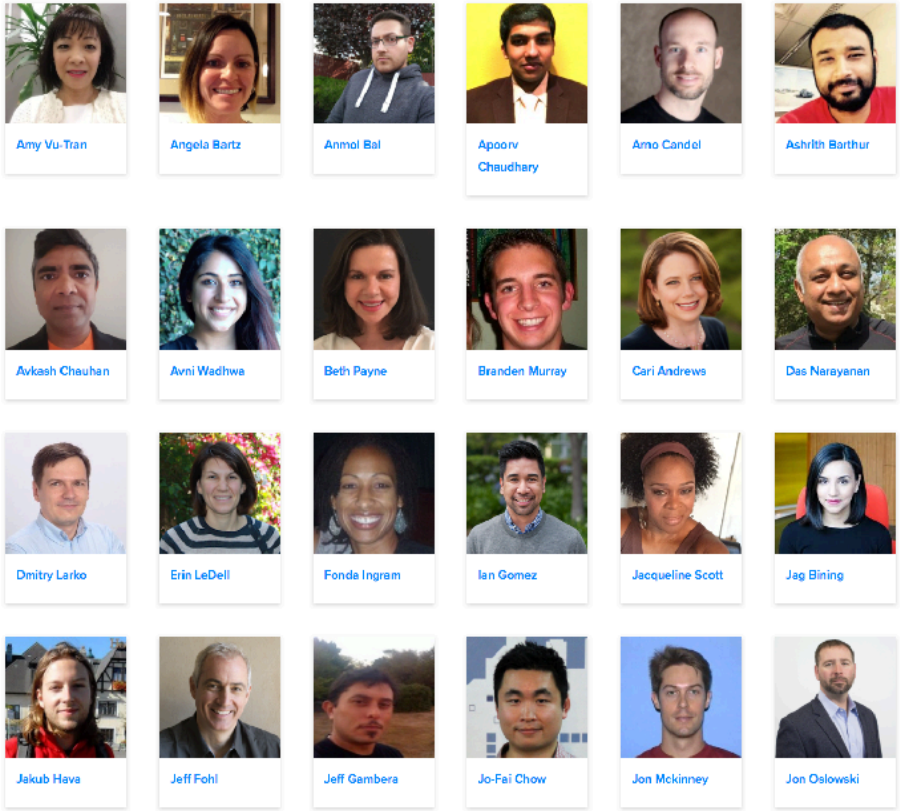

H2O
WORLD
2017

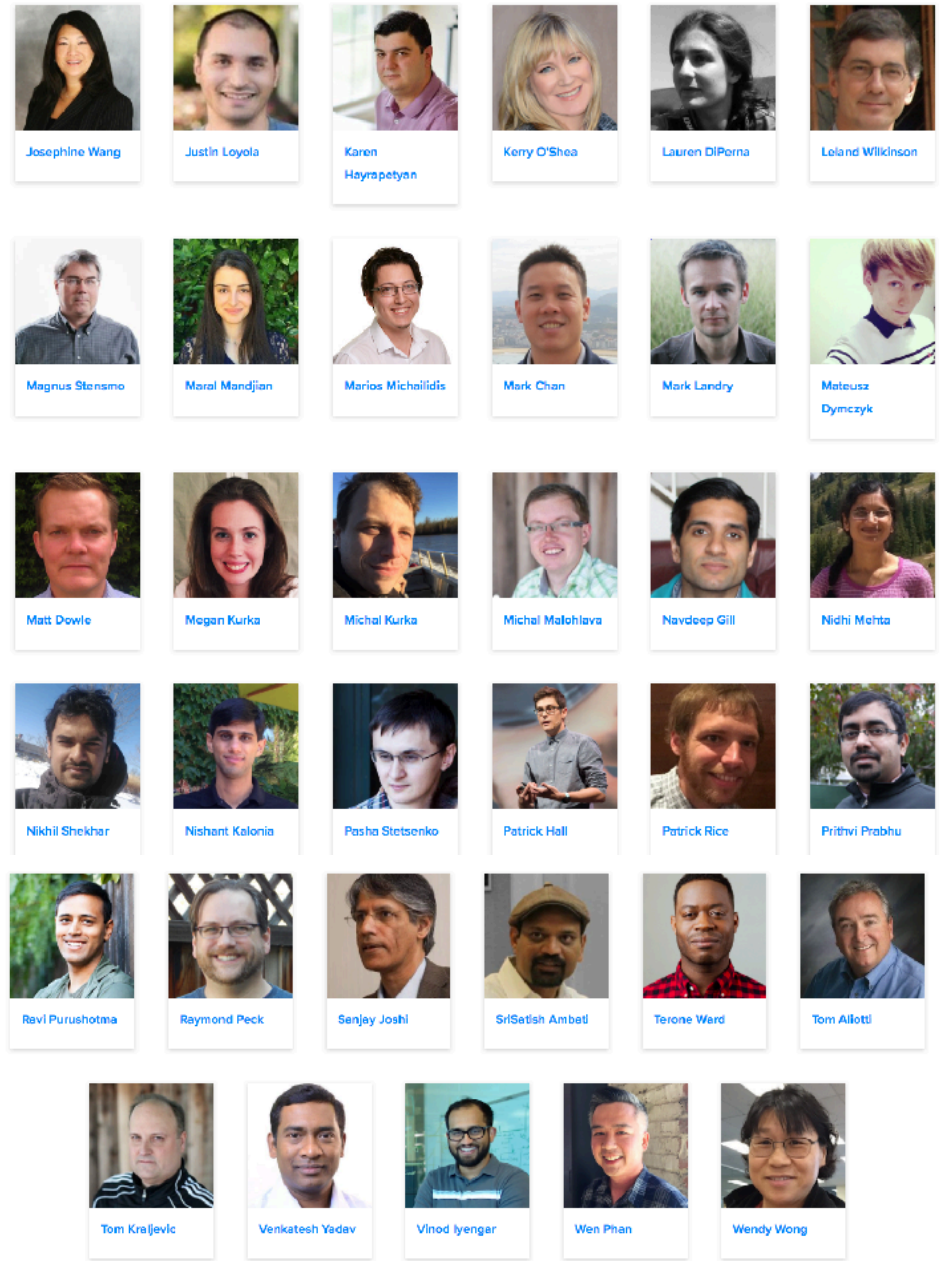
Driverless AI

Introduction and a Look under the Hood
+ Hands-On Lab

Arno Candel, CTO
@arnocandel

Team H2O!





H2O WORLD 2017

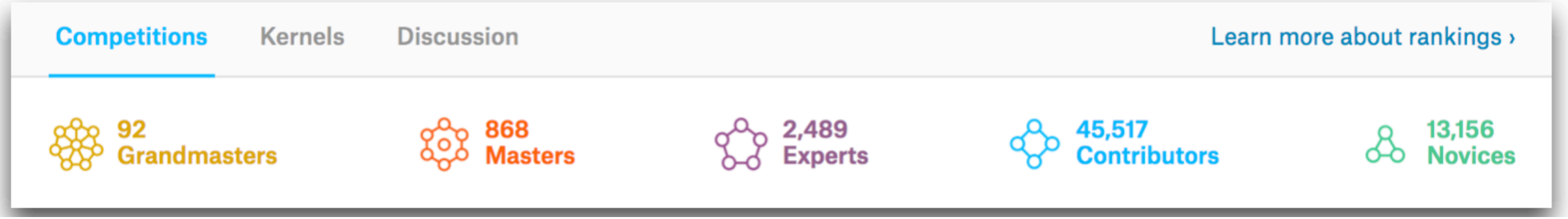
First-time Qwiklab Account Setup

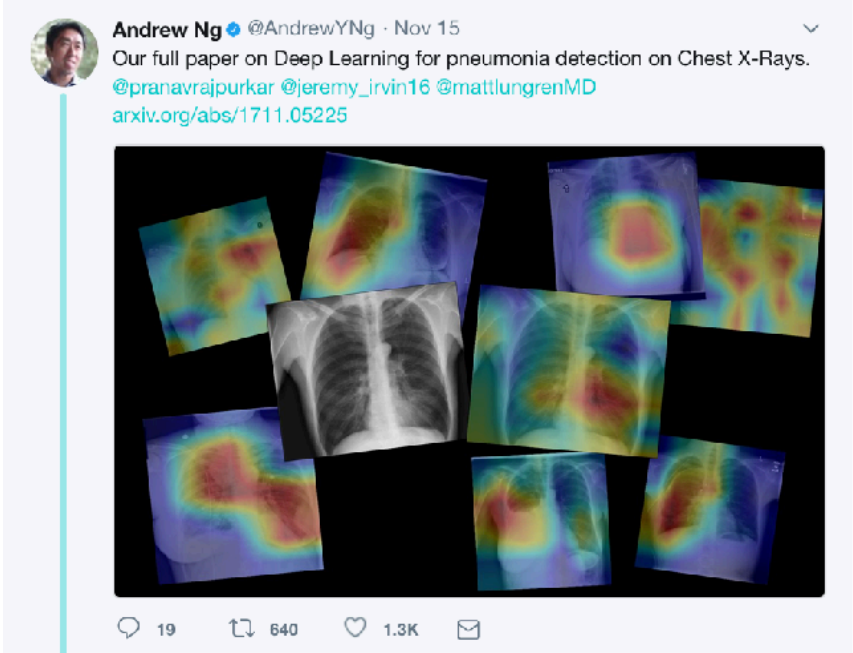
- Go to http://h2oai.qwiklab.com
- Click on “JOIN”
- Create a new account with a valid email address
- You will receive a confirmation email
  - Click on the link in the confirmation email
- Go back to http://h2oai.qwiklab.com and log in
- Go to the Catalog on the left bar
- Choose “*** FILL IN APPROPRIATELY FOR YOUR LAB ***”
- Wait for instructions

Shortage of Data Scientists

"The United States alone faces a shortage of 140,000 to 190,000 people with analytical expertise and 1.5 million managers and analysts"

–McKinsey Prediction for 2018





Automation needed to avoid human error

Submission history

From: Pranav Rajpurkar [view email]
[v1] Tue, 14 Nov 2017 17:58:50 GMT (16273kb,D)
[v2] Sat, 25 Nov 2017 04:21:27 GMT (321kb,D)

Mistake

We use the ChestX-ray14 dataset released by Wang et al. (2017) which contains 112,120 frontal-view X-ray images of 30,805 unique patients. Wang et al. (2017) annotate each image with up to 14 different thoracic pathology labels using automatic extraction methods on radiology reports. We label images that have pneumonia as one of the annotated pathologies as positive examples and label all other images as negative examples for the pneumonia detection task. We randomly split the entire dataset into 80% training, and 20% validation.

CheXNet (ours)	CheXNet (ours)
0.8209	0.8094
0.9048	0.9248
0.8831	0.8638
0.7204	0.7345
0.8618	0.8676
0.7766	0.7802
0.7632	0.7680
0.8932	0.8887
0.7939	0.7901
0.8932	0.8878
0.9260	0.9371
0.8044	0.8047
0.8138	0.8062
0.9387	0.9164

Correction

We use the ChestX-ray14 dataset released by Wang et al. (2017) which contains 112,120 frontal-view X-ray images of 30,805 unique patients. Wang et al. (2017) annotate each image with up to 14 different thoracic pathology labels using automatic extraction methods on radiology reports. We label images that have pneumonia as one of the annotated pathologies as positive examples and label all other images as negative examples. For the pneumonia detection task, we randomly split the dataset into training (28744 patients, 98637 images), validation (1672 patients, 6351 images), and test (389 patients, 420 images). There is no patient overlap between the sets.

InfoWorld
FROM IDG

Home > Artificial Intelligence > Machine Learning

INSIDER

Review: H2O.ai automates machine learning

Driverless AI really is able to create and train good machine learning models without requiring machine learning expertise from users

By Martin Heller

Contributing Editor, InfoWorld | NOV 6, 2017

AT A GLANCE

H2O.ai Driverless AI 1.0.5

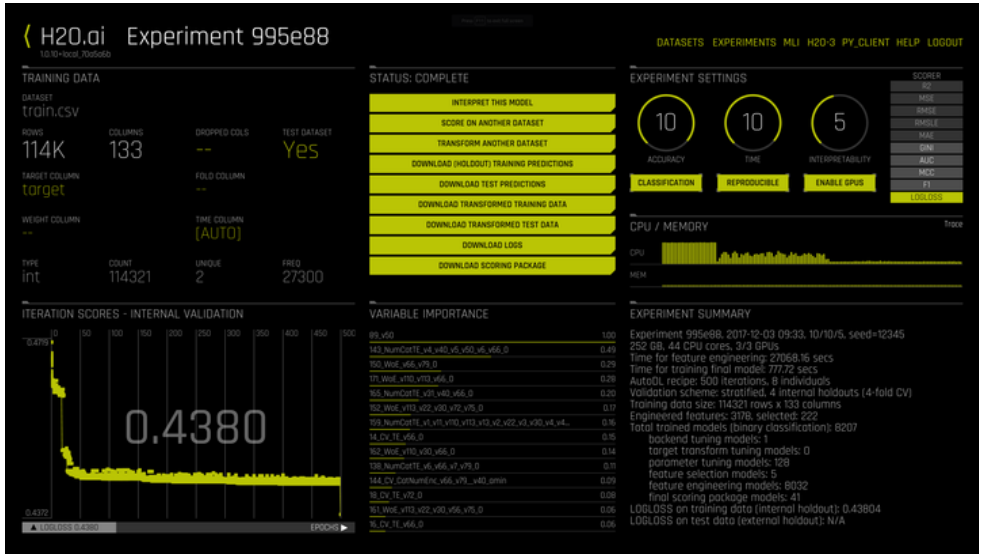
★★★★★

LEARN MORE

on H2O.ai

H2O WORLD 2017

Driverless AI: top 1% in BNP Paribas Kaggle competition



single run, <u>fully automated:</u> 6h on 3 GPUs

BNP Paribas Cardif Claims Management

Can you accelerate BNP Paribas Cardif's claims management process?

$30,000 · 2,926 teams · 2 years ago

Submission and Description	Private Score
test_preds.csv a few seconds ago by Arno Candel Driverless AI 1.0.10 10/10/5 on 3 GPUs	0.43316

In the money · Gold · Silver · Bronze

#	△pub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	Dexter's Lab			0.42037	198	2y
2	—	escalated chi			0.42079	162	2y
3	—	Exploding Kittens			0.42182	124	2y
4	—	Branden | Nickel | utility			0.42259	251	2y
5	—	the flying burrito brothers			0.42450	264	2y
6	—	n_m			0.42535	4	2y
7	—	PAFY			0.42557	310	2y
8	—	KAME			0.42688	121	2y
9	—	Jack (Japan)			0.42744	22	2y
10	▲1	Dmitry & Bohdan			0.43000	192	2y
11	▲1	Li-Der			0.43006	56	2y
12	▼2	BK3M2PRS			0.43089	338	2y
13	—	x2x4x8			0.43107	55	2y
14	—	Frenchies			0.43146	134	2y
15	▲1	AIns			0.43168	55	2y
17	—	BLR-2			0.43313	129	2y
18	▲3	no one			0.43317	88	2y

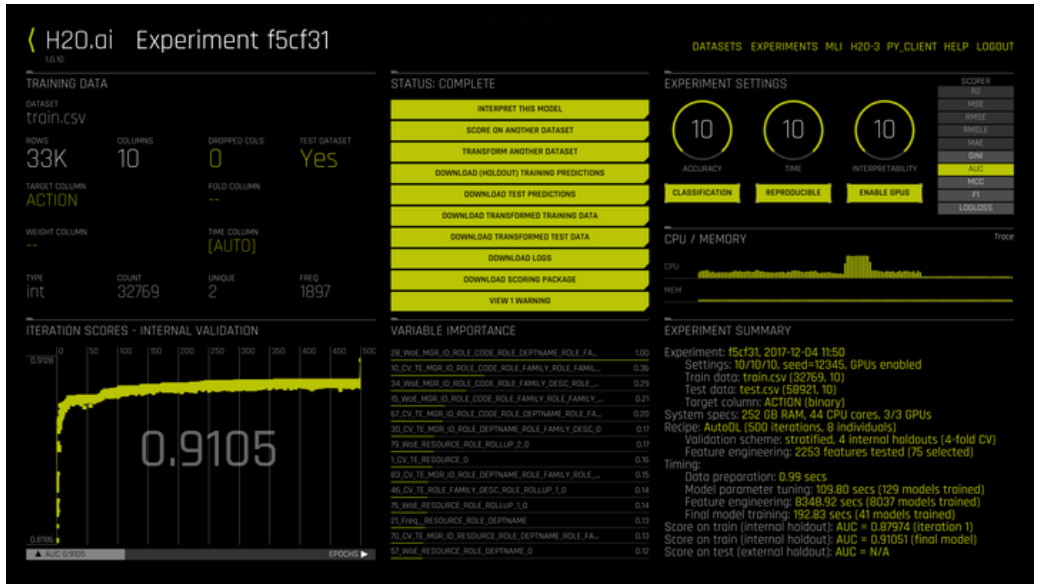
Driverless AI: 18th place in private LB (out of 2926)

Hours for Driverless AI — Weeks for grandmasters

H2O WORLD 2017

Driverless AI: top 5% in Amazon Kaggle competition

Driverless AI produces feature engineering pipeline ("more columns") for downstream use



Amazon.com - Employee Access Challenge

Predict an employee's access needs, given his/her job role

$5,000 · 1,687 teams · 4 years ago

Driverless AI: 80th place in private LB (out of 1687 - top 5%)

Empowering DAI (10) – Plus OHE

DAI derived data

Models built on DAI FE	Test LB
Lightgbm with gbdt	0.909
Lightgbm with dart	0.909
Extra Trees	0.910
Random Forest	0.907
Logistic Regression	0.898
Lightgbm rmse	0.906
Lightgbm Huber	0.900
Xgboost	0.908
DAI	0.909
Lightgbm plus counts	0.913
Logistic plus dummies	0.907

Logistic model does not perform as good. Because best features were found using tree methods

Stacking

From **0.914**
To **0.9158**

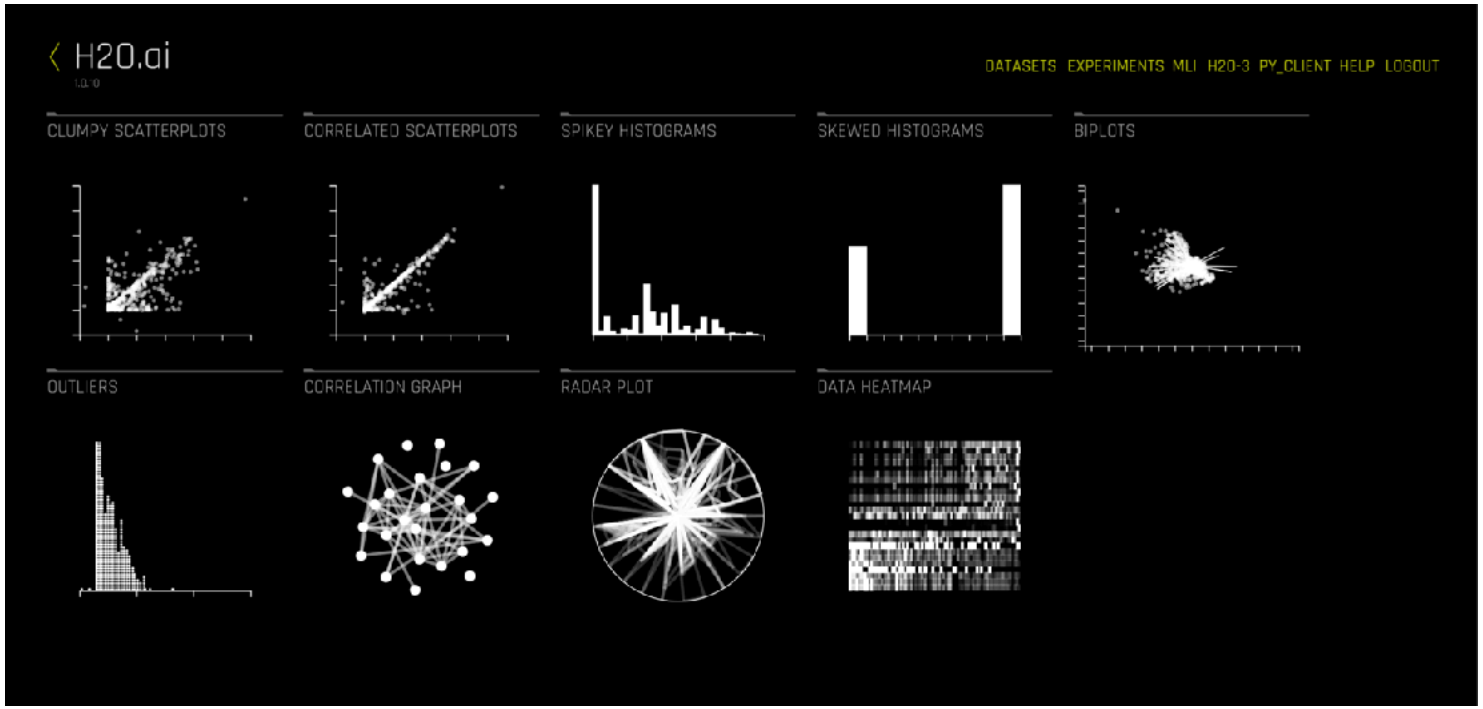
Dummy Variables or One-Hot Encoding can improve results for linear models.

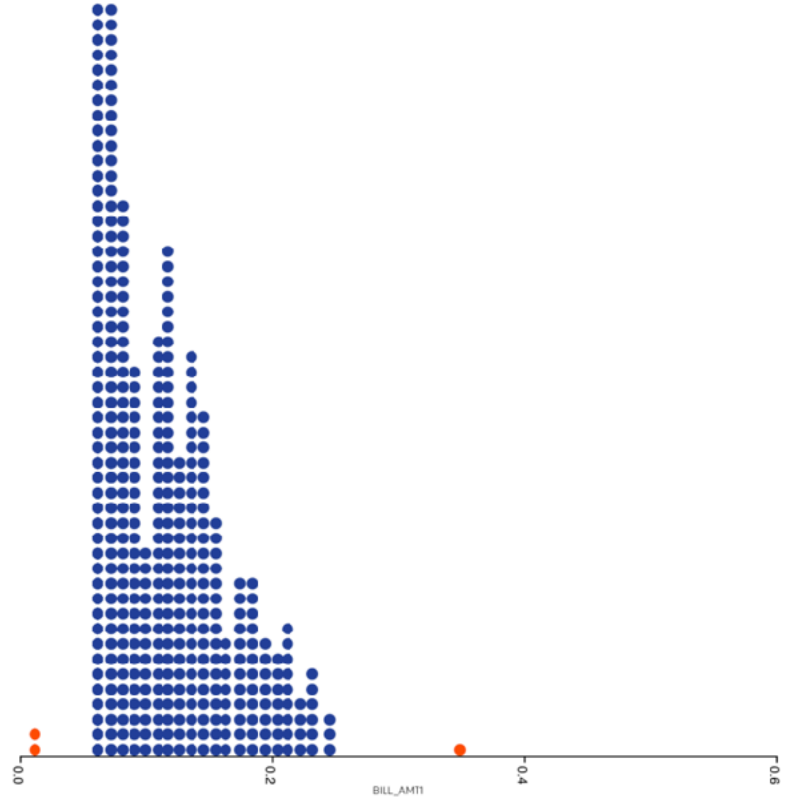
With a little bit of stacking: 20th place (top 1.5%)

https://www.youtube.com/watch?v=qtUNyJIAID0&t=11s
https://github.com/kaz-Anova/Competitive_Dai

H2O WORLD 2017

Automatic Visualization





Contains novel statistical algorithms to only show "relevant" aspects of the data
(soon: automated data cleaning)

Scalable outlier detection
(no sampling)

H2O WORLD 2017

Machine Learning Interpretation

Gain confidence in models before deploying them!

H2O WORLD 2017

MOJO: Pure Java Production Deployment

```
mojo_version = 2.0
uuid = 'ccf86100-6e7e-43ed-aaf2-1f88e081404f'
time_created = 2017-11-22T17:44:59.268021

[columns]
features = [
  ['sepal_len', 'float64'],
  ['sepal_wid', 'float64'],
  ['petal_len', 'float64'],
  ['petal_wid', 'float64'],
]
outputs = [
  ['0_NumToCatWoE_petal_len_0', 'float64'],
  ...
  ['25_NumToCatTE_petal_len_petal_wid_sepal_len_sepal_wid_2', 'float64'],
]

[[transforms]]
  name = 'IntervalMap'
  inputs = ['petal_len']
  outputs = ['petal_len.1']
  breakpoints = [1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.9, 3.3, 3.5, 3.8, 3.9,
                 4.0, 4.1, 4.2, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0, 5.1,
                 5.2, 5.4, 5.5, 5.6]
  values = [[1.0], [2.0], [3.0], ..., [-1.0], [NaN]]

[[transforms]]
  name = 'FillNa'
  inputs = ['petal_len.1']
  outputs = ['petal_len.2']
  repl = -1.0

[[transforms]]
  name = 'Map'
  inputs = ['petal_len.2']
  outputs = ['0_NumToCatWoE_petal_len_0', '0_NumToCatWoE_petal_len_1', '0_NumToCatWoE_petal_len_2']
  map = [
    [[1.0], [2.8903717578961645, -1.504077396776274, -1.504077396776274]],
    [[2.0], [3.4011973816621555, -2.0149030205422647, -2.0149030205422647]],
    ...
    [[27.0], [-2.9704144655697013, -2.9704144655697013, 4.356708826689592]],
  ]
  missing = [-0.68818439121781616, -0.68818439121781616, -0.68818439121781616]

...
```

- feature engineering and model scoring logic
- auto-generated human-readable representation
- minimal platform-independent storage format
- scoring backend can be in any language (C/Java/C#/Go/etc.)

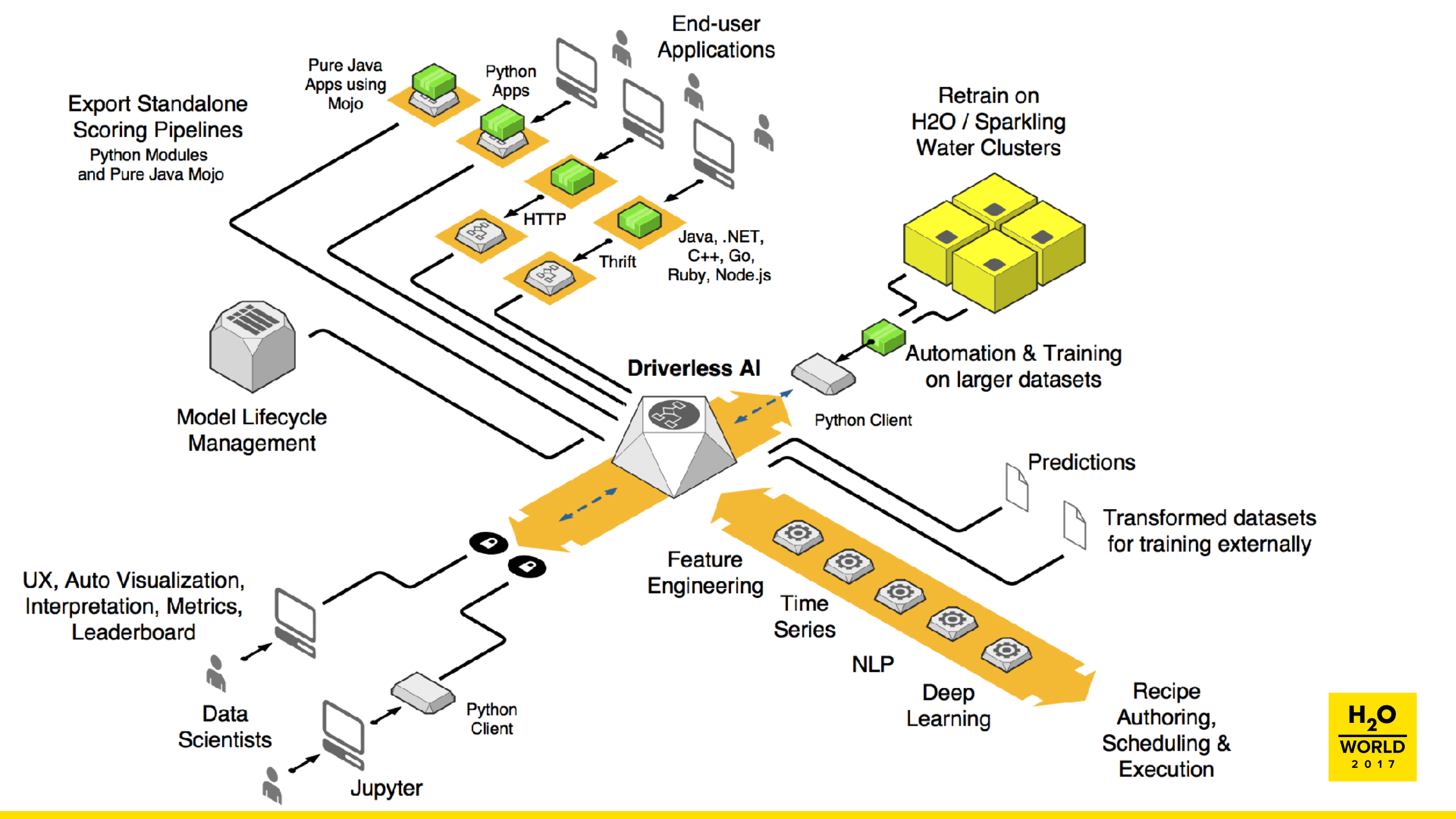
```
@Test
void testFillNaTransform() throws IOException {
  SimpleMojoBackend backend = new SimpleMojoBackend((new SB())
      .p("mojo_version = 2.0").nl()
      .p("[columns]").nl()
      .p("  features = [['len', 'int32']]").nl()
      .p("  outputs = [['len_', 'int32']]").nl()
      .nl()
      .p("[[transforms]]").nl()
      .p("  name = 'FillNa'").nl()
      .p("  inputs = ['len']").nl()
      .p("  outputs = ['len_']").nl()
      .p("  repl = -1").nl()
      .toString()
  );

  MojoModel2 model = MojoModel2.loadFrom(backend);
  MojoFrame iframe = model.getInputFrame();
  MojoFrame oframe = model.getOutputFrame();
  assertEquals(iframe.getNcols(),  actual: 1);
  assertEquals(oframe.getNcols(),  actual: 1);

  iframe.fillFromCsvData(
      ars("len"),
      arss(ars("724"), ars("12"), ars("-3"), ars(""), ars("0"))
  );
  model.transform();
  int[] out = (int[]) oframe.getColumnData( index: 0);

  assertEquals(iframe.getNrows(),  actual: 5);
  assertEquals(oframe.getNrows(),  actual: 5);
  assertArrayEquals(out, ari(724, 12, -3, -1, 0));
}
```

H2O WORLD 2017


End-user Applications
Pure Java Apps using Mojo
Python Apps
Export Standalone Scoring Pipelines
Python Modules and Pure Java Mojo
HTTP
Thrift
Java, .NET, C++, Go, Ruby, Node.js
Retrain on H2O / Sparkling Water Clusters
Automation & Training on larger datasets
Driverless AI
Python Client
Model Lifecycle Management
Predictions
Transformed datasets for training externally
UX, Auto Visualization, Interpretation, Metrics, Leaderboard
Feature Engineering
Time Series
NLP
Deep Learning
Recipe Authoring, Scheduling & Execution
Data Scientists
Python Client
Jupyter
H2O WORLD 2017

Driverless AI Roadmap

Feature	Now	Q1 2018	Q2 2018	Q3 2018
AutoDL Feature Engineering Recipe	■	■	■	■
Supervised Structured Data, CSV, Text	■	■	■	■
Overfitting and Leakage Prevention	■	■	■	■
Machine Learning Interpretation	■	■	■	■
Automatic Visualization	■	■	■	■
GUI	■	■	■	■
Python client API	■	■	■	■
Python scoring API HTTP Thrift Scoring API	■	■	■	■
Multi-GPU (shared data)	■	■	■	■
Scoring MOJO (100% Java or C)		■	■	■
Data connectors: HDFS, SQL		■	■	■
User Management: LDAP, Kerberos		■	■	■
TensorFlow Deep Learning NLP Recipes		■	■	■
Time Series Recipes		■	■	■
Multi-GPU (sharded data) - optimized for DGX Volta		■	■	■
UDR (User-Defined Recipes), Verticals			■	■
Multi-Node Multi-GPU - optimized for DGX Volta				■
Sparkling Water Backend for Driverless AI				■

H2O WORLD 2017

docs.h2o.ai

H2O Driverless AI

Thank you very much for your interest with our new and exciting Driverless AI product. This product leverages GPU Machines with a focus on Auto Feature Engineering, Model Interpretability, and Automatic Data Visualization.

DOWNLOAD DRIVERLESS AI DRIVERLESS AI DOCUMENTATION

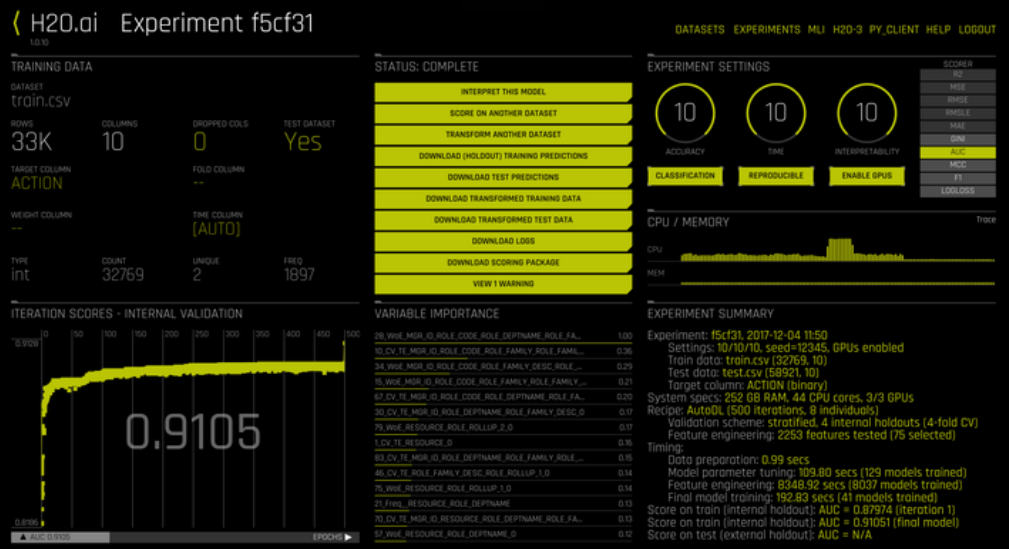
Don't have a registration key? Apply here to try Driverless AI.

Here are some other resources to help you get started:

Using Driverless AI Booklet
Machine Learning Interpretability with H2O Driverless AI Booklet
Driverless AI Data Sheet
Webinars recently delivered on Driverless AI

H2O WORLD 2017

Hands-on Lab



H2O WORLD 2017