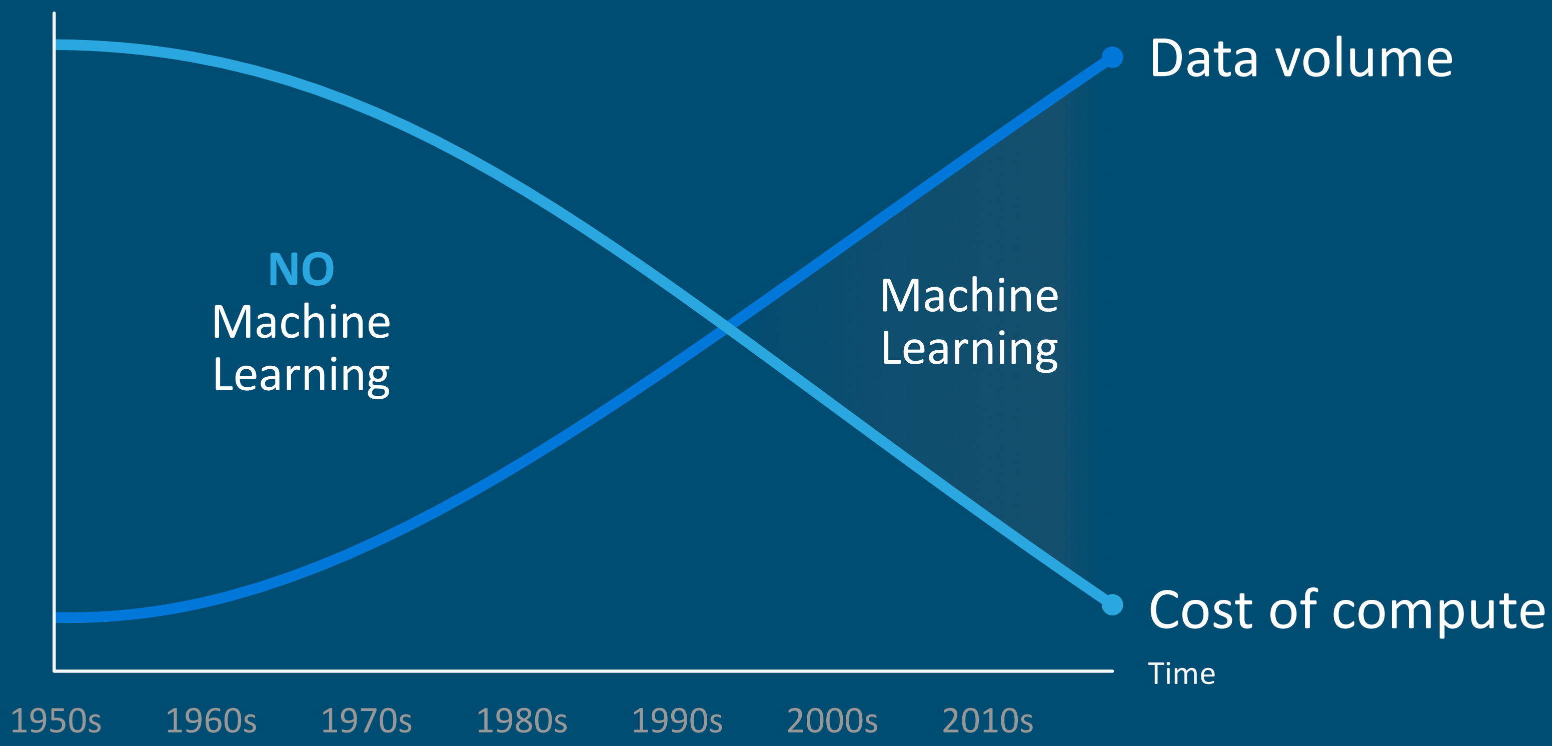
cloudera

Data Science on Hadoop

Justin Erickson
Senior Director, Product Management

Age of Machine Learning

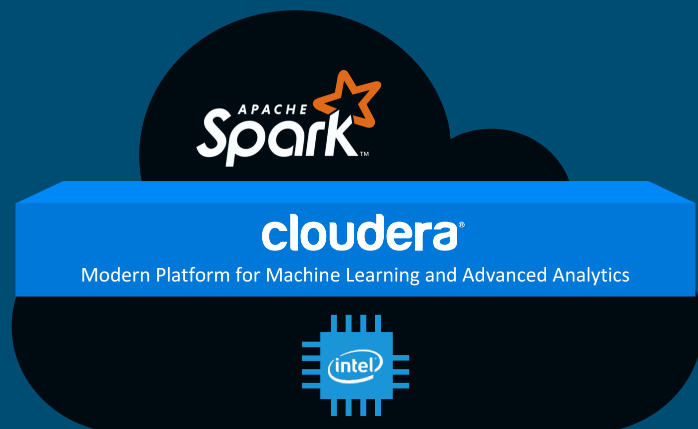


The Enterprise Platform for Data Science and Machine Learning

440x
MORE DATA

30B
CONNECTED DEVICES

The data is now here



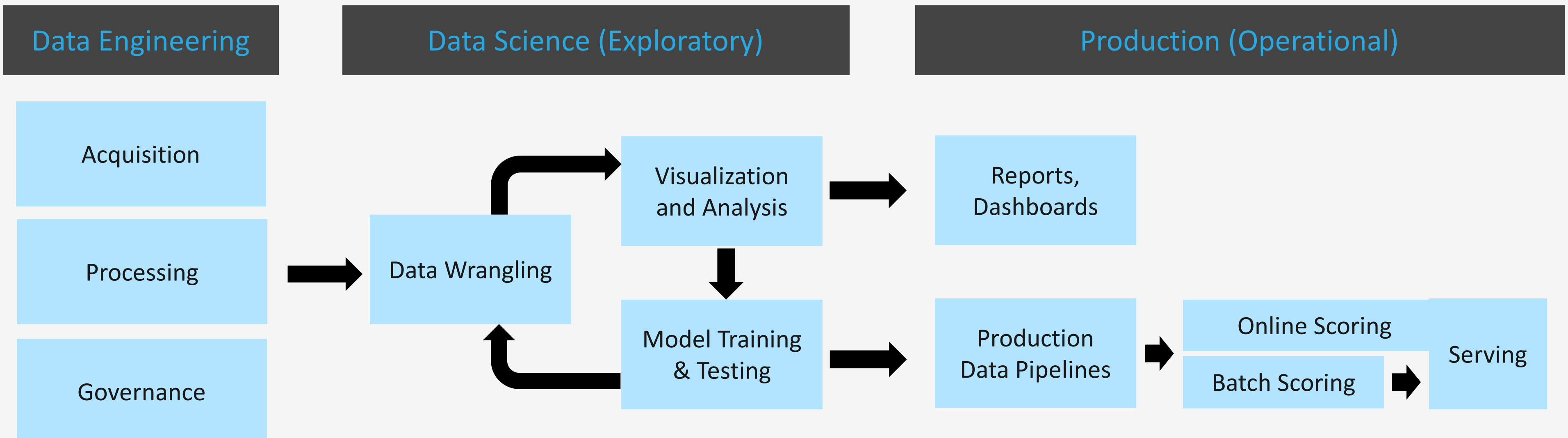
Cloudera first to integrate Spark

500
Customers
Run Spark on
cloudera

Leading adoption among enterprises

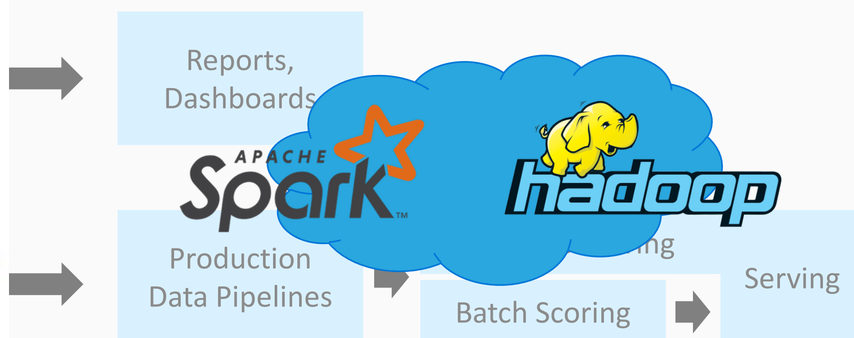
Sample data science / machine learning workflow

From data to exploration to action



The good news

Data Engineering	Data Science (Exploratory)	Production (Operational)



Data has never been more plentiful

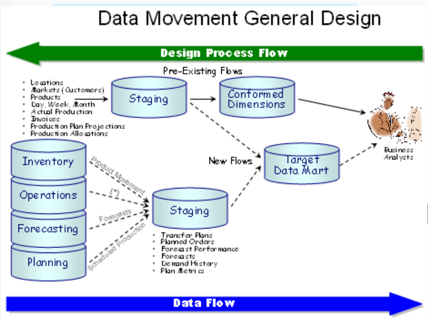
Open source data science and machine learning libraries are rapidly evolving

Commodity (and on-demand) compute makes scalable production machine learning affordable

cloudera

The bad news

Data Engineering	Data Science (Exploratory)	Production (Operational)



Governance

& Testing

Data Pipelines

Batch Scoring

Serving

Data needs to move across multiple different systems

Teams have different, conflicting requests for languages & libraries

Most data science done at small scale, individually, and is difficult to replicate

Very few models reach production

cloudera

Additional challenges

Access

For sensitive data, secure clusters are difficult to access. And IT typically doesn't want random packages installed on a secure cluster.

Popular open source tools don't easily connect to these environments, or always support Hadoop data formats.

Scale

Laptops rarely have capacity for medium, let alone big data. This leads to a lot of sampling.

Popular frameworks don't easily parallelize on a cluster. Typically code has to get rewritten for production.

Developer Experience

Notebooks, while awesome, don't easily support virtual environment and dependency management, especially for teams. This makes sharing and reproducibility hard.

Notebooks are also challenging to "put into production."

This year, our goal is to enable data science and machine learning at scale.

cloudera

Open data science in the enterprise

Data Scientist
explore, experiment, iterate

IT
drive adoption while maintaining compliance

cloudera

Our goal: An open platform for data science at scale

Help more data scientists use the power of Hadoop

Use a powerful, familiar environment with direct access to Hadoop data and compute

Data Scientist
Data Engineer

Make it easy and secure to add new users, use cases

Offer secure self-service analytics and a faster path to production on common, affordable infrastructure

Enterprise Architect
Hadoop Admin

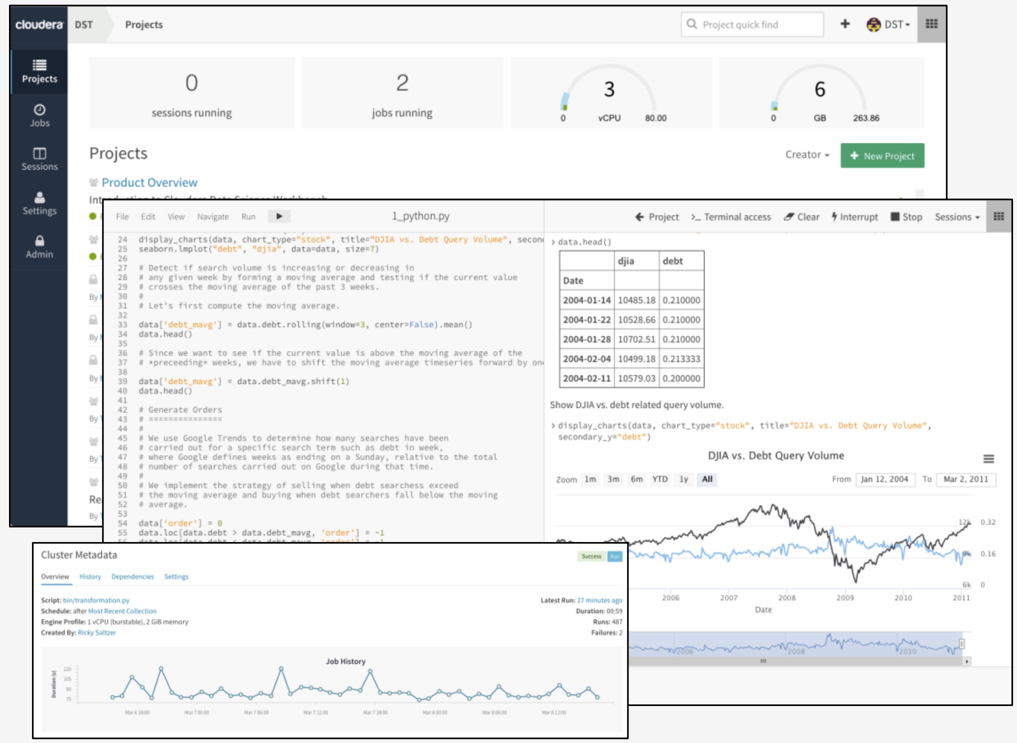
cloudera

Introducing Cloudera Data Science Workbench

Self-service data science for the enterprise

Accelerates data science from development to production with:

- Secure self-service environments for data scientists to work against Cloudera clusters
- Support for Python, R, and Scala, plus project dependency isolation for multiple library versions
- Workflow automation, version control, collaboration and sharing



Demo

With Cloudera Data Science Workbench…

Data scientists can:

- Use R, Python, or Scala from a web browser, with no desktop footprint
- Install any library or framework within isolated project environments
- Directly access data in secure clusters with Spark and Impala
- Share insights with their team for reproducible, collaborative research
- Automate and monitor data pipelines using built-in job scheduling

IT can:

- Give their data science team the freedom to work how they want, when they want
- Stay compliant with out-of-the-box support for full platform security, especially Kerberos
- Run on-premises or in the cloud, wherever data is managed

cloudera

Solving Data Science is a Full-Stack Problem

- Support unlimited data
- Provide sufficient tools for Analysts
- Provide sufficient tools for Data Scientists + Data Engineers
- Enable real-time use cases
- Provide data governance
- Provide full-stack security
- Deploy in the cloud
- Integrate with partner tools
- Be easy for IT to deploy/maintain

✓ Hadoop
✓ Impala, Hive, Hue
✓ Spark, Data Science Workbench
✓ Kafka, Spark Streaming
✓ Navigator + Partners
✓ Kerberos, Sentry, Record Service, KMS/KTS
✓ Cloudera Director
✓ Rich Ecosystem
✓ Cloudera Manager + Director

cloudera

The importance of an open ecosystem

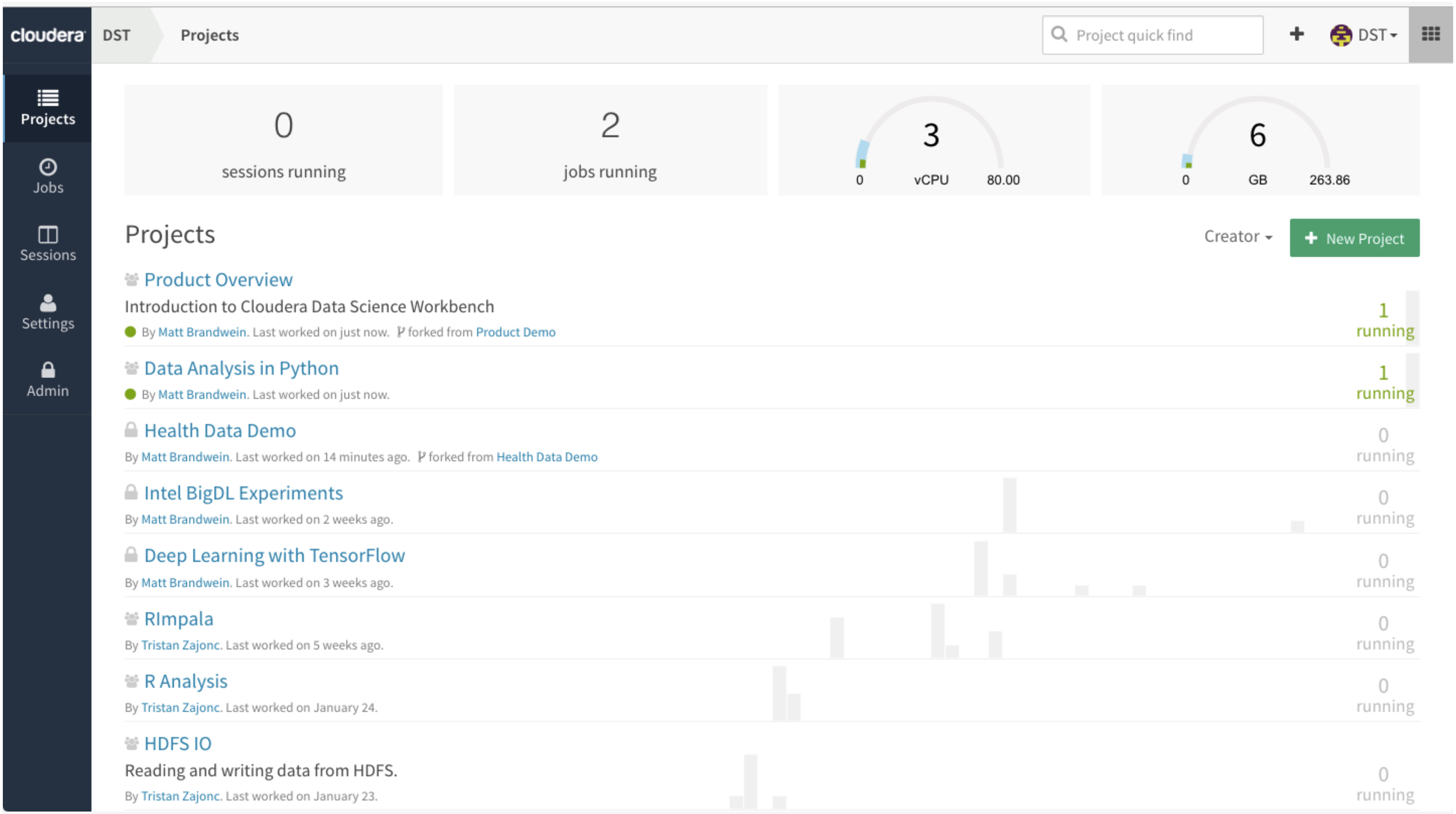
Open Ecosystem

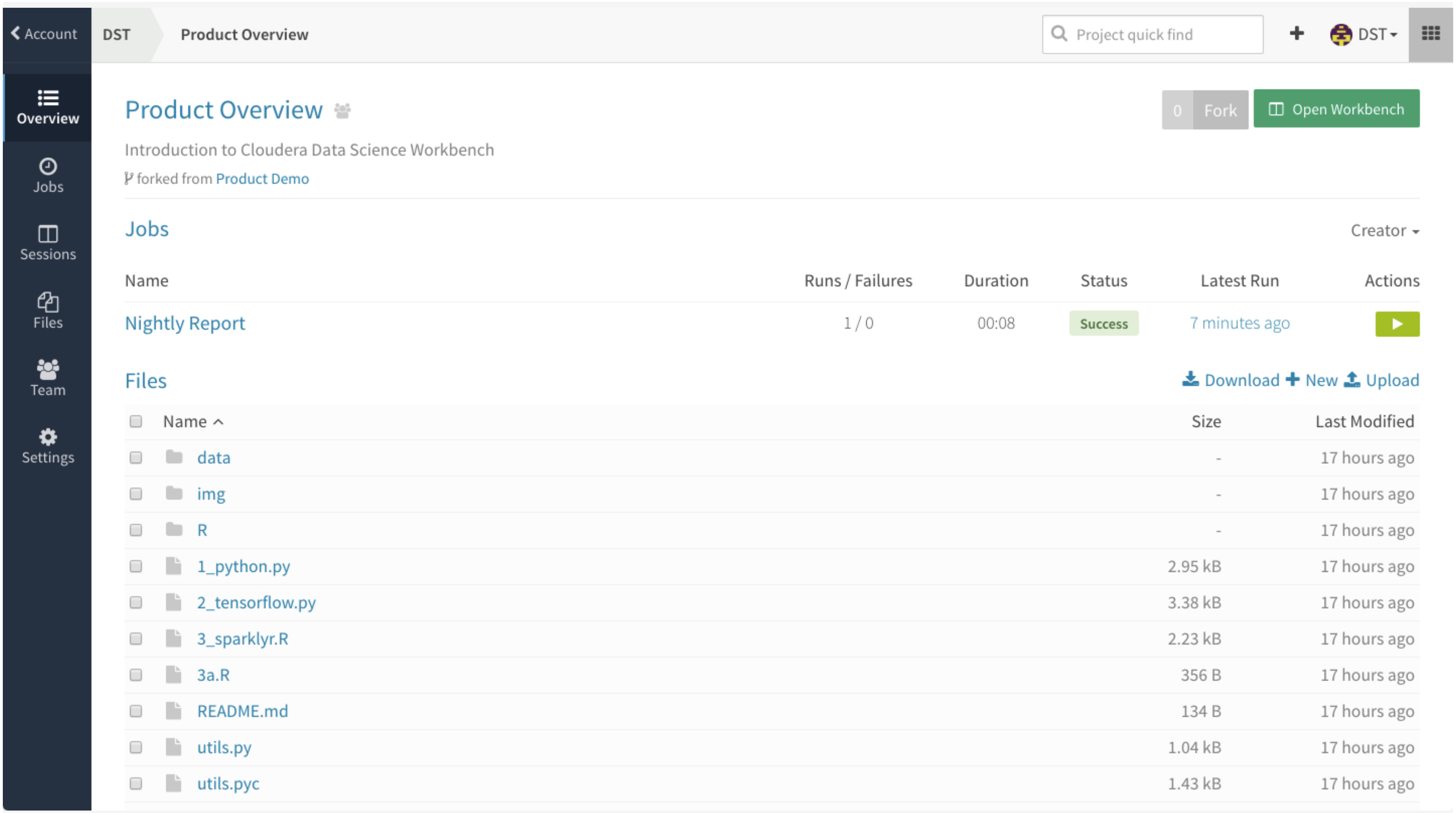
jupyter
Apache Zeppelin
RStudio
ANACONDA
TensorFlow
H_2O
DataRobot
skymind
dataiku
KNIME
rapidminer
SKYTREE
THE MACHINE LEARNING COMPANY
Digital Reasoning
Microsoft
REVOLUTION ANALYTICS
sas
intel

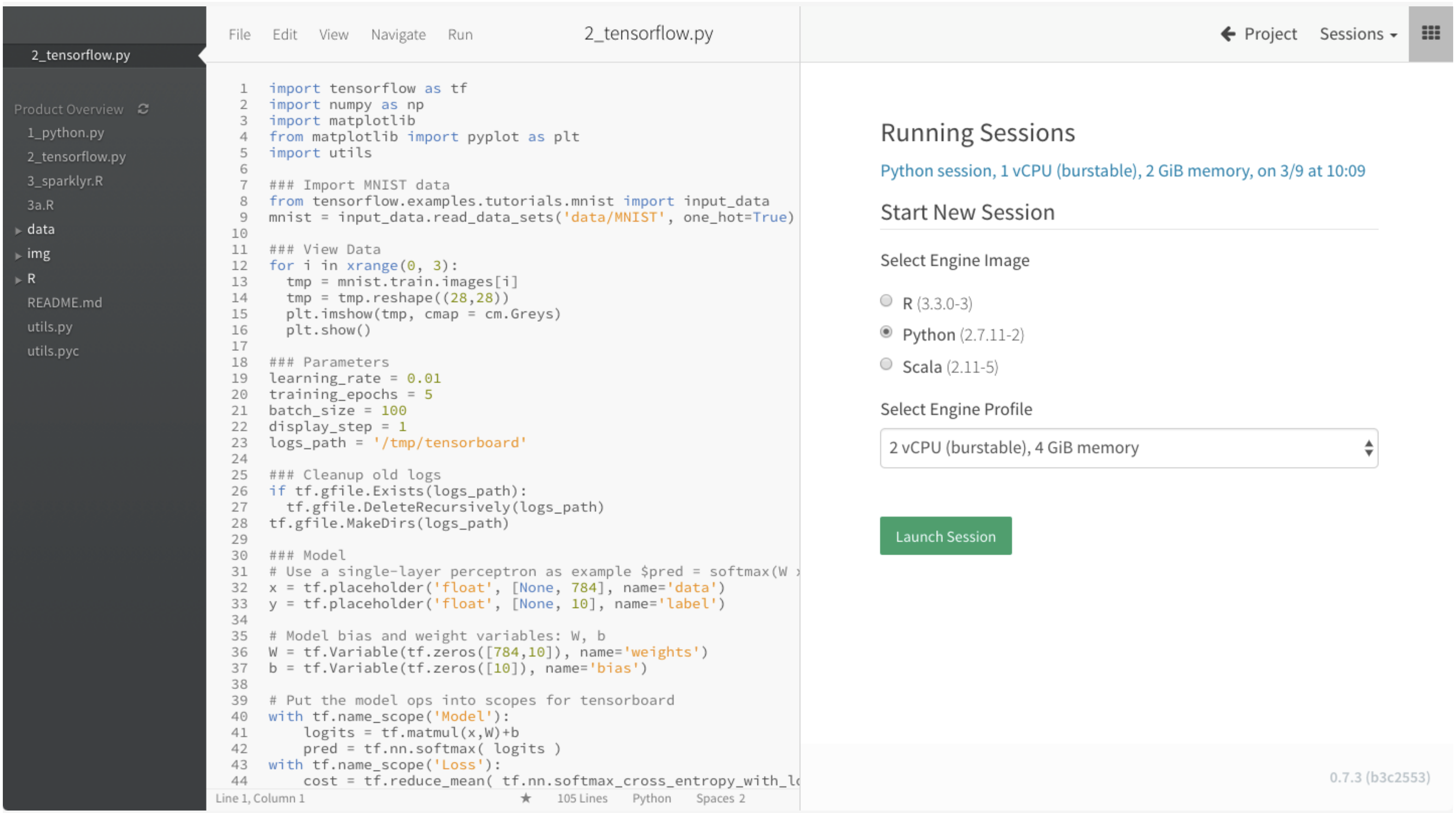
Black Box

Thank You

Justin Erickson


cloudera
DST
Projects
Project quick find
DST
Projects
Jobs
Sessions
Settings
Admin
0
sessions running
2
jobs running
3
0
vCPU
80.00
6
0
GB
263.86
Projects
Creator
New Project
Product Overview
Introduction to Cloudera Data Science Workbench
By Matt Brandwein. Last worked on just now. forked from Product Demo
1
running
Data Analysis in Python
By Matt Brandwein. Last worked on just now.
1
running
Health Data Demo
By Matt Brandwein. Last worked on 14 minutes ago. forked from Health Data Demo
0
running
Intel BigDL Experiments
By Matt Brandwein. Last worked on 2 weeks ago.
0
running
Deep Learning with TensorFlow
By Matt Brandwein. Last worked on 3 weeks ago.
0
running
RImpala
By Tristan Zajonc. Last worked on 5 weeks ago.
0
running
R Analysis
By Tristan Zajonc. Last worked on January 24.
0
running
HDFS IO
Reading and writing data from HDFS.
By Tristan Zajonc. Last worked on January 23.
0
running


Account
DST
Product Overview
Project quick find
DST
Overview
Jobs
Sessions
Files
Team
Settings
Product Overview
0
Fork
Open Workbench
Introduction to Cloudera Data Science Workbench
forked from Product Demo
Jobs
Creator
Name
Runs / Failures
Duration
Status
Latest Run
Actions
Nightly Report
1 / 0
00:08
Success
7 minutes ago
Files
Download
New
Upload
Name
Size
Last Modified
data
-
17 hours ago
img
-
17 hours ago
R
-
17 hours ago
1_python.py
2.95 kB
17 hours ago
2_tensorflow.py
3.38 kB
17 hours ago
3_sparklyr.R
2.23 kB
17 hours ago
3a.R
356 B
17 hours ago
README.md
134 B
17 hours ago
utils.py
1.04 kB
17 hours ago
utils.pyc
1.43 kB
17 hours ago


2_tensorflow.py
Product Overview
1_python.py
2_tensorflow.py
3_sparklyr.R
3a.R
data
img
R
README.md
utils.py
utils.pyc
File Edit View Navigate Run
2_tensorflow.py
import tensorflow as tf
import numpy as np
import matplotlib
from matplotlib import pyplot as plt
import utils
Import MNIST data
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets('data/MNIST', one_hot=True)
View Data
for i in xrange(0, 3):
tmp = mnist.train.images[i]
tmp = tmp.reshape((28,28))
plt.imshow(tmp, cmap = cm.Greys)
plt.show()
Parameters
learning_rate = 0.01
training_epochs = 5
batch_size = 100
display_step = 1
logs_path = '/tmp/tensorboard'
Cleanup old logs
if tf.gfile.Exists(logs_path):
tf.gfile.DeleteRecursively(logs_path)
tf.gfile.MakeDirs(logs_path)
Model
# Use a single-layer perceptron as example $pred = softmax(W
x = tf.placeholder('float', [None, 784], name='data')
y = tf.placeholder('float', [None, 10], name='label')
# Model bias and weight variables: W, b
W = tf.Variable(tf.zeros([784,10]), name='weights')
b = tf.Variable(tf.zeros([10]), name='bias')
# Put the model ops into scopes for tensorboard
with tf.name_scope('Model'):
logits = tf.matmul(x,W)+b
pred = tf.nn.softmax(logits)
with tf.name_scope('Loss'):
cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_l
Line 1, Column 1
105 Lines
Python
Spaces 2
Project
Sessions
Running Sessions
Python session, 1 vCPU (burstable), 2 GiB memory, on 3/9 at 10:09
Start New Session
Select Engine Image
R (3.3.0-3)
Python (2.7.11-2)
Scala (2.11-5)
Select Engine Profile
2 vCPU (burstable), 4 GiB memory
Launch Session
0.7.3 (b3c2553)

File Edit View Navigate Run ▶ 1_python.py

```python
display_charts(data, chart_type="stock", title="DJIA vs. Debt Query Volume", secon
seaborn.lmplot("debt", "djia", data=data, size=7)

# Detect if search volume is increasing or decreasing in
# any given week by forming a moving average and testing if the current value
# crosses the moving average of the past 3 weeks.
#
# Let's first compute the moving average.

data['debt_mavg'] = data.debt.rolling(window=3, center=False).mean()
data.head()

# Since we want to see if the current value is above the moving average of the
# *preceeding* weeks, we have to shift the moving average timeseries forward by on

data['debt_mavg'] = data.debt_mavg.shift(1)
data.head()

# Generate Orders
# ===============
#
# We use Google Trends to determine how many searches have been
# carried out for a specific search term such as debt in week,
# where Google defines weeks as ending on a Sunday, relative to the total
# number of searches carried out on Google during that time.
#
# We implement the strategy of selling when debt searches exceed
# the moving average and buying when debt searchers fall below the moving
# average.

data['order'] = 0
data.loc[data.debt > data.debt_mavg, 'order'] = -1
data.loc[data.debt < data.debt_mavg, 'order'] = -1
data.head()

# Compute Returns
# ===============

data['ret_djia'] = data.djia.pct_change()
data.head()

# Returns at week `t` are relative to week `t-1`. However, we are buying at
# week `t` and selling at week `t+1`, so we have to adjust by shifting
# the returns upward.

data['ret_djia'] = data['ret_djia'].shift(-1)

# The algorithm that is used by the authors makes a decision every Monday of
```

Line 1, Column 1 ★ 86 Lines Python Spaces 2

Project >_ Terminal access Clear Interrupt Stop Sessions

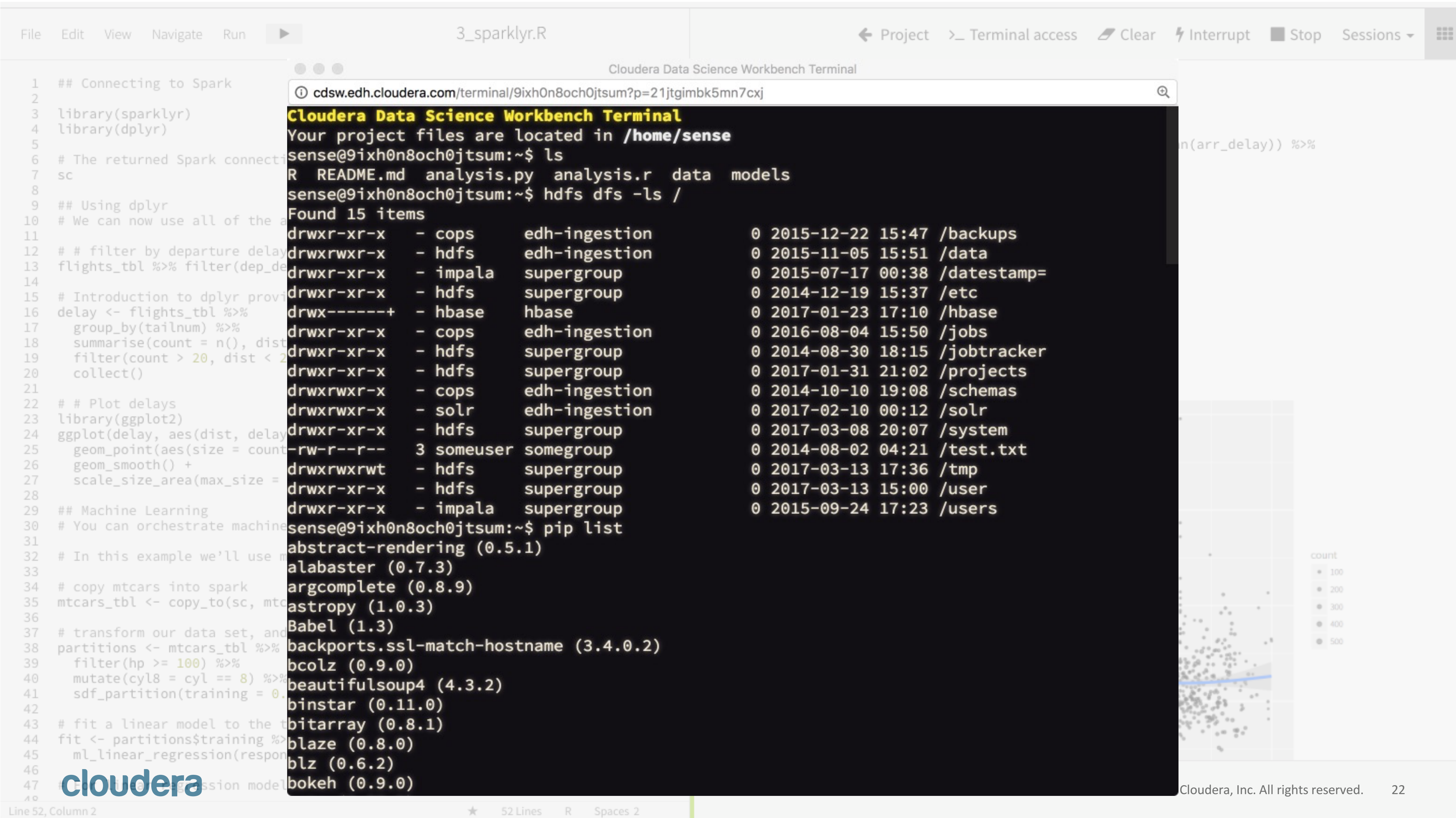
```
> data.head()
```

	djia	debt
Date		
2004-01-14	10485.18	0.210000
2004-01-22	10528.66	0.210000
2004-01-28	10702.51	0.210000
2004-02-04	10499.18	0.213333
2004-02-11	10579.03	0.200000

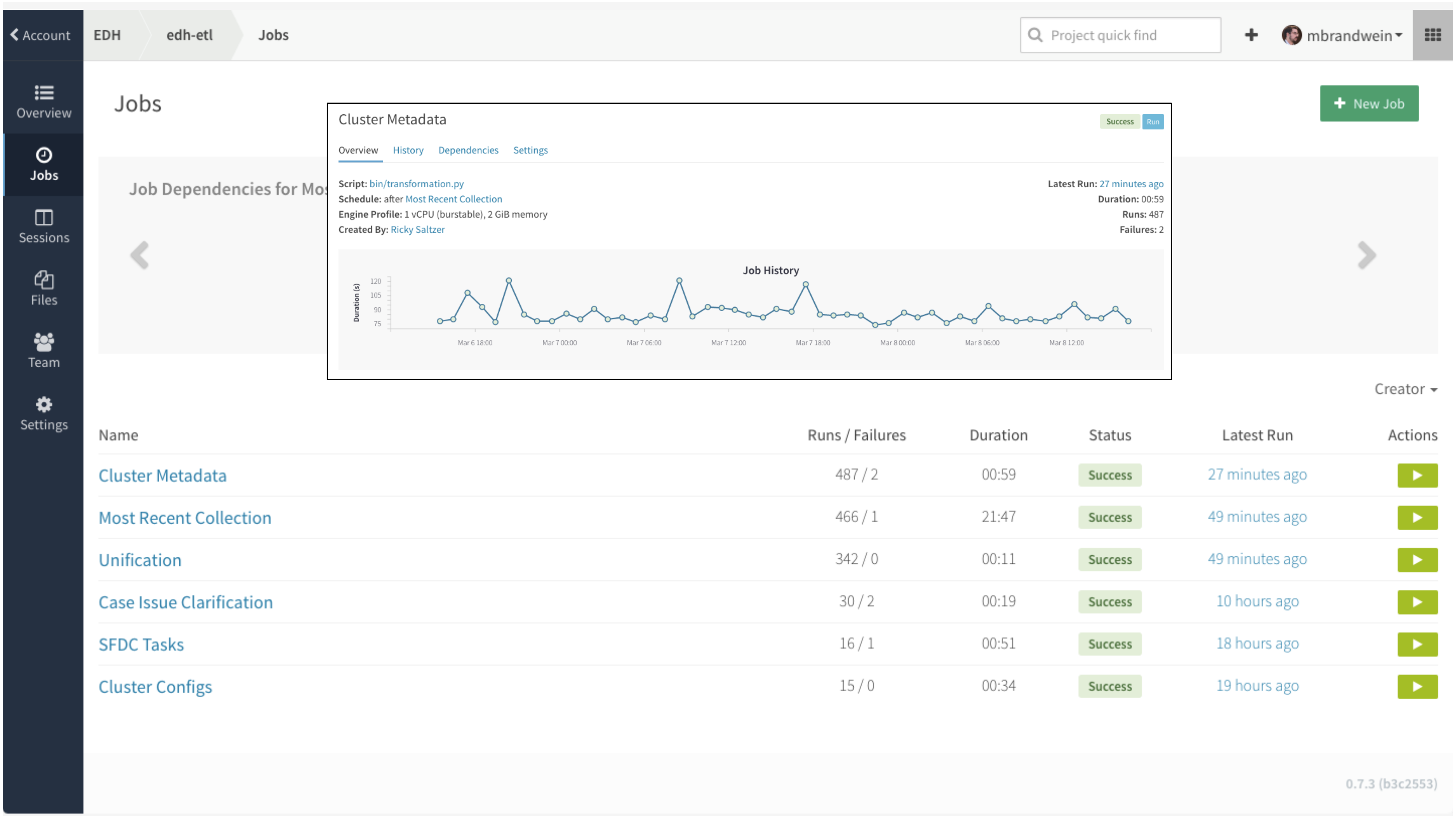
Show DJIA vs. debt related query volume.

```
> display_charts(data, chart_type="stock", title="DJIA vs. Debt Query Volume",
  secondary_y="debt")
```


File
Edit
View
Navigate
Run
3_sparklyr.R
Connecting to Spark
library(sparklyr)
library(dplyr)
# The returned Spark connection (sc) provides a remote dplyr data source to the
sc
Using dplyr
We can now use all of the available dplyr verbs against the tables within the
filter by departure delay
flights_tbl %>% filter(dep_delay == 2)
# Introduction to dplyr provides additional dplyr examples you can try. For exan
delay <- flights_tbl %>%
group_by(tailnum) %>%
summarise(count = n(), dist = mean(distance), delay = mean(arr_delay)) %>%
filter(count > 20, dist < 2000, !is.na(delay)) %>%
collect()
# # Plot delays
library(ggplot2)
ggplot(delay, aes(dist, delay)) +
geom_point(aes(size = count), alpha = 1/2) +
geom_smooth() +
scale_size_area(max_size = 2)
Machine Learning
# You can orchestrate machine learning algorithms in a Spark cluster via the mac
# In this example we'll use ml_linear_regression to fit a linear regression mode
# copy mtcars into spark
mtcars_tbl <- copy_to(sc, mtcars)
# transform our data set, and then partition into 'training', 'test'
partitions <- mtcars_tbl %>%
filter(hp >= 100) %>%
mutate(cyl8 = cyl == 8) %>%
sdf_partition(training = 0.5, test = 0.5, seed = 1099)
# fit a linear model to the training dataset
fit <- partitions$training %>%
ml_linear_regression(response = "mpg", features = c("wt", "cyl"))
# For linear regression models produced by Spark, we can use summary() to learn
Line 52, Column 2
52 Lines
R
Spaces 2
Project
Terminal access
Clear
Interrupt
Stop
Sessions
from the tutorial which plots data on flight delays:
> delay <- flights_tbl %>%
group_by(tailnum) %>%
summarise(count = n(), dist = mean(distance), delay = mean(arr_delay)) %>%
filter(count > 20, dist < 2000, !is.na(delay)) %>%
collect()
Plot delays
> library(ggplot2)
> ggplot(delay, aes(dist, delay)) +
geom_point(aes(size = count), alpha = 1/2) +
geom_smooth() +
scale_size_area(max_size = 2)
`geom_smooth()` using method = 'gam'
60
40
20
0
delay
count
100
200
300
400
500


File Edit View Navigate Run
3_sparklyr.R
Project
Terminal access
Clear
Interrupt
Stop
Sessions
Cloudera Data Science Workbench Terminal
cdsw.edh.cloudera.com/terminal/9ixh0n8och0jtsum?p=21jtgimbk5mn7cxj
Cloudera Data Science Workbench Terminal
Your project files are located in /home/sense
sense@9ixh0n8och0jtsum:~$ ls
R README.md analysis.py analysis.r data models
sense@9ixh0n8och0jtsum:~$ hdfs dfs -ls /
Found 15 items
drwxr-xr-x   - cops   edh-ingestion   0 2015-12-22 15:47 /backups
drwxrwxr-x   - hdfs   edh-ingestion   0 2015-11-05 15:51 /data
drwxr-xr-x   - impala   supergroup   0 2015-07-17 00:38 /datestamp=
drwxr-xr-x   - hdfs   supergroup   0 2014-12-19 15:37 /etc
drwx------+ - hbase   hbase   0 2017-01-23 17:10 /hbase
drwxr-xr-x   - cops   edh-ingestion   0 2016-08-04 15:50 /jobs
drwxr-xr-x   - hdfs   supergroup   0 2014-08-30 18:15 /jobtracker
drwxr-xr-x   - hdfs   supergroup   0 2017-01-31 21:02 /projects
drwxrwxr-x   - cops   edh-ingestion   0 2014-10-10 19:08 /schemas
drwxrwxr-x   - solr   edh-ingestion   0 2017-02-10 00:12 /solr
drwxr-xr-x   - hdfs   supergroup   0 2017-03-08 20:07 /system
-rw-r--r--   3 someuser somegroup   0 2014-08-02 04:21 /test.txt
drwxrwxrwt   - hdfs   supergroup   0 2017-03-13 17:36 /tmp
drwxr-xr-x   - hdfs   supergroup   0 2017-03-13 15:00 /user
drwxr-xr-x   - impala   supergroup   0 2015-09-24 17:23 /users
sense@9ixh0n8och0jtsum:~$ pip list
abstract-rendering (0.5.1)
alabaster (0.7.3)
argcomplete (0.8.9)
astropy (1.0.3)
Babel (1.3)
backports.ssl-match-hostname (3.4.0.2)
bcolz (0.9.0)
beautifulsoup4 (4.3.2)
binstar (0.11.0)
bitarray (0.8.1)
blaze (0.8.0)
blz (0.6.2)
bokeh (0.9.0)
Line 52, Column 2
52 Lines
R
Spaces 2

cloudera



Account
EDH
edh-etl
Jobs
Project quick find
mbrandwein
Overview
Jobs
Sessions
Files
Team
Settings
Jobs
New Job
Job Dependencies for Mo
Cluster Metadata
Success
Run
Overview
History
Dependencies
Settings
Script: bin/transformation.py
Schedule: after Most Recent Collection
Engine Profile: 1 vCPU (burstable), 2 GiB memory
Created By: Ricky Saltzer
Latest Run: 27 minutes ago
Duration: 00:59
Runs: 487
Failures: 2
Job History
Duration (s)
120
105
90
75
Mar 6 18:00
Mar 7 00:00
Mar 7 06:00
Mar 7 12:00
Mar 7 18:00
Mar 8 00:00
Mar 8 06:00
Mar 8 12:00
Creator
Name
Runs / Failures
Duration
Status
Latest Run
Actions
Cluster Metadata
487 / 2
00:59
Success
27 minutes ago
Most Recent Collection
466 / 1
21:47
Success
49 minutes ago
Unification
342 / 0
00:11
Success
49 minutes ago
Case Issue Clarification
30 / 2
00:19
Success
10 hours ago
SFDC Tasks
16 / 1
00:51
Success
18 hours ago
Cluster Configs
15 / 0
00:34
Success
19 hours ago
0.7.3 (b3c2553)