Navigating the Data Science Python Ecosystem

Christine Doig
Senior Data Scientist
Continuum Analytics
ANAconda® is....
Leading Open Data Science Platform
Powered by Python, the fastest growing data science language

• Accelerate Time-to-Value
• Connect Data, Analytics & Compute
• Empower Data Science Teams
NAVIGATING THE DATA SCIENCE PYTHON ECOSYSTEM
NAVIGATING THE DATA SCIENCE PYTHON ECOSYSTEM
NAVIGATING THE DATA SCIENCE PYTHON ECOSYSTEM

1. Introduction to Data Science
NAVIGATING THE DATA SCIENCE PYTHON ECOSYSTEM

1. Introduction to Data Science
NAVIGATING THE DATA SCIENCE PYTHON ECOSYSTEM

1. Introduction to Data Science
2. The State of Python for Data Science
NAVIGATING THE DATA SCIENCE PYTHON ECOSYSTEM

1. Introduction to Data Science
2. The State of Python for Data Science
3 From data to models to applications

NAVIGATING THE DATA SCIENCE PYTHON ECOSYSTEM

1 Introduction to Data Science
2 The State of Python for Data Science
INTRODUCTION TO DATA SCIENCE
The Data Science Venn Diagram

The Data Science Venn Diagram Revisited

- Machine Learning
- Visualization
- Big Data
- BI / ETL
- Scientific computing
- CS / Programming

Data Science
The Data Science Venn Diagram Revisited

Machine Learning
- Bayesian
- Statistics
- Neural Networks
- Deep Learning
- R
- SAS

Visualization
- Dashboards
- Tableau
- D3
- MS Excel
- SQL
- Postgres

BI / ETL
- Data warehouse
- Python
- Java

Big Data
- Hadoop
- Spark
- Hive
- MPI
- GPUs

Scientific computing
- Array computing
- Virtualization
- C++

CS / Programming
- Web development
- JavaScript

Data Science
- Tableau
- D3
- SQL

- Postgres
- Python
- Java

- Clojure
- JS

Array computing
- Virtualization
- C++

Data warehouse
- Python
- Java

NLP
- Computer Vision
- NLP
- HDFS

Web development
- JavaScript

- Docker

Software best practices
- Virtualization
- C++
The Data Science Venn Diagram Revisited

Machine Learning

Visualization

Big Data

BI / ETL

CS / Programming

Science computing

Data

R

SAS

Dashboards

D3

Tableau

MS Excel

SQL

Postgres

Data warehouse

Python

JS

Java

Web development

Software best practices

Virtualization

Array computing

C++

Docker

Hadoop

HDFS

Spark

Hive

Storm

MPI

GPUs

Matlab

Hive

MPL

HDFS

Spark

Storm

MPI

GPUs

Matlab
Data Scientists come with different skills and backgrounds
Data Scientists come with different skills and backgrounds
Data Science is about building teams
Data Science is about building teams
Statistician / Analyst
Research / Computational Scientist
Developer / Engineer

Works with

Thinks data

Delivers
Statistician / Analyst

Works with:
- R
- SAS
- Tableau
- SQL
- MS Excel

Thinks data:

Delivers:

Research / Computational Scientist

Works with:
- MPI
- Matlab
- Fortran
- C / C++

Thinks data:

Delivers:

Developer / Engineer

Works with:
- Java
- JS
- Redshift
- HDFS
- Docker
- Postgres

Thinks data:

Delivers:
Statistician / Analyst
Works with:
- R
- SAS
- SQL
- Tableau
- MS Excel

Statistician / Analyst

Research / Computational Scientist
Works with:
- MPI
- Matlab
- Fortran
- C / C++

Research / Computational Scientist

Developer / Engineer
Works with:
- Java
- JS
- Redshift
- Docker
- Postgres

Developer / Engineer

Dataframes & Tables
- Dataframes & tables

Arrays & Data Structures
- Arrays & data structures

Data Structures & JSON
- Data structures & JSON
<table>
<thead>
<tr>
<th>Role</th>
<th>Works with</th>
<th>Thinks data</th>
<th>Delivers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statistician / Analyst</td>
<td>R, SAS, Tableau, MS Excel</td>
<td>dataframes &amp; tables</td>
<td>insights, predictions, visualizations</td>
</tr>
<tr>
<td>Research / Computational Scientist</td>
<td>MPI, C / C++, Matlab, Fortran</td>
<td>arrays &amp; data structures</td>
<td>algorithms, libraries, performance</td>
</tr>
<tr>
<td>Developer / Engineer</td>
<td>Java, JS, Redshift, Docker, Postgres</td>
<td>data structures &amp; JSON</td>
<td>software, applications, containers</td>
</tr>
</tbody>
</table>
Challenges

• Get diverse data teams (languages, tools, data models, deliverables…) to collaborate effectively

• Move Data Scientist (Stats / Analyst) to use Big Data infrastructure

• Deploy predictive models into production applications

• Share insights with decision makers
Challenges

- Collaboration
- Big Data
- Deployment
- Sharing insights
The Data Science team workflow
The Data Science team workflow

- Implements a predictive modeling algorithm

Algorithm (e.g. SVM)
The Data Science team workflow

• Implements a predictive modeling algorithm

• Fits different models with different parameters to find the best one

Algorithm (e.g. SVM)
Algorithm (e.g. Neural Network)
Algorithm (e.g. Logistic Regression)
The Data Science team workflow

- Implements a predictive modeling algorithm
- Fits different models with different parameters to find the best one

Algorithm (e.g. SVM)
Algorithm (e.g. Neural Network)
Algorithm (e.g. Logistic Regression)

scripts to transform and select data
The Data Science team workflow

- Implements a predictive modeling algorithm
- Fits different models with different parameters to find the best one
- Build and deploy an application that uses the predictive model

Algorithm (e.g. SVM)
Algorithm (e.g. Neural Network)
Algorithm (e.g. Logistic Regression)

Show results to domain expert / decision maker
Integrate with existing deployment system

scripts to transform and select data + α
Why Open Data Science?
Why Open Data Science?

• Availability
• Innovation
• Interoperability
• Transparency
Why Python?

- Machine Learning
- Big Data
- Visualization
- BI / ETL
- CS / Programming
- Scientific Computing

Statistician / Analyst
Research / Computational Scientist
Developer / Engineer

Python
R
SAS
Tableau
SQL
MPI
C / C++
Java
JS
HDFS
Docker
Postgres

Algorithm (e.g. SVM)
Algorithm (e.g. Logistic Regression)
Algorithm (e.g. Neural Network)

script to transform and select data
THE STATE OF PYTHON FOR DATA SCIENCE
The state of Python for Data Science

- Machine Learning
  - TensorFlow
  - Theano
  - Dask

- Visualization
  - Matplotlib
  - Bokeh

- Big Data
  - Spark
  - Hadoop

- BI / ETL
  - Pandas
  - Blaze
  - Numba

- Scientific computing
  - SciPy
  - NumPy
  - PyTables

- CS / Programming
  - Anaconda

18
• Anaconda distribution: Python distribution that includes 150+ packages for data science (in the installer)
• conda: Cross-platform and language agnostic package and environment manager
• Miniconda: Minified version of Anaconda, with just Python and conda.
• Anaconda Cloud: Cloud service to host and share public and private packages, environments and notebooks
• conda environments: custom isolated sandboxes to easily reproduce and share data science projects
Why Anaconda distribution?

- Easy to install on all platforms
- Trusted by industry leaders
- Large user base: 3M+ downloads
- BSD license
- Extensible - easily build, share and install proprietary libraries with Anaconda Cloud
- Language agnostic - Python, R, Scala…
- Allows isolated custom sandboxes with different versions of packages
... and an amazing Anaconda community!
Python Data Science workflow

Interactive DS Environment
- jupyter

Data munging, prep, tidying
- pandas

Data visualization
- matplotlib

Data modeling
- scikit-learn
The Jupyter Notebook is a web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text.
Sharing insights with Decision makers

From text, code and visualizations directly to slides
Sharing insights with Decision makers

From text, code and visualizations directly to slides
Continuum Analytics contributions to the Python ecosystem

**Bokeh**
- Web interactive data visualizations (no JS)

**Datashader**
- Graphics pipeline system for creating meaningful representations of large amounts of data

**Dask**
- Parallel computing framework

**Blaze**
- Unified expression system to query heterogeneous data
Bokeh

Interactive visualization framework that targets modern web browsers for presentation

• No JavaScript
• Python, R, Scala and Lua bindings
• Easy to embed in web applications
• Server apps: data can be updated, and UI and selection events can be processed to trigger more visual updates.

Bokeh

Interactive visualization framework that targets modern web browsers for presentation

- No JavaScript
- Python, R, Scala and Lua bindings
- Easy to embed in web applications
- Server apps: data can be updated, and UI and selection events can be processed to trigger more visual updates.

Bokeh

Interactive visualization framework that targets modern web browsers for presentation

• No JavaScript
• Python, R, Scala and Lua bindings
• Easy to embed in web applications
• Server apps: data can be updated, and UI and selection events can be processed to trigger more visual updates.

Interactive visualization framework that targets modern web browsers for presentation

- No JavaScript
- Python, R, Scala and Lua bindings
- Easy to embed in web applications
- Server apps: data can be updated, and UI and selection events can be processed to trigger more visual updates.

Large data visualizations

69795 tweets on 2012-02-16
Datashader

Overplotting:

Undersampling:

https://anaconda.org/jbednar/plotting_pitfalls/notebook
Datashader

graphics pipeline system for creating meaningful representations of large amounts of data

• Provides automatic, nearly parameter-free visualization of datasets
• Allows extensive customization of each step in the data-processing pipeline
• Supports automatic downsampling and re-rendering with Bokeh and the Jupyter notebook
• Works well with dask and numba to handle very large datasets in and out of core (with examples using billions of datapoints)

https://github.com/bokeh/datashader

NYC census data by race
Datashader

More examples:

https://anaconda.org/jbednar/notebooks
Datashader

More examples:

https://anaconda.org/jbednar/notebooks
Datashader

More examples:

https://anaconda.org/jbednar/notebooks
Moving from small data to big data

Client Machine

Compute Node

Client Machine

Compute Node

Compute Node

Head Node

Small Data

pandas
NumPy

Big Data

Spark
Hadoop

ANAconda
Moving from small data to big data

Small Data

Big Data

Client Machine

Compute Node

Compute Node

Compute Node

Head Node

pandas

NumPy

Spark

hadoop

ANAconda
Moving from small data to big data
Moving from small data to big data

Small Data → Dask → Big Data

- Client Machine
- Compute Node
- Head Node
- Small Data
- Big Data
- pandas
- NumPy
- Anaconda
- Spark
- Hadoop
Dask Dataframes

```python
>>> import pandas as pd
>>> df = pd.read_csv('iris.csv')
>>> df.head()
sepal_length  sepal_width  petal_length  petal_width  species
0 5.1 3.5 1.4 0.2  Iris-setosa
1 4.9 3.0 1.4 0.2  Iris-setosa
2 4.7 3.2 1.3 0.2  Iris-setosa
3 4.6 3.1 1.5 0.2  Iris-setosa
4 5.0 3.6 1.4 0.2  Iris-setosa

>>> max_sepal_length_setosa = df[df.species == 'setosa'].sepal_length.max()
5.7999999999999998
```

Dask

```python
>>> import dask.dataframe as dd
>>> ddf = dd.read_csv('*.csv')
>>> ddf.head()
sepal_length  sepal_width  petal_length  petal_width  species
0 5.1 3.5 1.4 0.2  Iris-setosa
1 4.9 3.0 1.4 0.2  Iris-setosa
2 4.7 3.2 1.3 0.2  Iris-setosa
3 4.6 3.1 1.5 0.2  Iris-setosa
4 5.0 3.6 1.4 0.2  Iris-setosa
...  

>>> d_max_sepal_length_setosa = ddf[ddf.species == 'setosa'].sepal_length.max().compute()
5.7999999999999998
```
Dask Arrays

```python
>>> import numpy as np

>>> np_ones = np.ones((5000, 1000))

>>> np_ones
array([[ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       ...,
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.]])

>>> np_y = np.log(np_ones + 1)[:5].sum(axis=1)

>>> np_y
array([ 693.14718056,  693.14718056,  693.14718056,
        693.14718056, ...]),

>>> import dask.array as da

>>> da_ones = da.ones((5000000, 1000000),
                     chunks=(1000, 1000))

>>> da_ones.compute()
array([[ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       ...,
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.],
       [ 1.,  1.,  1., ...,  1.,  1.,  1.]])

>>> da_y = da.log(da_ones + 1)[:5].sum(axis=1)

>>> np_da_y = np.array(da_y)  # fits in memory
array([ 693.14718056,  693.14718056,  693.14718056,
        693.14718056, ...]),

# Result doesn’t fit in memory
>>> da_y.to_hdf5('myfile.hdf5', 'result')
```
Dask

A parallel computing framework through task scheduling and blocked algorithms

• **Familiar**: Implements parallel NumPy and Pandas objects
• **Fast**: Optimized for demanding for numerical applications
• **Flexible**: for sophisticated and messy algorithms
• **Scales up**: Runs resiliently on clusters of 100s of machines
• **Scales down**: Pragmatic in a single process on a laptop
• **Interactive**: Responsive and fast for interactive data science

Dask

A parallel computing framework through task scheduling and blocked algorithms

- **Familiar**: Implements parallel NumPy and Pandas objects
- **Fast**: Optimized for demanding for numerical applications
- **Flexible**: for sophisticated and messy algorithms
- **Scales up**: Runs resiliently on clusters of 100s of machines
- **Scales down**: Pragmatic in a single process on a laptop
- **Interactive**: Responsive and fast for interactive data science

Dask

A parallel computing framework through task scheduling and blocked algorithms

- **Familiar**: Implements parallel NumPy and Pandas objects
- **Fast**: Optimized for demanding for numerical applications
- **Flexible**: for sophisticated and messy algorithms
- **Scales up**: Runs resiliently on clusters of 100s of machines
- **Scales down**: Pragmatic in a single process on a laptop
- **Interactive**: Responsive and fast for interactive data science

Dask

A parallel computing framework through task scheduling and blocked algorithms

- **Familiar**: Implements parallel NumPy and Pandas objects
- **Fast**: Optimized for demanding for numerical applications
- **Flexible**: for sophisticated and messy algorithms
- **Scales up**: Runs resiliently on clusters of 100s of machines
- **Scales down**: Pragmatic in a single process on a laptop
- **Interactive**: Responsive and fast for interactive data science

Dask

A parallel computing framework through task scheduling and blocked algorithms

- **Familiar**: Implements parallel NumPy and Pandas objects
- **Fast**: Optimized for demanding for numerical applications
- **Flexible**: for sophisticated and messy algorithms
- **Scales up**: Runs resiliently on clusters of 100s of machines
- **Scales down**: Pragmatic in a single process on a laptop
- **Interactive**: Responsive and fast for interactive data science

Dask

A parallel computing framework through task scheduling and blocked algorithms

- **Familiar**: Implements parallel NumPy and Pandas objects
- **Fast**: Optimized for demanding for numerical applications
- **Flexible**: for sophisticated and messy algorithms
- **Scales up**: Runs resiliently on clusters of 100s of machines
- **Scales down**: Pragmatic in a single process on a laptop
- **Interactive**: Responsive and fast for interactive data science

Querying heterogenous data storage systems
Querying heterogenous data storage systems

Client Machine

Blaze

Write once, query anywhere!

- Flat files (CSV...)
- SQL
- NOSQL (mongoDB)
- HDFS
Conda and Docker

Data Scientist

Analysis 1
conda env 1
Laptop, server, EC2 instance

Analysis 2
conda env 2

Analysis 3
conda env 3

Development

Analysis 1
conda env 1
Docker container
Laptop, server, EC2 instance

Deployment
Conda and Docker

Data Scientist

Analysis 1
conda env 1
Laptop, server, EC2 instance

Analysis 2
conda env 2

Analysis 3
conda env 3

Laptop, server, EC2 instance

DevOps

Analysis 1
conda env 1

Docker container

Laptop, server, EC2 instance

Development

Deployment

Data Science Development

Deployment
Conda and Docker

Public | Automated Build

continuumio/miniconda

Last pushed: a month ago

Short Description
Powerful and flexible package manager

Full Description
docker-miniconda
Docker container with a bootstrapped miniconda installed and ready to use.
This installs conda and Python 2.7 into the /opt/conda environment and ensures that the default user has that on their path.

Usage
FROM DATA TO MODELS TO APPLICATIONS
Data Science team workflow
Data Science team workflow

- Present and tell your data story to decision makers
Data Science team workflow

- Present and tell your data story to decision makers
Data Science team workflow

- Setup your environment locally (single node) on any platform or a cluster
- Present and tell your data story to decision makers
Data Science team workflow

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Present and tell your data story to decision makers
Data Science team workflow

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Present and tell your data story to decision makers
Data Science team workflow

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Present and tell your data story to decision makers
- Use same expression to query data no matter where it lives
Data Science team workflow

• Scale your data processing, without changing frameworks or paradigms

• Setup your environment locally (single node) on any platform or a cluster

• Use same expression to query data no matter where it lives

• Present and tell your data story to decision makers

• Build large scale meaningful interactive data visualizations
Data Science team workflow

• Scale your data processing, without changing frameworks or paradigms
• Setup your environment locally (single node) on any platform or a cluster
• Present and tell your data story to decision makers
• Use same expression to query data no matter where it lives
• Deploy your interactive analytical/predictive application
• Build large scale meaningful interactive data visualizations
Challenges (reminder)

- Collaboration
- Big Data
- Deployment
- Sharing insights
Challenges revisited

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Present and tell your data story to decision makers
- Use same expression to query data no matter where it lives
- Build large scale meaningful interactive data visualizations
- Deploy your interactive analytical/predictive application

- Pandas
- Numpy
- Dask
- Conda env
- HDFS
- Databases
- Blaze
- Bokeh + datashader
- Anaconda
Challenges revisited

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Use same expression to query data no matter where it lives
- Deploy your interactive analytical/predictive application
- Build large scale meaningful interactive data visualizations
- Present and tell your data story to decision makers

Sharing insights
Challenges revisited

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Use same expression to query data no matter where it lives
- Deploy your interactive analytical/predictive application
- Build large scale meaningful interactive data visualizations

Big Data

- Databases
- HDFS
- Blaze
- Bokeh + datashader

Sharing insights

- Present and tell your data story to decision makers
- Conda env
- Pandas
- NumPy
- Anaconda
- Dask
Challenges revisited

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Use same expression to query data no matter where it lives
- Deploy your interactive analytical/predictive application
- Build large scale meaningful interactive data visualizations

Big Data

Databases
HDFS
Blaze

Client Machine
Compute Node
Compute Node
Compute Node
Head Node
dask

Collaboration

conda env

Sharing insights

pandas
NumPy
Bokeh + datashader

conda env
Challenges revisited

- Scale your data processing, without changing frameworks or paradigms
- Setup your environment locally (single node) on any platform or a cluster
- Use same expression to query data no matter where it lives
- Present and tell your data story to decision makers
- Deploy your interactive analytical/predictive application
- Build large scale meaningful interactive data visualizations
Thank you!

Twitter: @ch_doig

Questions?