Company Overview

Company
• Team: 35. Founded in 2012, Mountain View, CA
• Stanford Math & Systems Engineers

Product
• Open Source Leader in Machine & Deep learning
• Ease of Use and Smarter Applications
• R, Python, Spark & Hadoop Interfaces
• Expanding Predictions to Mass Analyst markets
Executive Team

Sri Satish Ambati
CEO & Co-founder
DataStax

Cliff Click
CTO & Co-founder
Sun, Java Hotspot

Tom Kraljevic
VP of Engineering
Abrizio, Intel

Arno Candel
Chief Architect
Physicist, Deep Learning

Board of Directors
Jishnu Bhattacharjee // Nexus Ventures
Ash Bhardwaj // Flextronics

Scientific Advisory Council
Trevor Hastie
Stephen Boyd
Rob Tibshirani
Scientific Advisory Council

**Dr. Trevor Hastie**
- PhD in Statistics, Stanford University
- John A. Overdeck Professor of Mathematics, Stanford University
- Co-author, *The Elements of Statistical Learning: Prediction, Inference and Data Mining*
- Co-author with John Chambers, *Statistical Models in S*
- Co-author, *Generalized Additive Models*
- 108,404 citations (via Google Scholar)

**Dr. Rob Tibshirani**
- PhD in Statistics, Stanford University
- Professor of Statistics and Health Research and Policy, Stanford University
- COPPS Presidents’ Award recipient
- Co-author, *The Elements of Statistical Learning: Prediction, Inference and Data Mining*
- Author, *Regression Shrinkage and Selection via the Lasso*
- Co-author, *An Introduction to the Bootstrap*

**Dr. Stephen Boyd**
- PhD in Electrical Engineering and Computer Science, UC Berkeley
- Professor of Electrical Engineering and Computer Science, Stanford University
- Co-author, *Convex Optimization*
- Co-author, *Linear Matrix Inequalities in System and Control Theory*
- Co-author, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*
Accuracy with Speed and Scale

H₂O

Flow

Nano Fast Scoring Engine

Parallel Distributed Processing

In-Memory Columnar Compression

Deep Learning

Features
- Outliers
- Cluster
- Classify
- Regression
- Boosting
- Forests
- Solvers

Ensembles

Spark

Hadoop

HDFS

S3

SQL

NoSQL
Accuracy with Speed and Scale

Fast Modeling Engine

- In-Memory
- Map Reduce/Fork Join
- Columnar Compression
- Matrix Factorization
- Feature Engineering
- Munging
- Clustering
- Classification Regression
- Deep Learning
- PCA, GLM, Cox
- Random Forest / GBM Ensembles

Streaming

Nano Fast Java Scoring Engines
Algorithms on H₂O

**Supervised Learning**

- Generalized Linear Models: Binomial, Gaussian, Gamma, Poisson and Tweedie
- Cox Proportional Hazards Models
- Naïve Bayes
- Distributed Random Forest: Classification or regression models
- Gradient Boosting Machine: Produces an ensemble of decision trees with increasing refined approximations
- Deep learning: Create multi-layer feed forward neural networks starting with an input layer followed by multiple layers of nonlinear transformations
Algorithms on H₂O

**Unsupervised Learning**

- **Clustering**
  - K-means: Partitions observations into $k$ clusters/groups of the same spatial size

- **Dimensionality Reduction**
  - Principal Component Analysis: Linearly transforms correlated variables to independent components

- **Anomaly Detection**
  - Autoencoders: Find outliers using a nonlinear dimensionality reduction using deep learning
Reading Data from HDFS into H2O with R

**STEP 1**

R user

```r
h2o_df = h2o.importFile("hdfs://path/to/data.csv")
```
Reading Data from HDFS into H2O with R

**STEP 2**

1. **2.1** R function call
   - `h2o.importFile()`

2. **2.2** HTTP REST API request to H2O has HDFS path

3. **2.3** Initiate distributed ingest

4. **2.4** Request data from HDFS

- H2O Cluster
- HDFS
- `data.csv`
Reading Data from HDFS into H2O with R

STEP 3

R

h2o_df

Cluster IP
Cluster Port
Pointer to Data

3.3
Distributed H2O Frame in DKV

3.2

Cluster IP
Cluster Port
Pointer to Data

Return pointer to data in REST API
JSON Response

3.4

h2o_df object created in R

3.1

HDFS provides data

data.csv

H2O Cluster

H2O Frame

H2O Frame in DKV

H2O
R Script Starting H2O GLM

- **R Script**
- **Standard R process**

- **H2O process**
  - **H2O - core**
  - **H2O - algos**
  - **User process**

**TCP/IP**

**HTTP**

- **REST/JSON**
  - `.h2o.startModelJob()`
  - POST `/3/ModelBuilders/glm`  
  - `h2o.glm()`  

**Legend**

- Network layer
- REST layer
- H2O - core
- H2O - algos
- User process
R Script Retrieving H2O GLM Result

```
H2O
```

```
Machine Intelligence
```

```
H2O
```

```
Legend
```

```
Network layer
```

```
REST layer
```

```
H2O - algos
```

```
H2O - core
```

```
User process
```

```
H2O process
```

```
HTTP
```

```
TCP/IP
```

```
REST/JSON
```

```
h2o.getModel()
GET /3/Models/glm_model_id
```

```
h2o.glm()
```

```
R script
```

```
Standard R process
```

```
Fork/Join framework
K/V store framework
```

```
H2O process
```

```
H2O.ai Machine Intelligence
```
H2O Billion Row Machine Learning Benchmark
GLM Logistic Regression

Hadoop/Mahout

H2O 16 EC2 nodes

H2O 16 EC2 nodes

H2O 48 EC2 nodes

H2O 48 EC2 nodes

34.9 sec, 3 iterations
numerical and categorical

16.5 sec, 2 iterations
numerical

14.2 sec, 3 iterations
numerical and categorical

5.6 sec, 2 iterations
numerical

Compute Hardware: AWS EC2 c3.2xlarge - 8 cores and 15 GB per node, 1 GbE interconnect
Airline Dataset 1987-2013, 42 GB CSV, 1 billion rows, 12 input columns, 1 outcome column
9 numerical features, 3 categorical features with cardinalities 30, 376 and 380
Demo Time!

Citi Bike
Community

R

Python

Hadoop

Spark

DataRobot