大數據分析
(Big Data Analysis)

Python 大數據分析基礎
(Foundations of Big Data Analysis in Python)

Min-Yuh Day
Associate Professor
Institute of Information Management, National Taipei University

1091BDA03
MBA, IM, NTPU (M5127) (Fall 2020)
Wed 7, 8, 9 (15:10-18:00) (B8F40)

https://web.ntpu.edu.tw/~myday
2020-09-30
<table>
<thead>
<tr>
<th>週次 (Week)</th>
<th>日期 (Date)</th>
<th>內容 (Subject/Topics)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2020/09/16</td>
<td>大數據分析介紹 (Introduction to Big Data Analysis)</td>
</tr>
<tr>
<td>2</td>
<td>2020/09/23</td>
<td>AI人工智慧與大數據分析 (AI and Big Data Analysis)</td>
</tr>
<tr>
<td>3</td>
<td>2020/09/30</td>
<td>Python 大數據分析基礎 (Foundations of Big Data Analysis in Python)</td>
</tr>
<tr>
<td>4</td>
<td>2020/10/07</td>
<td>數位沙盒第一堂課：數位沙盒服務平台簡介 (Digital Sandbox Lesson 1: Introduction to FintechSpace Digital Sandbox)</td>
</tr>
<tr>
<td>5</td>
<td>2020/10/14</td>
<td>數位沙盒第二堂課：工程師操作說明與實作教學 (Digital Sandbox Lesson 2: Hands-on Practices)</td>
</tr>
<tr>
<td>6</td>
<td>2020/10/21</td>
<td>Python Pandas 大數據量化分析 (Quantitative Big Data Analysis with Pandas in Python)</td>
</tr>
</tbody>
</table>
週次 (Week)  日期 (Date)  內容 (Subject/Topics)
7  2020/10/28  數位沙盒第三堂課：學生小組討論實作與成果發表  
               (Digital Sandbox Lesson 3: Learning Teams  
                Hands-on Project Discussion and Project Presentation)
8  2020/11/04  Python Scikit-Learn 機器學習 Ⅰ  
                (Machine Learning with Scikit-Learn In Python Ⅰ)
9  2020/11/11  期中報告 (Midterm Project Report)
10 2020/11/18  Python Scikit-Learn 機器學習 Ⅱ  
               (Machine Learning with Scikit-Learn In Python Ⅱ)
11 2020/11/25  TensorFlow 深度學習金融大數據分析 Ⅰ  
               (Deep Learning for Finance Big Data Analysis with TensorFlow Ⅰ)
12 2020/12/02  大數據分析個案研究  
               (Case Study on Big Data Analysis)
課程大綱 (Syllabus)

週次 (Week)  日期 (Date)  內容 (Subject/Topics)

13  2020/12/09  TensorFlow 深度學習金融大數據分析 II  
      (Deep Learning for Finance Big Data Analysis with TensorFlow II)

14  2020/12/16  TensorFlow 深度學習金融大數據分析 III  
      (Deep Learning for Finance Big Data Analysis with TensorFlow III)

15  2020/12/23  AI 機器人理財顧問  
      (Artificial Intelligence for Robo-Advisors)

16  2020/12/30  金融科技智慧型交談機器人  
      (Conversational Commerce and Intelligent Chatbots for Fintech)

17  2021/01/06  期末報告 I (Final Project Report I)

18  2021/01/13  期末報告 II (Final Project Report I)
Foundations of Big Data Analysis in Python
Outline

• Foundations of Big Data Analysis in Python
  – Python
    • Programming language
  – Numpy
    • Scientific computing
The Quant Finance PyData Stack

Quantopian

Pytables

StatsModels

NetworkX

scikits-image

Pandas

Matplotlib

Stats in Python

SciPy

NumPy

Python

IPython

IPython

PyAlgoTrade

Zipline

DX Analytics

QuantLib

Jupyter
Python
Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.

Source: https://www.python.org/doc/essays/blurb/
Google Colab

Welcome to Colaboratory!

Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. See our FAQ for more info.

Getting Started

- Overview of Colaboratory
- Loading and saving data: Local files, Drive, Sheets, Google Cloud Storage
- Importing libraries and installing dependencies
- Using Google Cloud BigQuery
- Forms, Charts, Markdown, & Widgets
- TensorFlow with GPU
- Machine Learning Crash Course: Intro to Pandas & First Steps with TensorFlow

Highlighted Features

Seedbank

Looking for Colab notebooks to learn from? Check out Seedbank, a place to discover interactive machine learning examples.

TensorFlow execution

Colaboratory allows you to execute TensorFlow code in your browser with a single click. The example below adds two matrices.

\[
\begin{bmatrix}
1 & 1 & 1
\end{bmatrix} +
\begin{bmatrix}
1 & 2 & 3
\end{bmatrix} =
\begin{bmatrix}
2 & 3 & 4
\end{bmatrix}
\]

https://colab.research.google.com/notebooks/welcome.ipynb
Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

```python
# Future Value
pv = 100
r = 0.1
n = 7
fv = pv * ((1 + (r)) ** n)
print(round(fv, 2))
```

```python
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7

future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```python
# Python Function
def getfv(pv, r, n):
    fv = pv * ((1 + (r)) ** n)
    return fv

fv = getfv(100, 0.1, 7)
print(round(fv, 2))
```

```python
# Python if else
score = 80
if score >= 60 :
    print("Pass")
else:
    print("Fail").
```

https://tinyurl.com/aintpuppython101
Numpy

NumPy
Base
N-dimensional array package
Python Pandas

$p_{it} = \beta^{'}x_{it} + \mu_i + \epsilon_{it}$

http://pandas.pydata.org/
Iris flower data set

setosa  versicolor  virginica

Source: https://en.wikipedia.org/wiki/Iris_flower_data_set
Iris Classification

<table>
<thead>
<tr>
<th>Sepal Length</th>
<th>Sepal Width</th>
<th>Petal Length</th>
<th>Petal Width</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>3.5</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.9</td>
<td>3.0</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.7</td>
<td>3.2</td>
<td>1.3</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>3.1</td>
<td>1.5</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.6</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.4</td>
<td>3.9</td>
<td>1.7</td>
<td>0.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>3.4</td>
<td>1.4</td>
<td>0.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.4</td>
<td>1.5</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.4</td>
<td>2.9</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.9</td>
<td>3.1</td>
<td>1.5</td>
<td>0.1</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.4</td>
<td>3.7</td>
<td>1.5</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.8</td>
<td>3.4</td>
<td>1.6</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.8</td>
<td>3.0</td>
<td>1.4</td>
<td>0.1</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.3</td>
<td>3.0</td>
<td>1.1</td>
<td>0.1</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.8</td>
<td>4.0</td>
<td>1.2</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.7</td>
<td>4.4</td>
<td>1.5</td>
<td>0.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.4</td>
<td>3.9</td>
<td>1.3</td>
<td>0.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.1</td>
<td>3.5</td>
<td>1.4</td>
<td>0.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.7</td>
<td>3.8</td>
<td>1.7</td>
<td>0.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.1</td>
<td>3.8</td>
<td>1.5</td>
<td>0.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.4</td>
<td>3.4</td>
<td>1.7</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.1</td>
<td>3.7</td>
<td>1.5</td>
<td>0.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>3.6</td>
<td>1.0</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.1</td>
<td>3.3</td>
<td>1.7</td>
<td>0.5</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.8</td>
<td>3.4</td>
<td>1.9</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.0</td>
<td>1.6</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.4</td>
<td>1.6</td>
<td>0.4</td>
<td>Iris-setosa</td>
</tr>
</tbody>
</table>
Iris Data Visualization

Source: https://seaborn.pydata.org/generated/seaborn.pairplot.html
Google Colab
Google Colab
Connect Colaboratory to Google Drive

Colaboratory was connected to Google Drive.
Make Colaboratory the default app for files it can open

OK
Google Colab
Google Colab

![Image of Google Colab interface with highlighted menu options]

- **Run all** (F9)
- **Run before** (F8)
- **Run the focused cell** (Enter)
- **Run selection** (Ctrl+Shift+Enter)
- **Run after** (F10)
- **Interrupt execution** (Ctrl+M)
- **Restart runtime** (M)
- **Restart and run all** (M)
- **Reset all runtimes**
- **Change runtime type**

Additional options:
- **Manage sessions**
Run Jupyter Notebook
Python3 GPU
Google Colab
print('Hello World')
Data Visualization in Google Colab

https://colab.research.google.com/drive/1KRqtEUd2Hg4dM2au9bfVQKrxFnWN3O9-

```
import seaborn as sns
sns.set(style="ticks", color_codes=True)
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species").
```
import seaborn as sns
sns.set(style="ticks", color_codes=True)
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species")
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import import scatter_matrix

# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)

print(df.head(10))
print(df.tail(10))
print(df.describe())
print(df.info())
print(df.shape)
print(df.groupby('class').size())

plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()

df.hist()
plt.show()

scatter_matrix(df)
plt.show()
sns.pairplot(df, hue="class", size=2)
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix

imported
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)
print(df.head(10))

# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)
print(df.head(10)).

<table>
<thead>
<tr>
<th>sepal-length</th>
<th>sepal-width</th>
<th>petal-length</th>
<th>petal-width</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>3.5</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.9</td>
<td>3.0</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.7</td>
<td>3.2</td>
<td>1.3</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>3.1</td>
<td>1.5</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.6</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.4</td>
<td>3.9</td>
<td>1.7</td>
<td>0.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>3.4</td>
<td>1.4</td>
<td>0.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.4</td>
<td>1.5</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.4</td>
<td>2.9</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.9</td>
<td>3.1</td>
<td>1.5</td>
<td>0.1</td>
<td>Iris-setosa</td>
</tr>
</tbody>
</table>
```python
print(df.tail(10)).
```

<table>
<thead>
<tr>
<th>sepal-length</th>
<th>sepal-width</th>
<th>petal-length</th>
<th>petal-width</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>140</td>
<td>6.7</td>
<td>3.1</td>
<td>5.6</td>
<td>2.4 Iris-virginica</td>
</tr>
<tr>
<td>141</td>
<td>6.9</td>
<td>3.1</td>
<td>5.1</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>142</td>
<td>5.8</td>
<td>2.7</td>
<td>5.1</td>
<td>1.9 Iris-virginica</td>
</tr>
<tr>
<td>143</td>
<td>6.8</td>
<td>3.2</td>
<td>5.9</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>144</td>
<td>6.7</td>
<td>3.3</td>
<td>5.7</td>
<td>2.5 Iris-virginica</td>
</tr>
<tr>
<td>145</td>
<td>6.7</td>
<td>3.0</td>
<td>5.2</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>146</td>
<td>6.3</td>
<td>2.5</td>
<td>5.0</td>
<td>1.9 Iris-virginica</td>
</tr>
<tr>
<td>147</td>
<td>6.5</td>
<td>3.0</td>
<td>5.2</td>
<td>2.0 Iris-virginica</td>
</tr>
<tr>
<td>148</td>
<td>6.2</td>
<td>3.4</td>
<td>5.4</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>149</td>
<td>5.9</td>
<td>3.0</td>
<td>5.1</td>
<td>1.8 Iris-virginica</td>
</tr>
</tbody>
</table>
```python
print(df.describe())
```

<table>
<thead>
<tr>
<th></th>
<th>sepal-length</th>
<th>sepal-width</th>
<th>petal-length</th>
<th>petal-width</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>150.000000</td>
<td>150.000000</td>
<td>150.000000</td>
<td>150.000000</td>
</tr>
<tr>
<td>mean</td>
<td>5.843333</td>
<td>3.054000</td>
<td>3.758667</td>
<td>1.198667</td>
</tr>
<tr>
<td>std</td>
<td>0.828066</td>
<td>0.433594</td>
<td>1.764420</td>
<td>0.763161</td>
</tr>
<tr>
<td>min</td>
<td>4.300000</td>
<td>2.000000</td>
<td>1.000000</td>
<td>0.100000</td>
</tr>
<tr>
<td>25%</td>
<td>5.100000</td>
<td>2.800000</td>
<td>1.600000</td>
<td>0.300000</td>
</tr>
<tr>
<td>50%</td>
<td>5.800000</td>
<td>3.000000</td>
<td>4.350000</td>
<td>1.300000</td>
</tr>
<tr>
<td>75%</td>
<td>6.400000</td>
<td>3.300000</td>
<td>5.100000</td>
<td>1.800000</td>
</tr>
<tr>
<td>max</td>
<td>7.900000</td>
<td>4.400000</td>
<td>6.900000</td>
<td>2.500000</td>
</tr>
</tbody>
</table>
print(df.info())
print(df.shape)

print(df.info())
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepal-length  150 non-null float64
sepal-width   150 non-null float64
petal-length  150 non-null float64
petal-width   150 non-null float64
class         150 non-null object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
None

print(df.shape)

(150, 5)
df.groupby('class').size()

```python
print(df.groupby('class').size())
```

<table>
<thead>
<tr>
<th>class</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris-setosa</td>
<td>50</td>
</tr>
<tr>
<td>Iris-versicolor</td>
<td>50</td>
</tr>
<tr>
<td>Iris-virginica</td>
<td>50</td>
</tr>
</tbody>
</table>

dtype: int64
plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()
df.hist()
plt.show()
scatter_matrix(df)
plt.show()
sns.pairplot(df, hue="class", size=2)
Anaconda
The Most Popular Python Data Science Platform

Source: https://www.anaconda.com/
Download Anaconda

Don't Miss AnacondaCon Apr 8-11 Austin TX!

Download Anaconda Distribution

Version 5.1 | Release Date: February 15, 2018

Download For: 🌐 🖥️ 🍎 🦠

High-Performance Distribution
Easily install 1,000+ data science packages

Package Management
Manage packages, dependencies and environments with conda

Portal to Data Science
Uncover insights in your data and create interactive visualizations

Anaconda 5.1 For macOS Installer

https://www.anaconda.com/download
Python

HelloWorld
Anaconda-Navigator
Jupyter Notebook
Jupyter Notebook
New Python 3
print("hello, world")

In [1]:
   print("hello, world")

   hello, world

In [ ]:
from platform import python_version
print("Python Version:", python_version())

In [1]:
print("hello, world")

hello, world

In [2]:
from platform import python_version
print("Python Version:", python_version())

Python Version: 3.6.5

In [ ]:
Python Programming
print("Hello Python Fiddle")

http://pythonfiddle.com/
print("Hello World")

print("Hello World\nThis is a message")

x = 3
print(x)

x = 2
y = 3
print(x, ' ', y)

name = input("Enter a name: ")

x = int(input("What is x? "))

x = float(input("Write a number "))
Python in Google Colab

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

https://tinyurl.com/aintpuppython101
Text input and output

```
In [1]: print("Hello World")
Hello World

In [2]: print("Hello World\nThis is a message")
Hello World
This is a message

In [3]: x = 3
print(x)
3

In [4]: x = 2
y = 3
print(x, ', ', y)
2 3

In [5]: name = input("Enter a name: ")
Enter a name: Myday

In [6]: x = int(input("What is x? "))
What is x? 80

In [7]: x = float(input("Write a number "))
Write a number 3.6
```
Variables

```
x = 2
price = 2.5
word = 'Hello'

word = 'Hello'
word = "Hello"
word = '''Hello'''

x = 2
x = x + 1
x = 5
```

Source: [http://pythonprogramminglanguage.com/](http://pythonprogramminglanguage.com/)
Python Basic Operators

```python
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

```
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

```
7 + 2 = 9
7 - 2 = 5
7 * 2 = 14
7 / 2 = 3.5
7 // 2 = 3
7 % 2 = 1
7 ** 2 = 49
```
BMI Calculator in Python

```python
height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

height_m = height_cm/100
BMI = (weight_kg/(height_m**2))
print("Your BMI is: "+str(round(BMI,1)))
```

Source: http://code.activestate.com/recipes/580615-bmi-code/
BMI Calculator in Python

```python
In [1]:
height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

# Convert cm to m
height_m = height_cm/100
BMI = (weight_kg/(height_m**2))

print("Your BMI is: " + str(round(BMI,1)))

Enter your height in cm: 170
Enter your weight in kg: 60
Your BMI is: 20.8
```
Future value of a specified principal amount, rate of interest, and a number of years.

Source: https://www.w3resource.com/python-exercises/python-basic-exercise-39.php
Future Value (FV)

# How much is your $100 worth after 7 years?

```
print(100 * 1.1 ** 7)
# output = 194.87
```
Future Value (FV)

\[ \text{fv} = \text{pv} \times ((1 + (r))^n) \]

```python
pv = 100
r = 0.1
n = 7
fv = pv * ((1 + (r)) ** n)
print(round(fv, 2))
```

194.87
Future Value (FV)

```python
amount = 100
interest = 10  #10% = 0.01 * 10
years = 7

future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```
194.87
```

if statements

> greater than
< smaller than
== equals
!= is not

score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```python
def assign_grade(score):
    grade = ""
    if score >= 90:
        grade = "A"
    elif score >= 80:
        grade = "B"
    elif score >= 70:
        grade = "C"
    elif score >= 60:
        grade = "D"
    else:
        grade = "E"
    print(grade)
    # grade = "A"
```

Source: [http://pythonprogramminglanguage.com/](http://pythonprogramminglanguage.com/)

http://pythontutor.com/visualize.html
https://goo.gl/E6w5ph
for loops

for i in range(1,11):
    print(i)
for loops

```python
for i in range(1,10):
    for j in range(1,10):
        print(i, ' * ' , j , ' = ', i*j)
```

9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81

Source: [http://pythonprogramminglanguage.com/](http://pythonprogramminglanguage.com/)
while loops

age = 10

while age < 20:
    print(age)
    age = age + 1

10
11
12
13
14
15
16
17
18
19

Source: https://learnpython.trinket.io/learn-python-part-8-loops/#/while-loops/about-while-loops
```python
def convertCMtoM(xcm):
    m = xcm/100
    return m

cm = 180
m = convertCMtoM(cm)
print(str(m))
```

1.8
Lists

```python
x = [60, 70, 80, 90]
print(len(x))
print(x[0])
print(x[1])
print(x[-1])
```

```
4
60
70
90
```
A tuple in Python is a collection that cannot be modified. A tuple is defined using parenthesis.

\[
x = (10, 20, 30, 40, 50)
\]

print(x[0])
print(x[1])
print(x[2])
print(x[[-1]])

Source: http://pythonprogramminglanguage.com/tuples/
```python
k = { 'EN':'English', 'FR':'French' }
print(k['EN'])
```

**Dictionary**

```
'EN'    →    'English'
'FR'    →    'French'
```

**English**

Sets

animals = {'cat', 'dog'}

animals = {'cat', 'dog'}
print('cat' in animals)  # Check if an element is in a set; prints "True"
print('fish' in animals)  # prints "False"
animals.add('fish')      # Add an element to a set
print('fish' in animals)  # Prints "True"
print(len(animals))      # Number of elements in a set; prints "3"
animals.add('cat')       # Adding an element that is already in the set does nothing
print(len(animals))      # Prints "3"
animals.remove('cat')    # Remove an element from a set
print(len(animals))      # Prints "2"

True
False
True
3
3
2

Source: http://cs231n.github.io/python-numpy-tutorial/
with open('myfile.txt', 'w') as file:
    file.write('Hello World\nThis is Python File Input Output')

with open('myfile.txt', 'r') as file:
    text = file.read()
print(text)

Hello World
This is Python File Input Output

text

'Hello World\nThis is Python File Input Output'
with open('myfile.txt', 'a+') as file:
    file.write('
' + 'New line')

with open('myfile.txt', 'r') as file:
    text = file.read()
print(text)
Big Data Analytics with Numpy in Python
Numpy

NumPy
Base
N-dimensional array package
NumPy is the fundamental package for scientific computing with Python.

Source: http://www.numpy.org/
NumPy

• **NumPy** provides a **multidimensional array** object to store homogenous or heterogeneous data; it also provides **optimized functions/methods** to operate on this array object.

Source: Yves Hilpisch (2014), Python for Finance: Analyze Big Financial Data, O'Reilly
NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

NumPy is licensed under the BSD license, enabling reuse with few restrictions.

Getting Started

- Getting NumPy
- Installing the SciPy Stack
- NumPy and SciPy documentation page
- NumPy Tutorial
- NumPy for MATLAB© Users
- NumPy functions by category
- NumPy Mailing List

For more information on the SciPy Stack (for which NumPy provides the fundamental array data structure), see scipy.org.

http://www.numpy.org/
NumPy ndarray

One-dimensional Array
(1-D Array)

\[
\begin{array}{cccc}
0 & 1 & 2 & 3 & 4 & 5 \\
1 & 2 & 3 & 4 & 5 & 6 \\
m-1 & 16 & 17 & 18 & 19 & 20 \\
\end{array}
\]

Two-dimensional Array
(2-D Array)
v = list(range(1, 6))
v
2 * v

import numpy as np
v = np.arange(1, 6)
v
2 * v
```python
1 v = list(range(1, 6))
2 v
[1, 2, 3, 4, 5]

1 2 * v

[1, 2, 3, 4, 5, 1, 2, 3, 4, 5]

1 import numpy as np
2 v = np.arange(1, 6)
3 v
array([[1, 2, 3, 4, 5]])

1 2 * v

array([[ 2,  4,  6,  8, 10]])
```
```
import numpy as np
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
c = a * b
c
```

```
array([ 4, 10, 18])
```
```python
import numpy as np

a = np.zeros((2,2))  # Create an array of all zeros
print(a)  # Prints "[[ 0.  0.]
# [ 0.  0.]]"

b = np.ones((1,2))  # Create an array of all ones
print(b)  # Prints "[[ 1.  1.]]"

c = np.full((2,2), 7)  # Create a constant array
print(c)  # Prints "[[ 7.  7.]
# [ 7.  7.]]"

d = np.eye(2)  # Create a 2x2 identity matrix
print(d)  # Prints "[[ 1.  0.]
# [ 0.  1.]]"

e = np.random.random((2,2))  # Create an array filled with random values
print(e)  # Might print "[[ 0.91940167 0.08143941]
# [ 0.68744134 0.87236687]]"
```

Source: http://cs231n.github.io/python-numpy-tutorial/
Quickstart tutorial

Prerequisites

Before reading this tutorial you should know a bit of Python. If you would like to refresh your memory, take a look at the Python tutorial.

If you wish to work the examples in this tutorial, you must also have some software installed on your computer. Please see http://scipy.org/install.html for instructions.

The Basics

NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called axes. The number of axes is rank.

For example, the coordinates of a point in 3D space \([1, 2, 1]\) is an array of rank 1, because it has one axis. That axis has a length of 3. In the example pictured below, the array has rank 2 (it is 2-dimensional). The first dimension (axis) has a length of 2, the second dimension has a length of 3.

```
[[ 1., 0., 0.],
 [ 0., 1., 2.]]
```

NumPy's array class is called ndarray. It is also known by the alias array. Note that numpy.array is not the same as the Standard Python Library class array.array, which only handles one-dimensional arrays and offers less functionality. The more important attributes of an ndarray object are:

- ndarray.ndim: the number of axes (dimensions) of the array. In the Python world, the number of dimensions is referred to as rank.
- ndarray.shape:
import numpy as np
a = np.arange(15).reshape(3, 5)
a.shape
a.ndim
a.dtype.name

import numpy as np
a = np.arange(15).reshape(3, 5)
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])

print(a.shape)
(3, 5)
a.ndim
2
a.dtype.name
'int64'

Source: https://docs.scipy.org/doc/numpy-dev/user/quickstart.html
Matrix

$m$-by-$n$ matrix

$a_{i,j}$

$m$ rows

$n$ columns

$i$ changes

$j$ changes

\[
\begin{bmatrix}
  a_{1,1} & a_{1,2} & a_{1,3} & \ldots \\
  a_{2,1} & a_{2,2} & a_{2,3} & \ldots \\
  a_{3,1} & a_{3,2} & a_{3,3} & \ldots \\
  \vdots & \vdots & \vdots & \ddots \\
  \vdots & \vdots & \vdots & \ddots \\
\end{bmatrix}
\]

Source: https://simple.wikipedia.org/wiki/Matrix_(mathematics)
NumPy ndarray:
Multidimensional Array Object
**NumPy ndarray**

### One-dimensional Array (1-D Array)

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

### Two-dimensional Array (2-D Array)

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>m-1</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
</tbody>
</table>
import numpy as np

a = np.array([1,2,3,4,5])

One-dimensional Array
(1-D Array)

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

a = np.array([1,2,3,4,5])
a
array([1, 2, 3, 4, 5])
Two-dimensional Array
(2-D Array)

\[
a = \text{np.array}([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15],[16,17,18,19,20]])
\]

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>m-1</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>
```python
import numpy as np

a = np.array([[0, 1, 2, 3],
              [10, 11, 12, 13],
              [20, 21, 22, 23]])

a
```

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
</tr>
</tbody>
</table>
```python
import numpy as np

a = np.array([[0, 1, 2, 3], [10, 11, 12, 13], [20, 21, 22, 23]])

print(a)

array([[ 0,  1,  2,  3],
       [10, 11, 12, 13],
       [20, 21, 22, 23]])

print(a.ndim)

2

print(a.shape)

(3, 4)
```

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
</tr>
</tbody>
</table>
NumPy Basics: Arrays and Vectorized Computation

## NumPy Array

![NumPy Array Diagram](https://www.safaribooksonline.com/library/view/python-for-data/9781449323592/ch04.html)

### Axis 0
- 0
  - 0,0
  - 1,0
  - 2,0
- 1
  - 0,1
  - 1,1
  - 2,1
- 2
  - 0,2
  - 1,2
  - 2,2

### Axis 1
- 0
  - 0,0
  - 0,1
  - 0,2
- 1
  - 1,0
  - 1,1
  - 1,2
- 2
  - 2,0
  - 2,1
  - 2,2

# Numpy Array

<table>
<thead>
<tr>
<th>Expression</th>
<th>Shape</th>
</tr>
</thead>
<tbody>
<tr>
<td>arr[:2, 1:]</td>
<td>(2, 2)</td>
</tr>
<tr>
<td>arr[2]</td>
<td>(3,)</td>
</tr>
<tr>
<td>arr[2, :]</td>
<td>(3,)</td>
</tr>
<tr>
<td>arr[2:, :]</td>
<td>(1, 3)</td>
</tr>
<tr>
<td>arr[:, :2]</td>
<td>(3, 2)</td>
</tr>
<tr>
<td>arr[1, :2]</td>
<td>(2,)</td>
</tr>
<tr>
<td>arr[1:2, :2]</td>
<td>(1, 2)</td>
</tr>
</tbody>
</table>


Materials and IPython notebooks for "Python for Data Analysis" by Wes McKinney, published by O'Reilly Media

<table>
<thead>
<tr>
<th>Branch: 2nd-edition</th>
<th>New pull request</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>Last Commit Date</th>
<th>Last Update Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>datasets</td>
<td>Add Kaggle titanic dataset</td>
<td></td>
<td>5 months ago</td>
</tr>
<tr>
<td>examples</td>
<td>Remove sex column from tips dataset</td>
<td></td>
<td>4 months ago</td>
</tr>
<tr>
<td>.gitignore</td>
<td>Add gitignore</td>
<td></td>
<td>2 years ago</td>
</tr>
<tr>
<td>COPYING</td>
<td>Use MIT license for code examples</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>README.md</td>
<td>Add launch in Azure Notebooks button (#70)</td>
<td></td>
<td>19 days ago</td>
</tr>
<tr>
<td>appa.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch02.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch03.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch04.ipynb</td>
<td>Convert all notebooks to v4 format</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch05.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch06.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch07.ipynb</td>
<td>Convert all notebooks to v4 format</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch08.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch09.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
<tr>
<td>ch10.ipynb</td>
<td>Make more cells markdown instead of raw</td>
<td></td>
<td>a month ago</td>
</tr>
</tbody>
</table>

https://github.com/wesm/pydata-book
NumPy Basics: Arrays and

In [ ]:
```python
import numpy as np
np.random.seed(12345)
import matplotlib.pyplot as plt
plt.rc('figure', figsize=(10, 5))
np.set_printoptions(precision=4, suppress=True)
```

In [ ]:
```python
import numpy as np
my_arr = np.arange(1000000)
my_list = list(range(1000000))
```

In [ ]:
```python
%time for _ in range(10); my_arr2 = my_arr * 2
%time for _ in range(10); my_list2 = [x * 2 for x in my_list]
```

The NumPy ndarray: A Multidimensional Array Object

In [ ]:
```python
import numpy as np
# Generate some random data
data = np.random.randn(2, 3)
data
```
Summary

• Foundations of Big Data Analysis in Python
  – Python
    • Programming language
  – Numpy
    • Scientific computing
References


• Ties de Kok (2017), Learn Python for Research, https://github.com/TiesdeKok/LearnPythonforResearch


• Python Programming, https://pythonprogramming.net/

• Python, https://www.python.org/

• Python Programming Language, http://pythonprogramminglanguage.com/

• Numpy, http://www.numpy.org/

• Pandas, http://pandas.pydata.org/

• Skikit-learn, http://scikit-learn.org/

• Data School (2015), Machine learning in Python with scikit-learn, https://www.youtube.com/playlist?list=PL5-da3qGB5lCeMbQuqbbCOQWcS6OYBr5A
