation**을 자동으로 해준다.
- **Custom Modeling**이 가능하다.
- **GPU**를 사용한다.
- 기타 등등

H₂O는 무엇인가?

그런데 한가지 어려운 점이 있어요

H₂O는 무엇인가?

생각보다 많은 코드가 들어가요

H2O는 무엇인가?

Tensorflow로 예를 들어볼까요?

H2O는 무엇인가?

```
1 import numpy as np
2 import tensorflow as tf
3 from tensorflow import layers
4 from tensorflow.examples.tutorials.mnist import input_data
5
6 mnist = input_data.read_data_sets("../MNIST_data/", one_hot=True)
7 num_out = 10
8
9 x = tf.placeholder(tf.float32, [None, 784])
10 y = tf.placeholder(tf.int32)
11
12 h1 = layers.dense(x, 200, activation=tf.nn.relu)
13 h2 = layers.dense(x, 200, activation=tf.nn.relu)
14 out = layers.dense(x, num_out, activation=None)
15
16 loss = tf.reduce_mean(tf.losses.softmax_cross_entropy(onehot_labels=y, logits=out))
17 optim = tf.train.AdamOptimizer(learning_rate=1e-4)
18 train_op = optim.minimize(loss)
19
20 correct_prediction = tf.equal(tf.argmax(out,1), tf.argmax(y,1))
21 accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
22
23 sess = tf.InteractiveSession()
24 init = tf.global_variables_initializer()
25 init.run()
26
27 num_epoch = 20
28 batch_size = 100
29 num_iter_in_epoch = mnist.train.num_examples // batch_size
30
31 for i in range(num_epoch):
32     train_cp = []
33     for _ in range(num_iter_in_epoch):
34         batch_xs, batch_ys = mnist.train.next_batch(100)
35         sess.run(train_op, feed_dict={x: batch_xs, y: batch_ys})
36         train_cp += sess.run([correct_prediction], feed_dict={x: batch_xs, y: batch_ys})
37     print("Epoch : {} | Train Accuracy : {}".format(i, np.mean(train_cp)))
38
39 print("\nFinally, Test Accuracy : {}".format(sess.run(accuracy, feed_dict={x: mnist.test.images, y: mnist.test.labels})))
```

Tensorflow로 짜보는 Deep Neural Network for MNIST
(version 1 : layers 이용)

H₂O는 무엇인가?

물론 Estimator 라는 편한 모듈이 있습니다.

H2O는 무엇인가?

```
1 import numpy as np
2 import tensorflow as tf
3 from tensorflow import layers
4 from tensorflow.examples.tutorials.mnist import input_data
5
6 mnist = input_data.read_data_sets("../MNIST_data/", one_hot=False)
7 feature_columns = [tf.feature_column.numeric_column("x", shape=[28, 28])]
8
9 classifier = tf.estimator.DNNClassifier(feature_columns=feature_columns, hidden_units=[200, 200], optimizer=tf.train.AdamOptimizer(1e-4), n_classes=10, dropout=0.5, model_dir="./tmp/m")
10
11 def input(dataset):
12     return dataset.images, dataset.labels.astype(np.int32)
13
14 train_input_fn = tf.estimator.inputs.numpy_input_fn(x={"x": input(mnist.train)[0]}, y=input(mnist.train)[1], num_epochs=None, batch_size=100, shuffle=True)
15 test_input_fn = tf.estimator.inputs.numpy_input_fn(x={"x": input(mnist.test)[0]}, y=input(mnist.test)[1], num_epochs=1, shuffle=False)
16
17 classifier.train(input_fn=train_input_fn, steps=10000)
18
19 accuracy_score = classifier.evaluate(input_fn=test_input_fn)["accuracy"]
20 print("\nTest Accuracy: {0:f}%\n".format(accuracy_score*100))
```

**Tensorflow로 짜보는 Deep Neural Network for MNIST
(version 2 : Estimator 이용)**

H2O는 무엇인가?

**Q : 그래서 뭘 말하고 싶은 건데요?
코드 많지도 않은데요?**

H₂O는 무엇인가?

A : 그럼 이건 어때요?

H2O는 무엇인가?

```
1 library(h2o)
2 h2o.init(nthreads = -1)
3
4 train_file <- "https://h2o-public-test-data.s3.amazonaws.com/bigdata/laptop/mnist/train.csv.gz"
5 test_file <- "https://h2o-public-test-data.s3.amazonaws.com/bigdata/laptop/mnist/test.csv.gz"
6 train <- h2o.importFile(train_file)
7 test <- h2o.importFile(test_file)
8
9 y <- "C785"
10 x <- setdiff(names(train), y)
11 train[,y] <- as.factor(train[,y])
12 test[,y] <- as.factor(test[,y])
13
14 model <- h2o.deeplearning(x = x, y = y, training_frame = train, validation_frame = test, distribution = "multinomial",
15   . . . . . activation = "Rectifier", hidden = c(200, 200), epochs = 1000)
16 summary(model)
```

H2O로 짜보는 Deep Neural Network for MNIST

H₂O는 무엇인가?

복잡한 구조가 아닌 간단한 Deep Neural Network 모델 생성에는 제법 괜찮아요.

H2O는 무엇인가?

오늘은 H2O라는 패키지를 소개해볼까합니다

H₂O는 무엇인가?

H₂O요?



H₂O는 무엇인가?

아니요. H₂O요!

H₂O.ai

H2O는 무엇인가? - 소개

H2O is an open source, in-memory, distributed, fast, and scalable machine learning and predictive analytics platform that allows you to build machine learning models on big data and provides easy productionalization of those models in an enterprise environment.

H2O's core code is written in Java. Inside H2O, a Distributed Key/Value store is used to access and reference data, models, objects, etc., across all nodes and machines. The algorithms are implemented on top of H2O's distributed Map/Reduce framework and utilize the Java Fork/Join framework for multi-threading. The data is read in parallel and is distributed across the cluster and stored in memory in a columnar format in a compressed way. H2O's data parser has built-in intelligence to guess the schema of the incoming dataset and supports data ingest from multiple sources in various formats.

H2O는 무엇인가? - 소개

**아래와 같은 특징을 지닌
머신 러닝 분석 플랫폼**

- 1) Open Source & Based Java**
- 2) In Memory**
- 3) Multi-threading**
- 4) Distributed Map/Reduce**

H2O는 무엇인가? - 소개

1) Open Source & Based Java

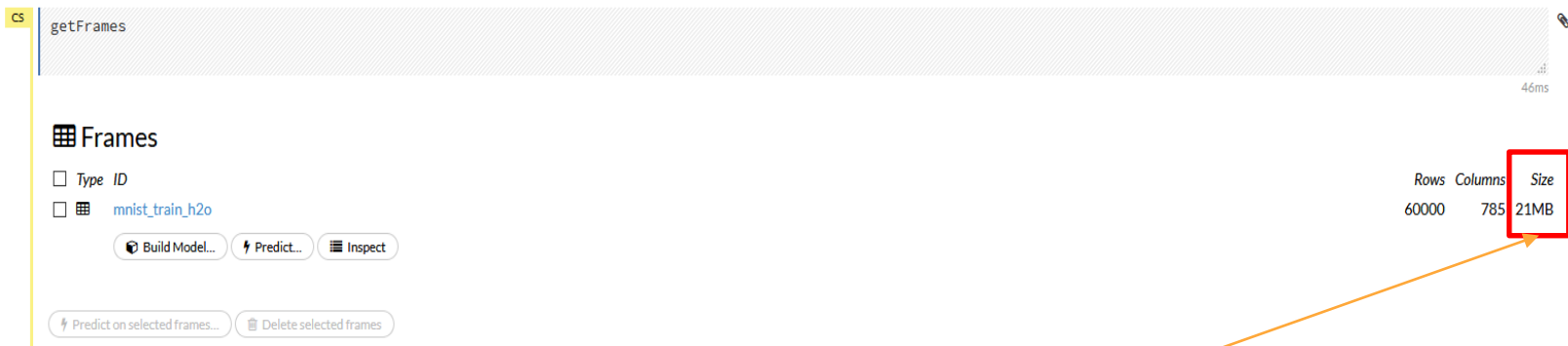
The left screenshot shows the GitHub repository for `h2oai/h2o-3`. It features a description of H2O as an "Open Source Fast Scalable Machine Learning Platform For Smarter Applications" and lists various machine learning models and frameworks it supports. The repository has 23,083 commits, 871 branches, 1,702 releases, and 107 contributors.

The right screenshot shows a specific commit in the `h2o-core` branch, specifically the `src/test/java/hex/HitRatioTest.java` file. The commit was made by `tomasnykodym` on May 21, 2015, and added observation weight to AUCBuilder. The file content is as follows:

```
1 package hex;
2
3 import static hex.ModelMetricsMultinomial.updateHits;
4 import org.junit.Assert;
5 import org.junit.BeforeClass;
6 import org.junit.Test;
7 import static water.TestUtil.stall_till_cloudsize;
8
9 import java.util.Arrays;
10
11 public class HitRatioTest {
12
13     @BeforeClass() public static void setup() { stall_till_cloudsize(1); }
14
15     @Test
16     public void testHits() {
17         double[] hits = new double[4];
18         double[] pred_dist;
19         int actual_label;
20
21         // No ties
22         // top 1
23         Arrays.fill(hits, 0);
24         actual_label = 0; pred_dist = new double[]{0.4f, .1f, .2f, .3f}; updateHits(1, actual_label, pred_dist, hits);
```

H2O는 무엇인가? - 소개

2) In Memory



CS getFrames 46ms

Frames

- Type ID
- mnist_train_h2o

Build Model... Predict... Inspect

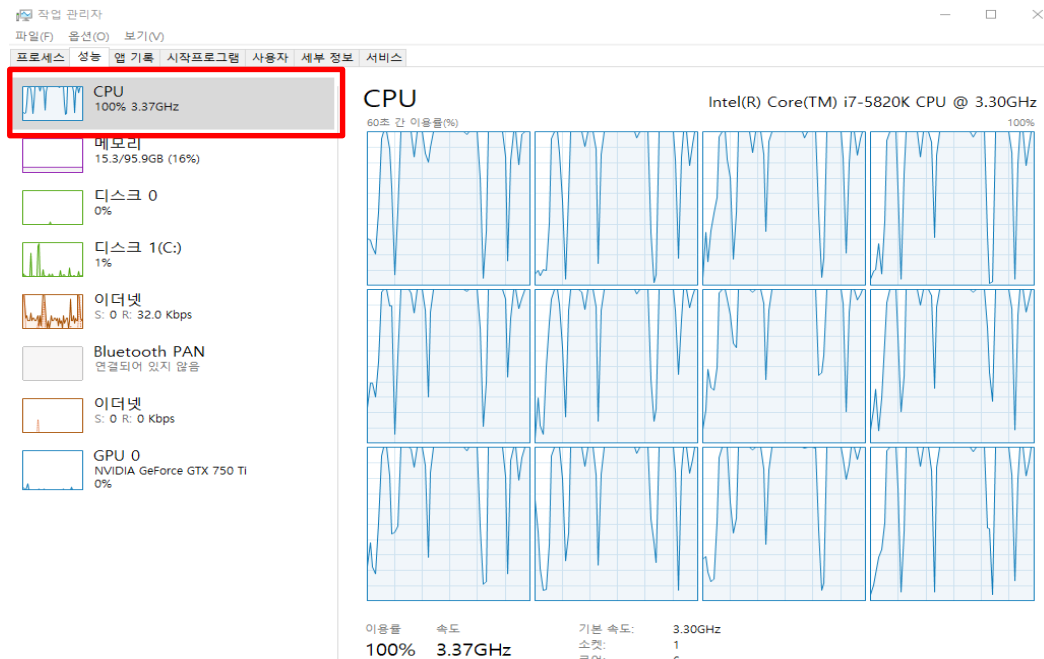
Predict on selected frames... Delete selected frames

Rows	Columns	Size
60000	785	21MB

21MB 사이즈의 데이터가 Java Virtual Machine상에 올라가 있음

H2O는 무엇인가? - 소개

3) Multi-Threading

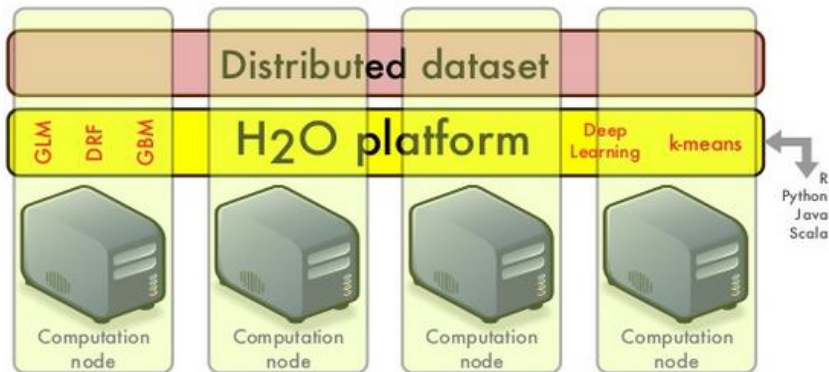


CPU의 이용률이 100%로 모든 thread를 활용할 수 있음

H2O는 무엇인가? - 소개

4) Distributed Map/Reduce

H2O platform



Oxdata

Random Forest, Gradient Boosting Machine, Deep Learning 등
여러 알고리즘들을 분산 처리 할 수 있게끔 구현하였음

H₂O는 무엇인가? - 대상

어떤 분들이 사용하면 좋을까요?

H2O는 무엇인가? - 대상

A ~ Z까지 전부 구현
일부 구현 + 패키지
Only 패키지

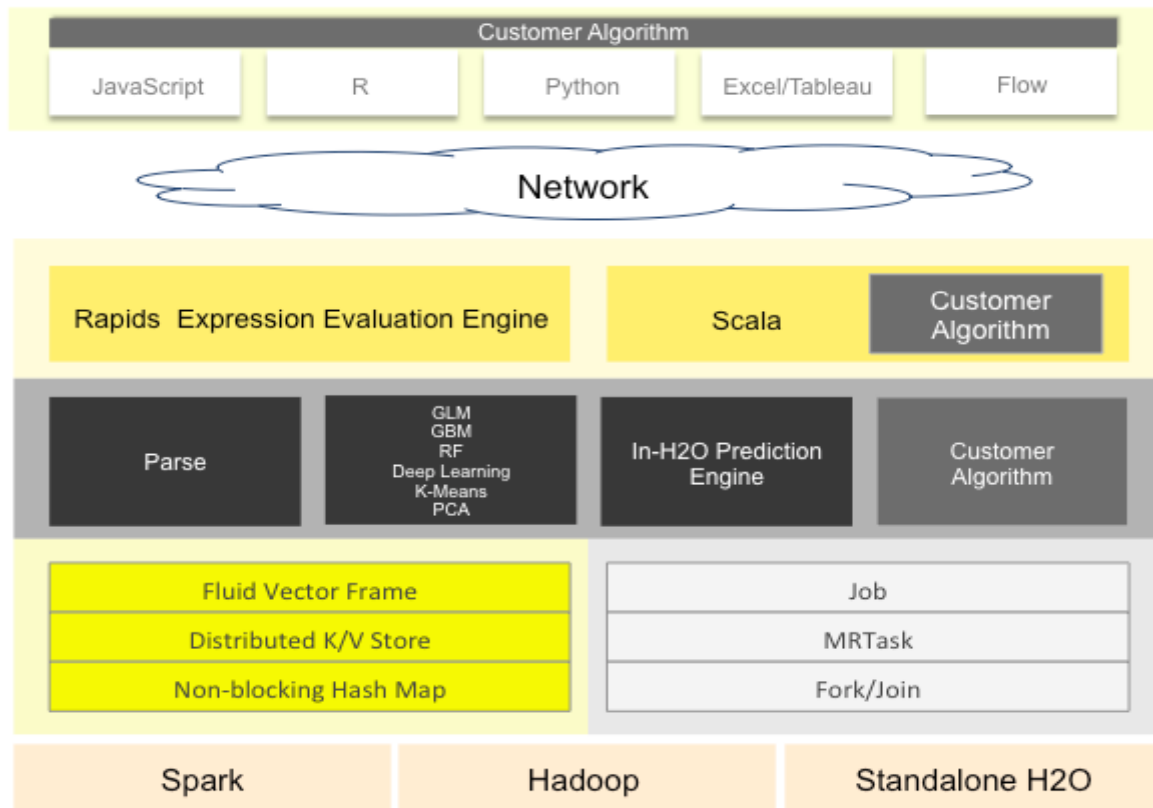
H2O는 무엇인가? - 대상

~~A ~ Z까지 전부 구현~~

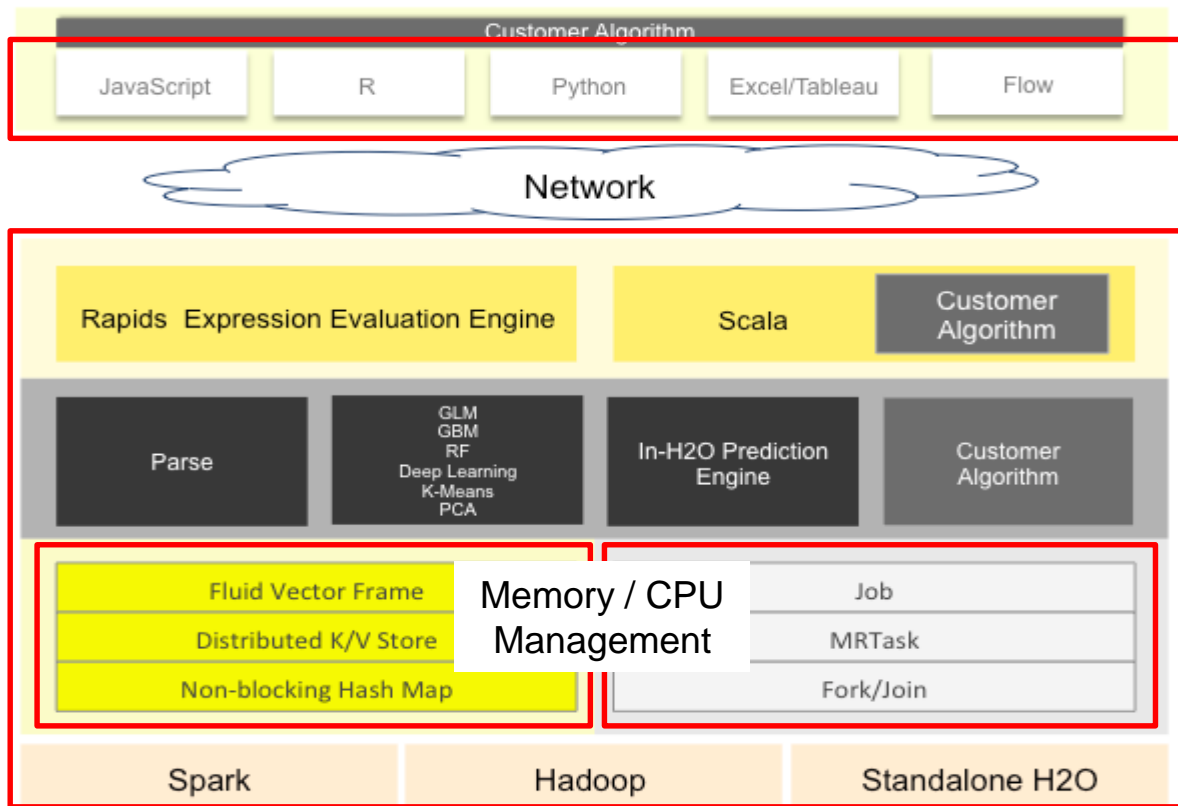
▲ 일부 구현 + 패키지

✓ Only 패키지

H2O는 무엇인가? - 구조



H2O는 무엇인가? - 구조



Rest API Clients

JVM Components

H₂O는 무엇인가? - 구조

R과 H₂O는 어떻게 연동하는 걸까요?

H2O는 무엇인가? - 구조

**HDFS에서 데이터를 가져오는 과정을 한번
살펴볼게요**

H2O는 무엇인가? - 구조

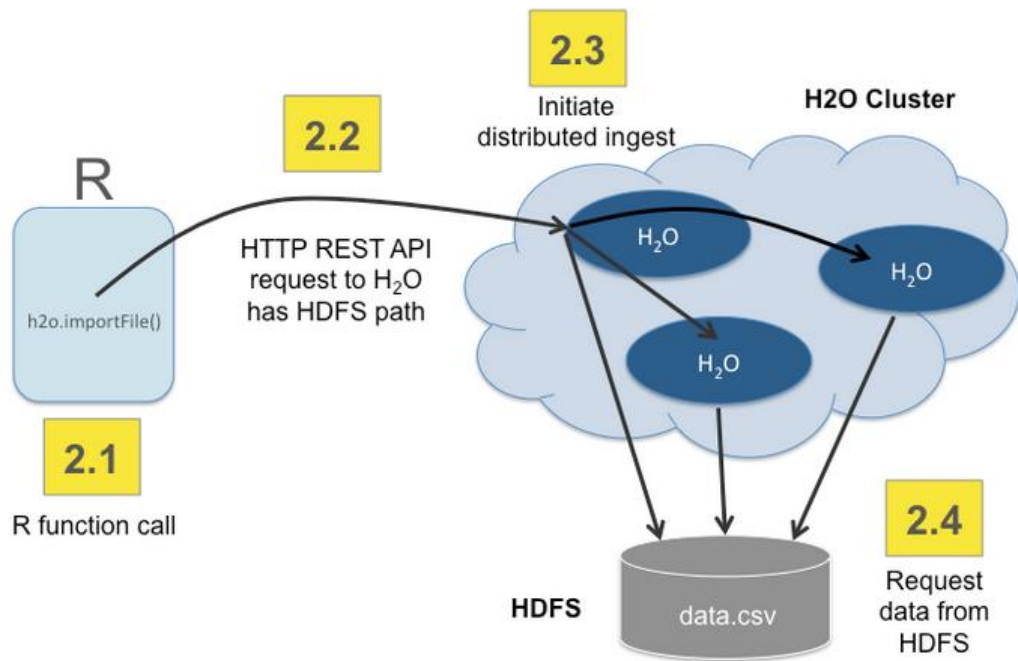


```
h2o_df = h2o.importFile("hdfs://path/to/data.csv")
```

R user

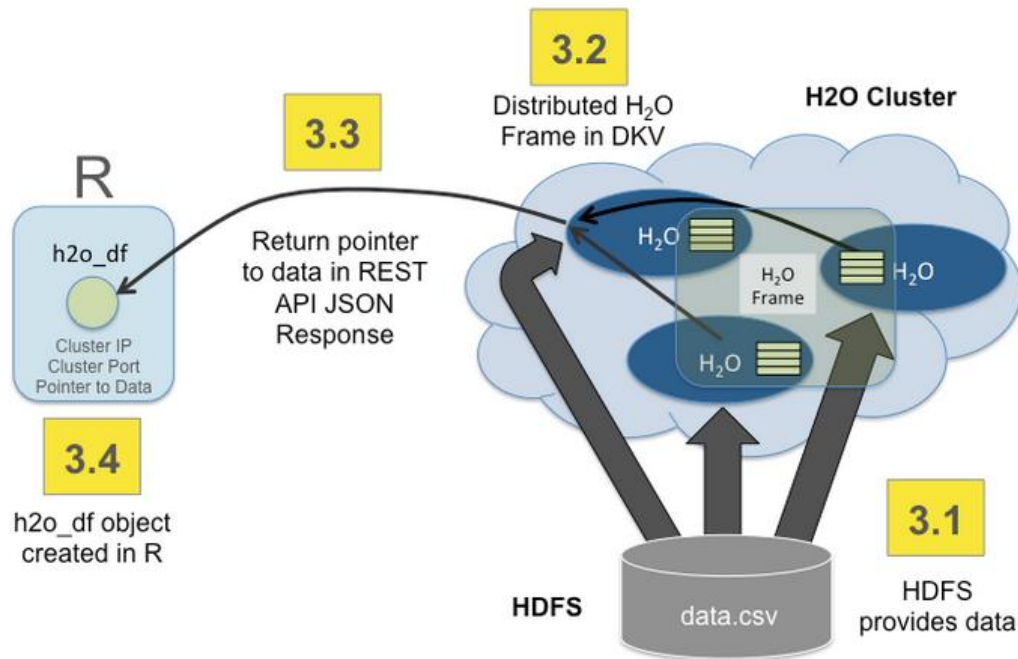
Step1 : R 유저는 h2o.importFile 함수를 호출한다.

H2O는 무엇인가? - 구조



Step2 : 함수호출을 통해서 R client는 H2O cluster와 통신한다.

H2O는 무엇인가? - 구조



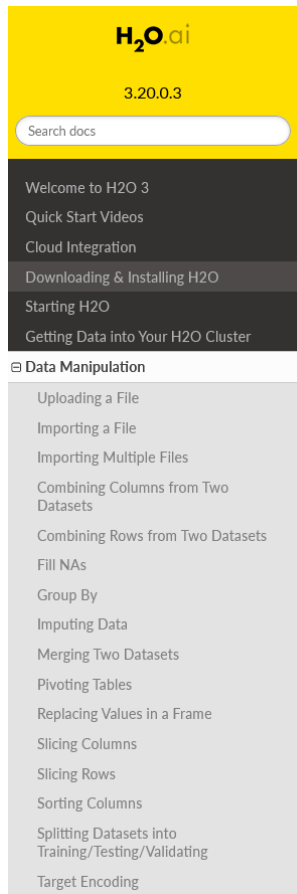
Step3 : 해당 데이터를 HDFS으로부터 가져와서 H2O Frame으로 반환한다.

H₂O는 무엇인가? - 기능

어떤 기능이 있을까요?

H2O는 무엇인가? - 기능

1) 데이터 전처리



The screenshot shows the H2O.ai website interface. At the top, there's a yellow header with the H2O.ai logo and the version number 3.20.0.3. Below the header is a search bar labeled 'Search docs'. A dark grey navigation bar contains links: 'Welcome to H2O 3', 'Quick Start Videos', 'Cloud Integration', 'Downloading & Installing H2O', 'Starting H2O', and 'Getting Data into Your H2O Cluster'. Below this, a sidebar menu is visible with a 'Data Manipulation' section expanded, showing a list of topics: 'Uploading a File', 'Importing a File', 'Importing Multiple Files', 'Combining Columns from Two Datasets', 'Combining Rows from Two Datasets', 'Fill NAs', 'Group By', 'Imputing Data', 'Merging Two Datasets', 'Pivoting Tables', 'Replacing Values in a Frame', 'Slicing Columns', 'Slicing Rows', 'Sorting Columns', 'Splitting Datasets into Training/Testing/Validating', and 'Target Encoding'.

[Docs](#) » [Data Manipulation](#)

[View page source](#)

Data Manipulation

This section provides examples of common tasks performed when preparing data for machine learning. These examples are run on a local cluster.

Note: The examples in this section include datasets that are pulled from GitHub and S3.

- [Uploading a File](#)
- [Importing a File](#)
- [Importing Multiple Files](#)
- [Combining Columns from Two Datasets](#)
- [Combining Rows from Two Datasets](#)
- [Fill NAs](#)
- [Group By](#)
- [Imputing Data](#)
- [Merging Two Datasets](#)
- [Pivoting Tables](#)
- [Replacing Values in a Frame](#)
- [Slicing Columns](#)
- [Slicing Rows](#)
- [Sorting Columns](#)
- [Splitting Datasets into Training/Testing/Validating](#)
- [Target Encoding](#)

[Previous](#)

[Next](#)

© Copyright 2016-2018 H2O.ai. Last updated on Jul 11, 2018.

Built with [Sphinx](#) using a [theme](#) provided by [Read the Docs](#).

2) 모델링

H₂O.ai

3.20.0.3

Search docs

Welcome to H2O 3
Quick Start Videos
Cloud Integration
Downloading & Installing H2O
Starting H2O
Getting Data into Your H2O Cluster
Data Manipulation

⊟ Algorithms

- Common
- Supervised
- Unsupervised
- Miscellaneous

Cross-Validation
Grid (Hyperparameter) Search
Checkpointing Models
AutoML: Automatic Machine Learning
Saving and Loading a Model
Productionizing H2O
Using Flow - H2O's Web UI
Downloading Logs
H2O Architecture
Security
FAQ
Glossary
Migrating to H2O 3

Docs » Algorithms

View page source

Algorithms

This section provides an overview of each algorithm available in H2O. For detailed information about the parameters that can be used for building models, refer to [Appendix A - Parameters](#).

Common

- [Quantiles](#)
- [Early Stopping](#)

Supervised

- [Cox Proportional Hazards \(CoxPH\)](#)
- [Deep Learning \(Neural Networks\)](#)
- [Distributed Random Forest \(DRF\)](#)
- [Generalized Linear Model \(GLM\)](#)
- [Gradient Boosting Machine \(GBM\)](#)
- [Naïve Bayes Classifier](#)
- [Stacked Ensembles](#)
- [XGBoost](#)

Unsupervised

- [Aggregator](#)
- [Generalized Low Rank Models \(GLRM\)](#)
- [K-Means Clustering](#)
- [Principal Component Analysis \(PCA\)](#)

Miscellaneous

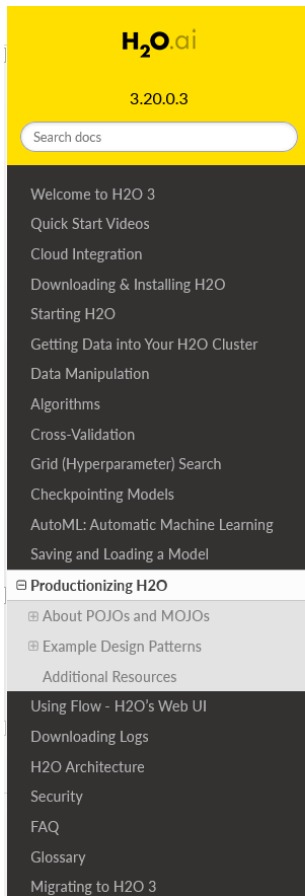
- [Word2vec](#)

Previous

Next

H2O는 무엇인가? - 기능

3) 모델 Deploy



This section describes how to build and implement a MOJO (Model Object, Optimized) to use predictive scoring. Java developers should refer to the [Javadoc](#) for more information, including packages.

What is a MOJO?

A MOJO (Model Object, Optimized) is an alternative to H2O's POJO. As with POJOs, H2O allows you to convert models that you build to MOJOs, which can then be deployed for scoring in real time.

Note: MOJOs are supported for AutoML, Deep Learning, DRF, GBM, GLM, GLRM, K-Means, Stacked Ensembles, SVM, Word2vec, and XGBoost models.

Building a MOJO

MOJOs are built in much the same way as POJOs. The example code below shows how to start H2O, build a model using either R or Python, and then compile and run the MOJO. This example uses GBM, but any supported algorithm can be used to build a model and run the MOJO.

Step 1: Start H2O, then build and extract the model

The examples below describe how to start H2O and create a model using R and Python. The `download_mojo()` function saves the model as a zip file. You can unzip the file to view the options used to build the file along with each tree built in the model. Note that each tree file is saved as a binary file type.

Build and extract a model using R:

1. Open a terminal window and start R.
2. Run the following commands to build a simple GBM model.

```
library(h2o)
h2o.init(nthreads=-1)
path <- system.file("extdata", "prostate.csv", package="h2o")
h2o_df <- h2o.importFile(path)
h2o_df$CAPSULE <- as.factor(h2o_df$CAPSULE)
model <- h2o.gbm(y="CAPSULE",
  x=c("AGE", "RACE", "PSA", "GLEASON"),
```


H₂O는 무엇인가? – 장/단점

장점 및 단점은 무엇일까요?

H2O는 무엇인가? – 장/단점

장점

- 1) 모델링이 쉽다.
- 2) R, Python 등 다른 언어와 같이 사용 할 수 있다.
- 3) 대용량 데이터를 이용한 모델링도 빠르다.
- 4) 모델배포가 어렵지 않다.

H2O는 무엇인가? – 장/단점

단점

- 1) REST API 방식으로 사용시 h2o버전이 맞지 않으면 이전 버전으로 생성된 모델 사용이 불가능하다.
- 2) 필요없는 데이터를 만들어내기도 한다.
- 3) Deep Learning의 경우 DNN의 구조이외에 다른 구조(예 : CNN, RNN 등)를 지니는 모델을 생성할 수 없다.

공지사항

- Python 등 기타 다른 언어에 대해서는 진행하지 않습니다.
- h2o를 이용한 Data Manipulation은 극히 일부분만 수행하며, 모델링에 포커스를 맞추어 진행합니다.
- 예제 데이터를 제공하되 혹시 가지고 계신 데이터가 있다면 직접 적용해서 해보시는걸 권장합니다.
- HDFS, DB 연결하는 방법은 다루지 않습니다.

실습시간!

자주 나오는 질문들 정리

- Q : R에서 H2O Frame에 NA를 포함한 row를 제거하려면 어떻게 해야하나요?
- A : `na.omit` 함수를 사용하면 됩니다.

자주 나오는 질문들 정리

- Q : Word2Vec도 있던데 사용 가능한가요?
- A : 사용 할 수 있긴 합니다. 계속 개선시키는 중입니다.

자주 나오는 질문들 정리

- Q : Deep Learning 회귀 예측 모델에 대해 deviance를 어떻게 계산하나요?
- A : Loss가 MeanSquare일 때 MSE가 deviance 입니다. 그러나 Absolute, Laplace 또는 Huber로 Loss를 셋팅하면 MSE가 deviance를 가리키지 않습니다.

자주 나오는 질문들 정리

- Q : H2O를 새로운 버전으로 업데이트 했는데 왜 이전에 학습시킨 모델을 불러 올 수 없는 건가요?
- A : H2O Binary 모델을 저장할 때, 오로지 같은 버전의 h2o에서만 작동되도록 만들었습니다. 만일 모델 배포를 위한 거라면 POJO 혹은 MOJO로 저장하여 사용하기 바랍니다.

자주 나오는 질문들 정리

```
> basic_rf_model <- h2o.loadModel("./basic_rf_180601/basic_rf_model")
```

```
ERROR: Unexpected HTTP Status code: 400 Bad Request (url = http://localhost:54321/99/Models.bin/)
```

```
java.lang.IllegalArgumentException
```

```
[1] "java.lang.IllegalArgumentException: Found version 3.18.0.8, but running version 3.20.0.2"  
[2] "    water.AutoBuffer.checkVersion(AutoBuffer.java:302)"  
[3] "    water.AutoBuffer.<init>(AutoBuffer.java:288)"  
[4] "    hex.Model.importBinaryModel(Model.java:2380)"  
[5] "    water.api.ModelsHandler.importModel(ModelsHandler.java:209)"  
[6] "    sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)"  
[7] "    sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)"  
[8] "    sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)"  
[9] "    java.lang.reflect.Method.invoke(Method.java:498)"  
[10] "    water.api.Handler.handle(Handler.java:63)"  
[36] "    org.eclipse.jetty.util.thread.QueuedThreadPool.runJob(QueuedThreadPool.java:608)"  
[37] "    org.eclipse.jetty.util.thread.QueuedThreadPool$3.run(QueuedThreadPool.java:543)"  
[38] "    java.lang.Thread.run(Thread.java:748)"
```

```
Error in .h2o.doSafeREST(h2oRestApiVersion = h2oRestApiVersion, urlSuffix = page, :
```

```
ERROR MESSAGE:
```

```
Found version 3.18.0.8, but running version 3.20.0.2
```

자주 나오는 질문들 정리

- Q : H2O 사용시 GPU도 지원하나요?
- A : H2O4GPU라는 모듈이 따로 있긴 하지만 Linux(64bit)에서만 사용이 가능합니다. 자세한 사항은 다음 링크를 참조하세요.

링크 :

<https://github.com/h2oai/h2o4gpu/blob/master/README.md>

Reference:

- <http://docs.h2o.ai/h2o/latest-stable/h2o-docs/faq.html>
- <http://docs.h2o.ai/h2o/latest-stable/h2o-docs/architecture.html>
- <https://www.slideshare.net/0xdata/rf-brighttalk>
- <https://github.com/h2oai/h2o-3>

감사합니다