GENERAL ASSEMBLY

SCIENTIFIC PYTHON

Mason Gallo, Data Scientist

AGENDA

- data munging overview
- numpy introduction
- pandas introduction
- matplotlib introduction
- pandas exercises

CLASS 3 – SCIENTIFIC PYTHON

- Understand the roles of numpy, pandas, matplotlib
- Import and export data using Pandas
- Comfort with the major data structures in Pandas

DATA MUNGING == EDA?

INVENTOR OF EDA

“Exploratory Data Analysis is an attitude, a state of flexibility, a willingness to look for those things that we believe are not there, as well as those we believe to be there.”

- John Tukey, Professor Emeritus, Yale

THE GOALS OF EDA

- Gain intuition
- Sanity check
- Handle variable types
- Identify and treat missing data
- Identify and treat outliers
- Summarize the data
- Visualize

THE GOALS OF DATA MUNGING

- Prepare the data
- Clean data
- Create new features
- Gain intuition
- Sanity check
- Handle variable types
- Identify and treat missing data
- Identify and treat outliers
- Summarize the data

EDA COMES FROM STATISTICIANS USED TO MUCH CLEANER DATASETS

- Data munging still covers everything that is considered EDA
- Usually involves a lot more time programmatically “cleaning” the data
- Statisticians typically receive the data ready to be modeled
- Data Scientists need to clean it first!

DATA MUNGING TOOLBOX - SCIENTIFIC PYTHON

- numpy provides fast matrix computation
- pandas provides easy manipulation through dataframes
- matplotlib provides visualization (I’m not sure I consider it easy)

INTRODUCING NUMPY

A COLLECTION OF OPTIMIZED NUMERICAL OPERATIONS

- Based in C - very fast!
- Major data structures are the array and the scalar
- Vectorization means avoiding slow for loops
- Provides tons of functions for linear algebra, such as the dot product
- Serves as backbone for many other libraries, such as pandas and matplotlib

All numpy methods available here: http://docs.scipy.org/doc/numpy/genindex.html

TRY IT OUT: ARRAYS AND SCALARS

THE FUNDAMENTAL DATA STRUCTURES: ARRAY AND SCALAR

- Open your terminal, type ipython and copy the below:

```
import numpy as np
array_a = np.array([1, 2, 3, 4])
array_b = np.array([2, 4, 6, 8])
scalar = 10
```

All numpy methods available here: http://docs.scipy.org/doc/numpy/genindex.html

NUMPY EXAMPLE: VECTORIZATION

THE CENTRAL TENET OF NUMPY IS VECTORIZATION - AVOIDING LOOPS

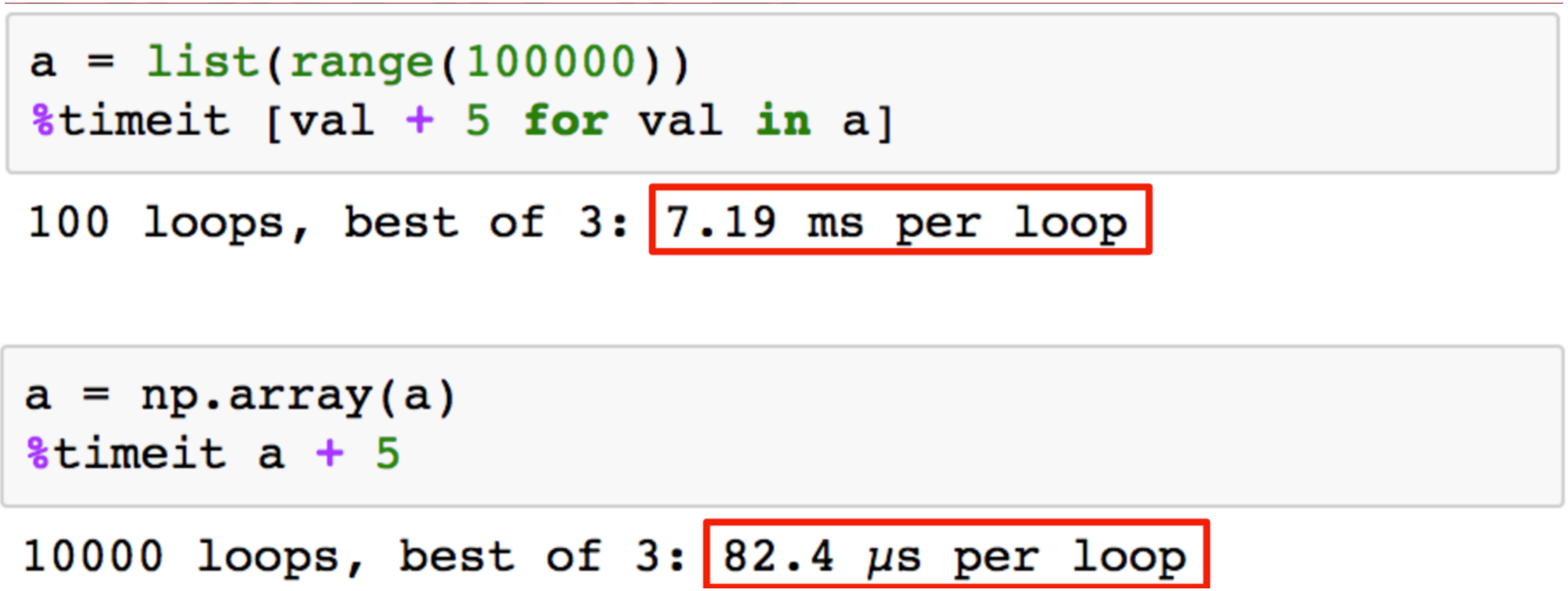
- We can do operations on an entire array rather than looping
- This is very useful when we're working with large datasets!

```
array_a + array_b
scalar * array_a
```

All numpy methods available here: http://docs.scipy.org/doc/numpy/genindex.html

NUMPY IS REALLY FAST

SAVINGS OF 100X!



```
a = list(range(100000))
%timeit [val + 5 for val in a]
```

```
100 loops, best of 3: 7.19 ms per loop
```

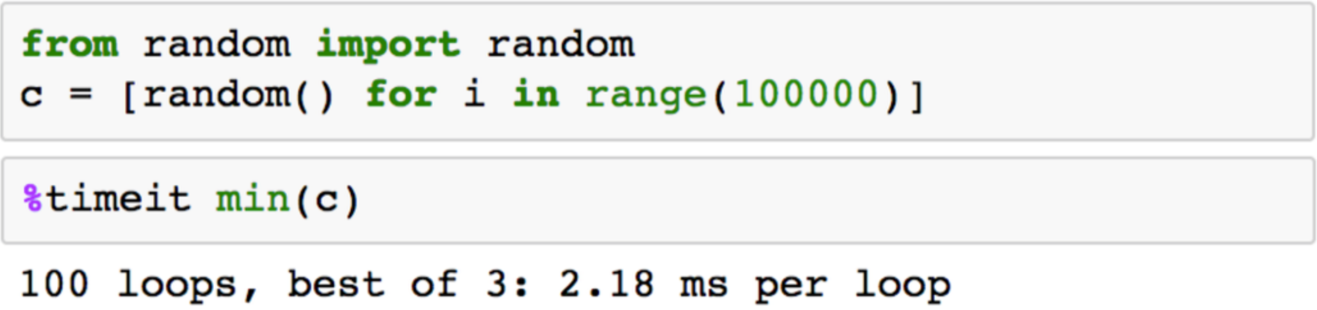
```
a = np.array(a)
%timeit a + 5
```

```
10000 loops, best of 3: 82.4 µs per loop
```

All numpy methods available here: http://docs.scipy.org/doc/numpy/genindex.html

NUMPY IS REALLY FAST PART 2

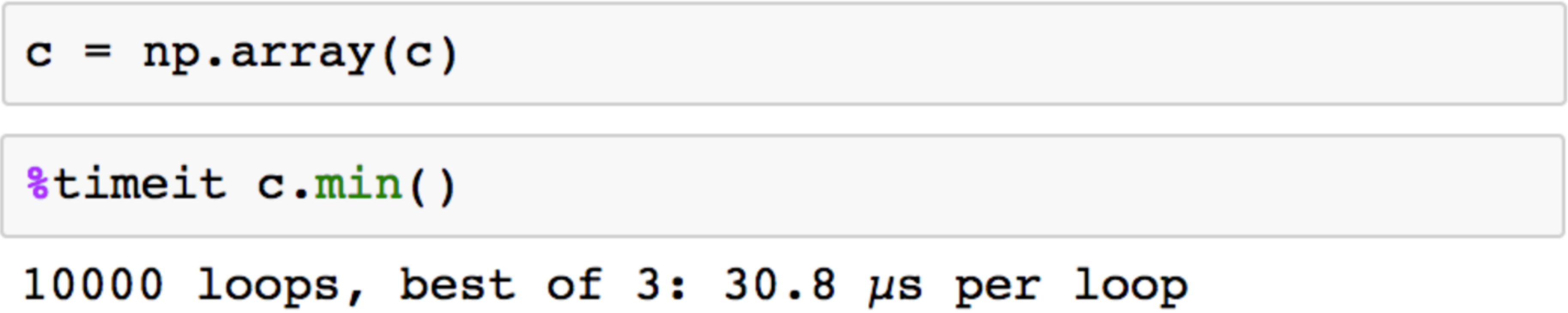
SAVINGS OF 70X!



```
from random import random
c = [random() for i in range(100000)]
```

```
%timeit min(c)
```

```
100 loops, best of 3: 2.18 ms per loop
```



```
c = np.array(c)
```

```
%timeit c.min()
```

```
10000 loops, best of 3: 30.8 µs per loop
```

All numpy methods available here: http://docs.scipy.org/doc/numpy/genindex.html

BY ITSELF VS SUPPORTING ANOTHER PACKAGE

- You'll use numpy by itself mostly for linear algebra work
- The majority of the time, numpy will be called by another package
- Avoid using numpy unless you really need it - other packages are easier to use
- Important for you to understand the basics!

All numpy methods available here: http://docs.scipy.org/doc/numpy/genindex.html

INTRODUCING PANDAS

PANDAS LOWDOWN

A LIBRARY FOR DATA ANALYSIS BASED ON THE DATAFRAME FROM R

- Very fast for datasets that can fit in memory
- A database or other solution would be better for large datasets
- Optimized for easy I/O and data manipulation
- Based on the dataframe, which functions like a python dictionary

TRY IT OUT: DATAFRAME

THE FUNDAMENTAL DATA STRUCTURE: DATAFRAME

- Open your terminal, type ipython and copy the below:

```
import pandas as pd
data = pd.read_csv('http://www.ats.ucla.edu/stat/data/binary.csv', sep=',')
data? (press q to quit)
data.head()
```

All numpy methods available here: http://docs.scipy.org/doc/numpy/genindex.html

WHEN TO USE PANDAS

SMALL TO MEDIUM DATA THAT FITS IN MEMORY

- You expect to do a ton of data munging
- Your dataset is small to medium sized (fits in memory of a single computer)
- If your dataset is huge, you should use a database or big data method

INTRODUCING MATPLOTLIB

A LIBRARY FOR DATA VISUALIZATION

- Usually the fastest option available in python
- Tons of capabilities but not intuitive to use
- Many libraries built on top, but they have a speed penalty
- Good for prototyping, but consider the strengths and weaknesses

SECRET: USE THE GALLERY LUKE!

A LIBRARY FOR DATA VISUALIZATION

http://matplotlib.org/gallery.html

WHEN TO USE MATPLOTLIB

DATA MUNGING OR QUICK PROCESSING

- Very quick and easy if you are “just looking”
- Easy to programmatically create many visuals
- Much more time-consuming if you need to produce client-ready visuals
- Most likely will need to be passed off to a designer
- Don’t underestimate the importance of visuals!

CHECK GITHUB FOR HW