

Section 08

NATURAL LANGUAGE TOOLKIT (NLTK)

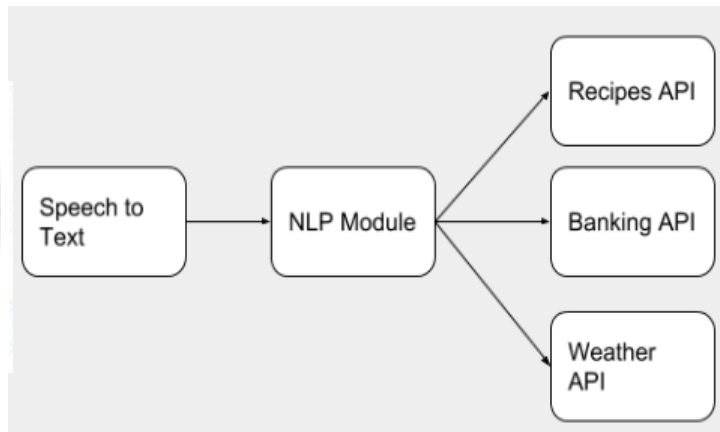
Cscie63 Big Data Analytics
Harvard Extension School

Section 08 Agenda

- NLTK overview
- Python/Anaconda install
- Install NLTK (load data, XXX)
- Jupyter Notebook navigation
- How to import Collections and load data into Python / Tables
- Text & List & string manipulations
- Setting up Data Frames using Pandas
- Parts of Speech
- Python Notebook demo

NLTK Examples

Example 1



Siri: *What is the weather and traffic conditions for my drive to work?*
Alexa: *Dim the lights and thermostat.*
Google Assistant...

Example 2

1. Tokenization

```
['What', 'is', 'the', 'weather', 'in', 'Chicago', '?']
```

2. Stop word removal

```
['What', 'weather', 'Chicago', '?']
```

3. Parts of Speech Tagging

```
[('What', 'WP'), ('weather', 'NN'), ('Chicago', 'NNP'), ('?', '.')]
```

4. Named Entity Recognition

```
>>> print(nltk.ne_chunk(tagged))  
(S What/WP weather/NN (GPE Chicago/NNP) ?/.)
```

Meaning of sentence can now be analyzed
'Chicago'=location & weather (can be associated with a weather service -> Call a Weather Web Service

Natural Language Toolkit

Online documentation: <http://www.nltk.org/>

From Zoran's slide:

- Used for unstructured text data to process data for classification, tokenization, stemming, tagging, parsing, and semantic reasoning
- Open Source package to Python with its own prepackaged data sets.
 - Very efficient for dealing with large text files & “functions for processing linguistic data”
- Excellent for extracting key words & phrases.

NLP applications

- Text Categorization
 - Classify documents by topics, language, author, spam filtering, information retrieval (relevant, not relevant), sentiment classification (positive, negative)
- Spelling & Grammar Corrections
- Information Extraction
- Speech Recognition
- Information Retrieval
 - Synonym Generation
- Summarization
- Machine Translation
- Question Answering
- Dialog Systems
 - Language generation

What is Anaconda?

[https://en.wikipedia.org/wiki/Anaconda_\(Python_distribution\)](https://en.wikipedia.org/wiki/Anaconda_(Python_distribution))

- ‘open source distribution of the **Python** and R **programming** languages for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment. Package versions are managed by the package management system conda’
- ‘Python and over 150 scientific packages automatically installed at once’

<https://www.anaconda.com/what-is-anaconda/>



My Environment Set up

My environment:

1. Windows 10 64-bit

View basic information about your computer	
Windows edition	
Windows 10 Home	
© 2017 Microsoft Corporation. All rights reserved.	
System	
Manufacturer:	Dell
Model:	Inspiron 5423
Processor:	Intel(R) Core(TM) i3-2367M CPU @ 1.40GHz 1.40 GHz
Installed memory (RAM):	6.00 GB (5.87 GB usable)
System type:	64-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

2. I have Anaconda 3.6.2 64-bit (latest installed). **2.7 will install similar to the following slides. It is fine to use 2.7 or 3.6.2.**

```
C:\Users\dhoward>python
Python 3.6.2 |Anaconda, Inc.| (default, Sep 19 2017, 08:03:39) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
```

3. My environment variables:

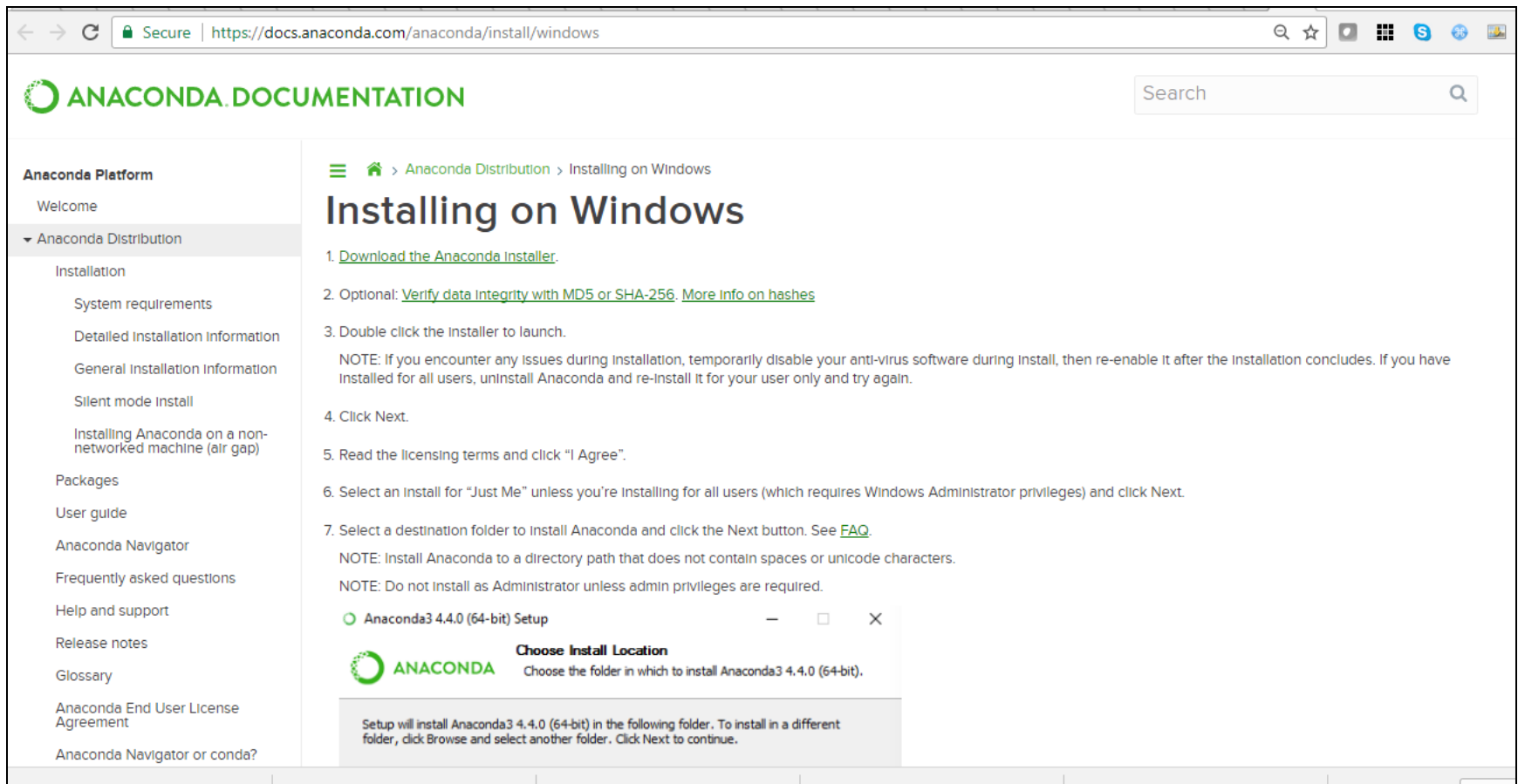
C:\Users\dhoward\anaconda3_64bit

c:\users\dhoward\anaconda3_64bit\scripts

```
C:\Users\dhoward\anaconda3_64bit
c:\users\dhoward\anaconda3_64bit\scripts
```

Anaconda Windows Installation Instructions

Note: You do not need to install as Administrator unless admin privileges are needed...



The screenshot shows a web browser displaying the Anaconda documentation page for Windows installation. The browser address bar shows the URL <https://docs.anaconda.com/anaconda/install/windows>. The page title is "Installing on Windows". The left sidebar contains a navigation menu with the following items: "Anaconda Platform", "Welcome", "Anaconda Distribution" (expanded), "Installation", "System requirements", "Detailed Installation Information", "General Installation Information", "Silent mode Install", "Installing Anaconda on a non-networked machine (air gap)", "Packages", "User guide", "Anaconda Navigator", "Frequently asked questions", "Help and support", "Release notes", "Glossary", "Anaconda End User License Agreement", and "Anaconda Navigator or conda?". The main content area lists the installation steps:

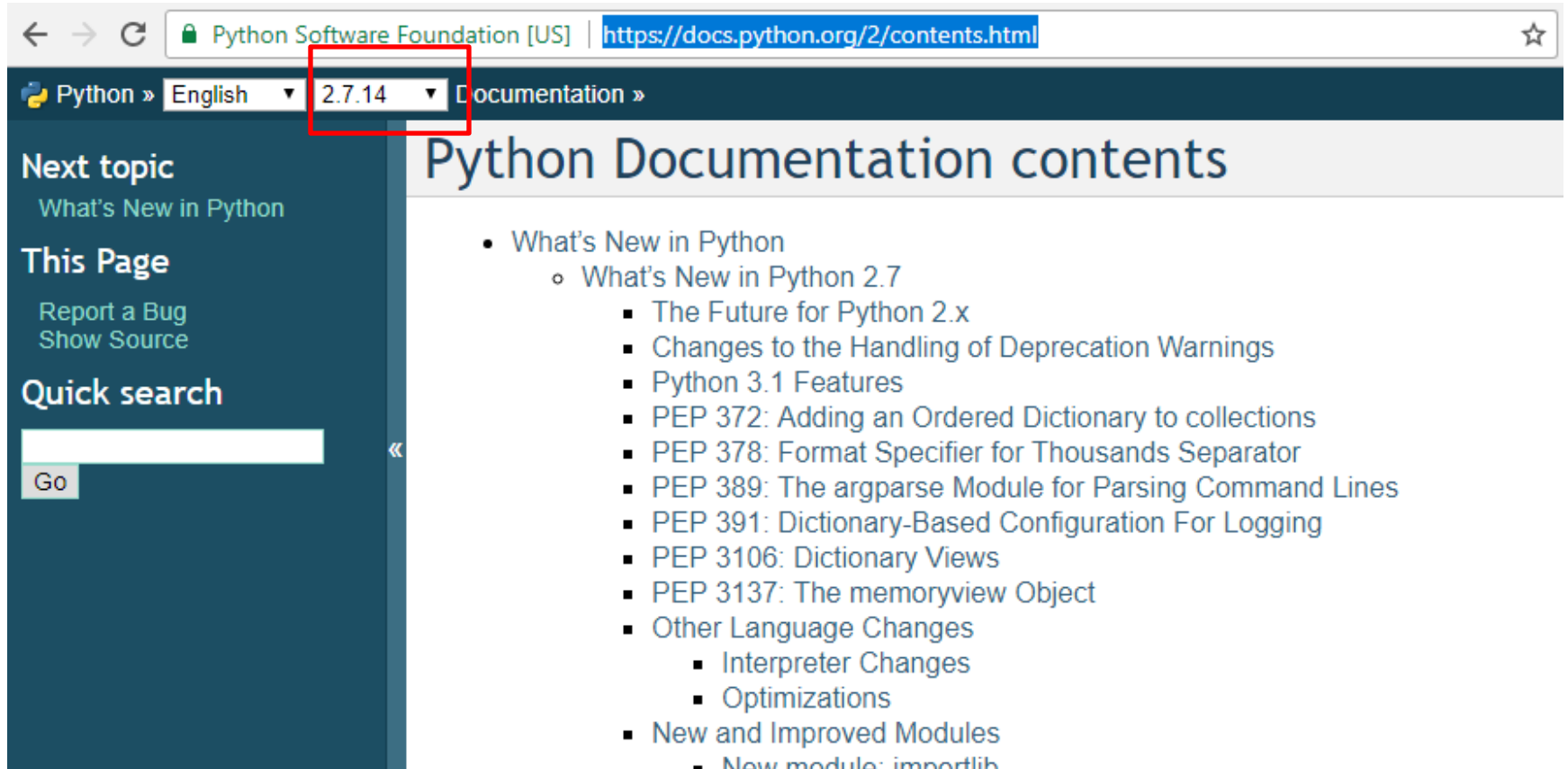
1. [Download the Anaconda Installer.](#)
2. Optional: [Verify data integrity with MD5 or SHA-256.](#) [More info on hashes](#)
3. Double click the installer to launch.
NOTE: If you encounter any issues during installation, temporarily disable your anti-virus software during install, then re-enable it after the installation concludes. If you have installed for all users, uninstall Anaconda and re-install it for your user only and try again.
4. Click Next.
5. Read the licensing terms and click "I Agree".
6. Select an install for "Just Me" unless you're installing for all users (which requires Windows Administrator privileges) and click Next.
7. Select a destination folder to install Anaconda and click the Next button. See [FAQ](#).

NOTE: Install Anaconda to a directory path that does not contain spaces or unicode characters.
NOTE: Do not install as Administrator unless admin privileges are required.

The screenshot also shows the "Anaconda3 4.4.0 (64-bit) Setup" window. The window title is "Anaconda3 4.4.0 (64-bit) Setup". The window content shows the "Choose Install Location" dialog. The dialog text says: "Choose the folder in which to install Anaconda3 4.4.0 (64-bit).". Below the text, it says: "Setup will install Anaconda3 4.4.0 (64-bit) in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue."

2.7 Python documentation

<https://docs.python.org/2/contents.html>

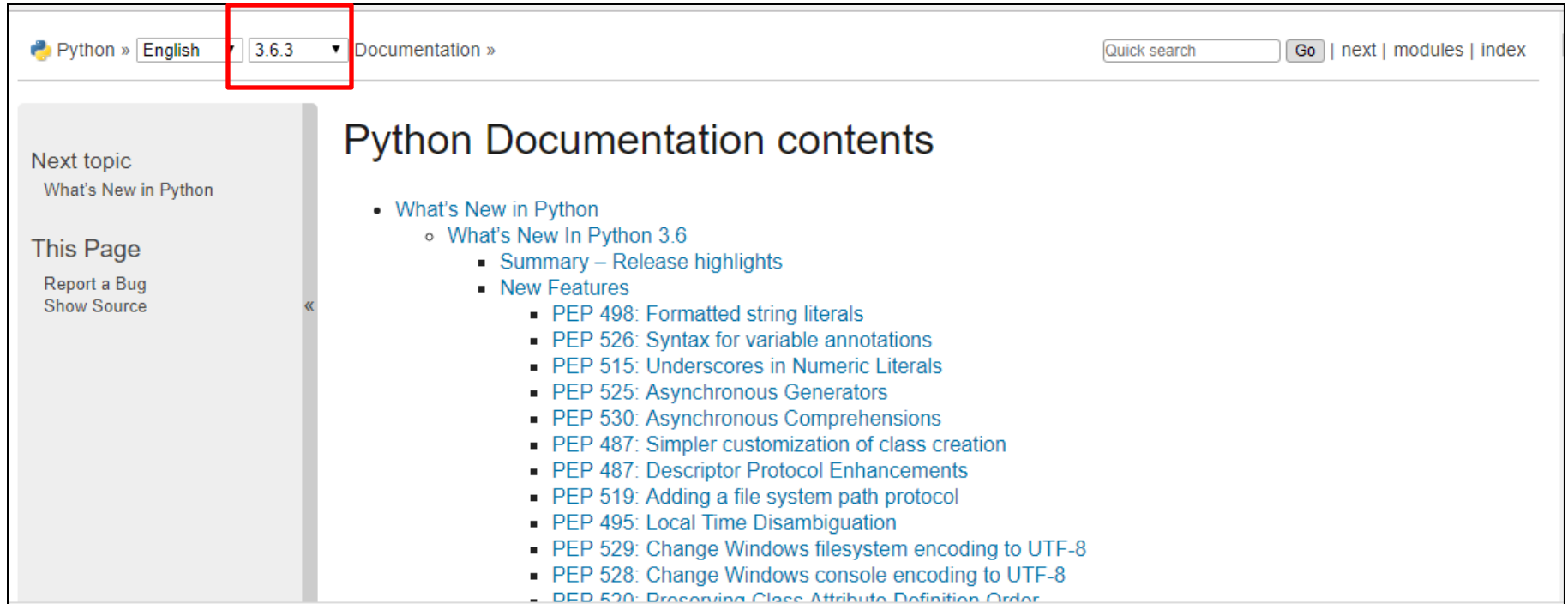


The screenshot shows a web browser window displaying the Python Software Foundation's documentation for Python 2.7.14. The browser's address bar shows the URL <https://docs.python.org/2/contents.html>. The page header includes a navigation bar with "Python »", a language dropdown set to "English", a version dropdown set to "2.7.14" (highlighted with a red box), and "Documentation »". The main heading is "Python Documentation contents". The left sidebar contains links for "Next topic" (What's New in Python), "This Page" (Report a Bug, Show Source), and a "Quick search" box with a "Go" button. The main content area lists the following topics:

- What's New in Python
 - What's New in Python 2.7
 - The Future for Python 2.x
 - Changes to the Handling of Deprecation Warnings
 - Python 3.1 Features
 - PEP 372: Adding an Ordered Dictionary to collections
 - PEP 378: Format Specifier for Thousands Separator
 - PEP 389: The argparse Module for Parsing Command Lines
 - PEP 391: Dictionary-Based Configuration For Logging
 - PEP 3106: Dictionary Views
 - PEP 3137: The memoryview Object
 - Other Language Changes
 - Interpreter Changes
 - Optimizations
 - New and Improved Modules
 - New module: importlib

3.0 Python documentation

- <https://docs.python.org/3/contents.html>



The screenshot shows the Python documentation website for version 3.6.3. The top navigation bar includes the Python logo, a language selector set to 'English', a version dropdown set to '3.6.3' (highlighted with a red box), and a 'Documentation »' link. To the right is a search bar labeled 'Quick search' with a 'Go' button and links for 'next', 'modules', and 'index'. The main content area is titled 'Python Documentation contents' and features a bulleted list of links. A left sidebar contains links for 'Next topic' (What's New in Python) and 'This Page' (Report a Bug, Show Source).

Python » English 3.6.3 Documentation »

Quick search Go | next | modules | index

Python Documentation contents

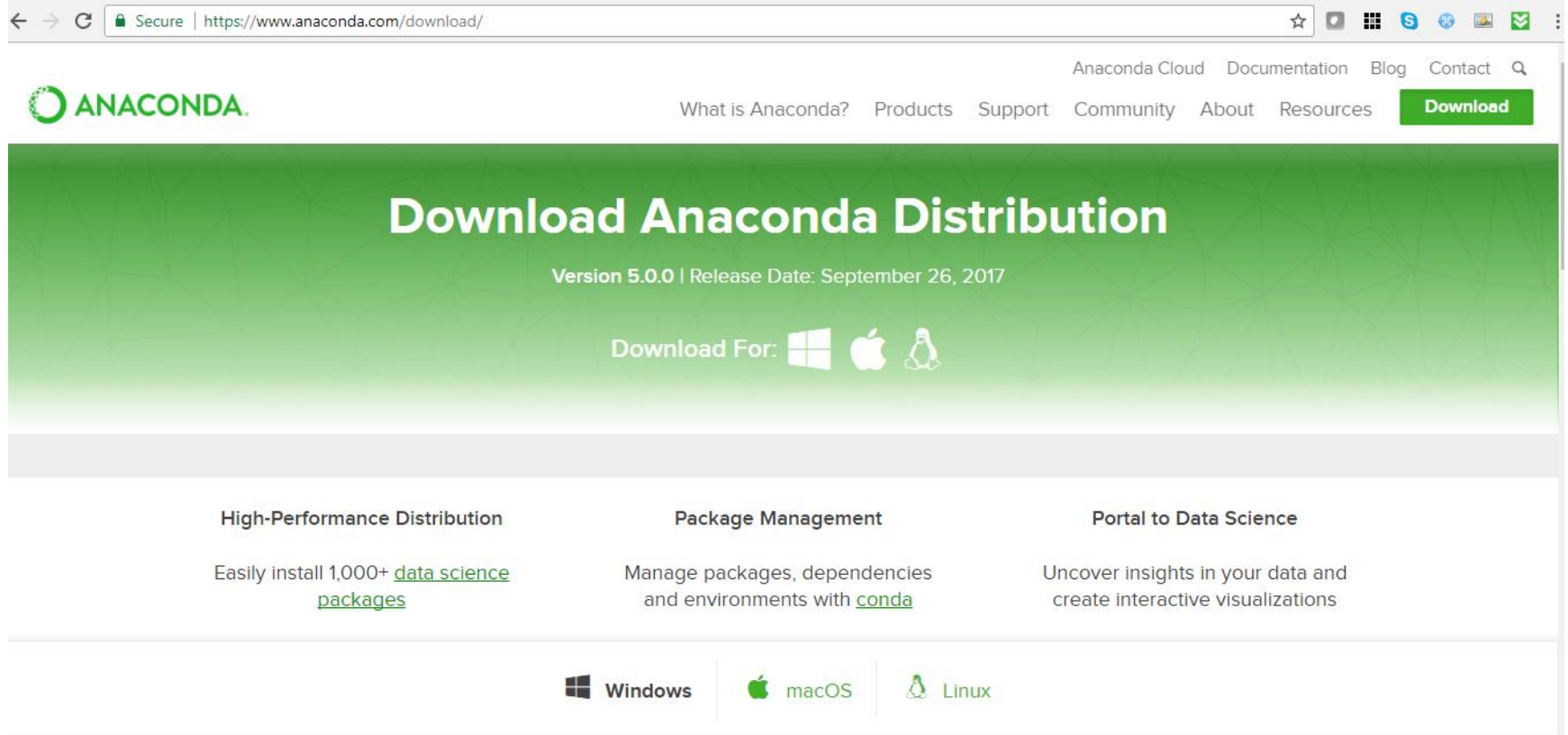
- [What's New in Python](#)
 - [What's New In Python 3.6](#)
 - [Summary – Release highlights](#)
 - [New Features](#)
 - [PEP 498: Formatted string literals](#)
 - [PEP 526: Syntax for variable annotations](#)
 - [PEP 515: Underscores in Numeric Literals](#)
 - [PEP 525: Asynchronous Generators](#)
 - [PEP 530: Asynchronous Comprehensions](#)
 - [PEP 487: Simpler customization of class creation](#)
 - [PEP 487: Descriptor Protocol Enhancements](#)
 - [PEP 519: Adding a file system path protocol](#)
 - [PEP 495: Local Time Disambiguation](#)
 - [PEP 529: Change Windows filesystem encoding to UTF-8](#)
 - [PEP 528: Change Windows console encoding to UTF-8](#)
 - [PEP 520: Preserving Class Attribute Definition Order](#)

Next topic
[What's New in Python](#)

This Page
[Report a Bug](#)
[Show Source](#)

Where do you get Anaconda?

<https://www.anaconda.com/download/>



Scroll down to the bottom of the Anaconda Screen

The screenshot shows the Anaconda download page for Windows. The page title is "Anaconda 5.0.0 For Windows Installer". There are two main sections: "Python 3.6 version *" and "Python 2.7 version *". Each section has a green "Download" button. Below the buttons are links to "64-Bit Graphical Installer" and "32-Bit Graphical Installer". A red arrow points from the "32-Bit Graphical Installer (436 MB)" link to the downloaded file in the taskbar. The taskbar shows a file named "Anaconda3-5.0.0-Windows-x86_64" with a size of "522,210 KB".

Secure | <https://www.anaconda.com/download/>

ANAconda

Anaconda Cloud Documentation Blog Contact

What is Anaconda? Products Support Community About Resources [Download](#)

Windows macOS Linux

Anaconda 5.0.0 For Windows Installer

I am using 3.6

Python 3.6 version *

[Download](#)

[64-Bit Graphical Installer \(535 MB\)](#) ⓘ

[32-Bit Graphical Installer \(436 MB\)](#)

Zoran uses 2.7

Python 2.7 version *

[Download](#)

[64-Bit Graphical Installer \(522 MB\)](#) ⓘ

[32-Bit Graphical Installer \(421 MB\)](#)

[Behind a firewall?](#)

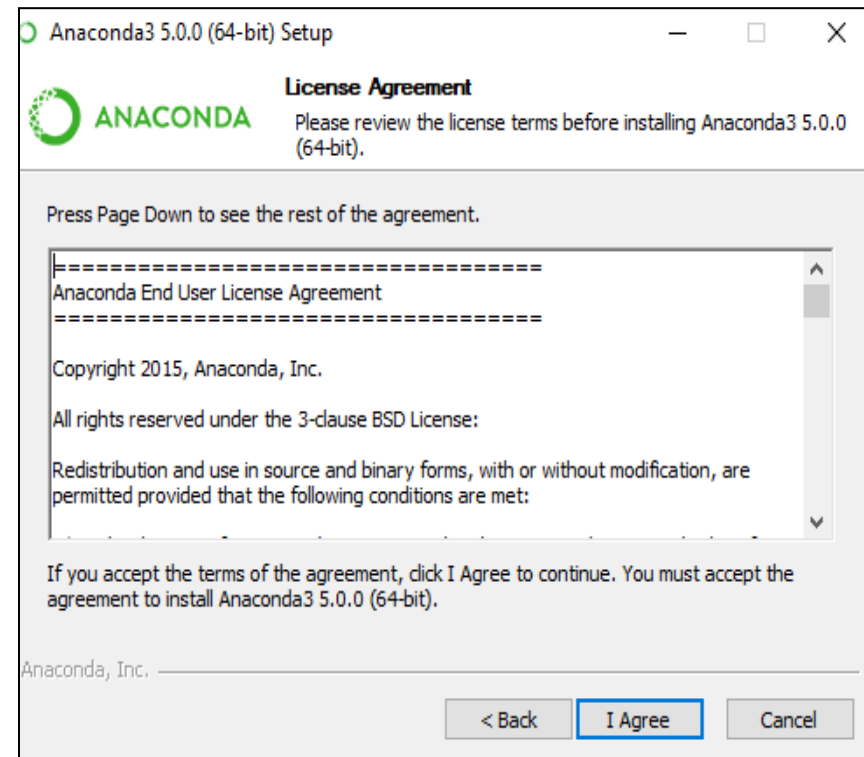
[How to get Python 3.5 or other Python versions](#)

[How to Install ANACONDA](#)

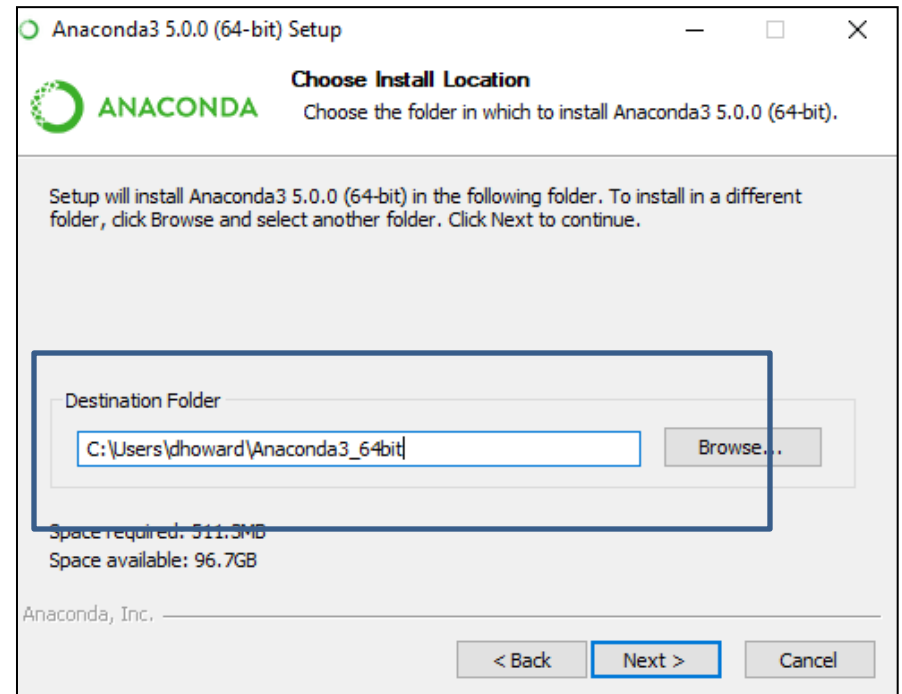
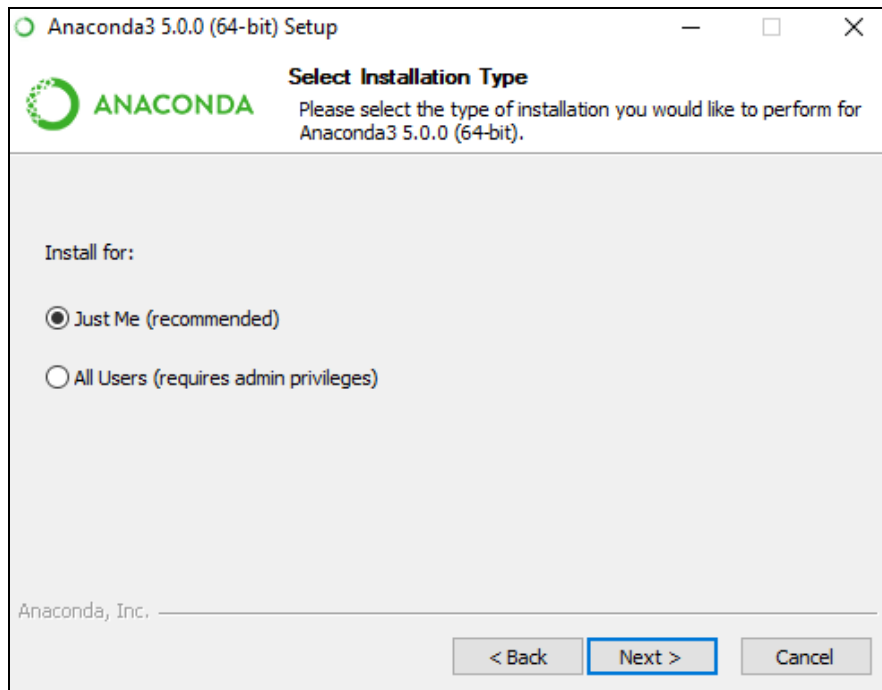
Anaconda3-5.0.0-Windows-x86_64	10/17/2017 10:26 ...	Application	522,210 KB
--------------------------------	----------------------	-------------	------------

Anaconda3 (64-bit) Installation

 Anaconda3-5.0.0-Windows-x86_64	10/17/2017 10:26 ...	Application	522,210 KB
--	----------------------	-------------	------------

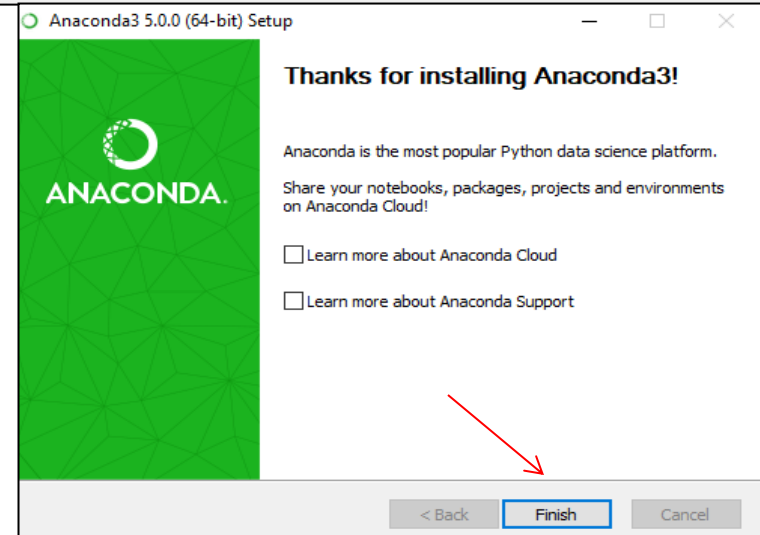
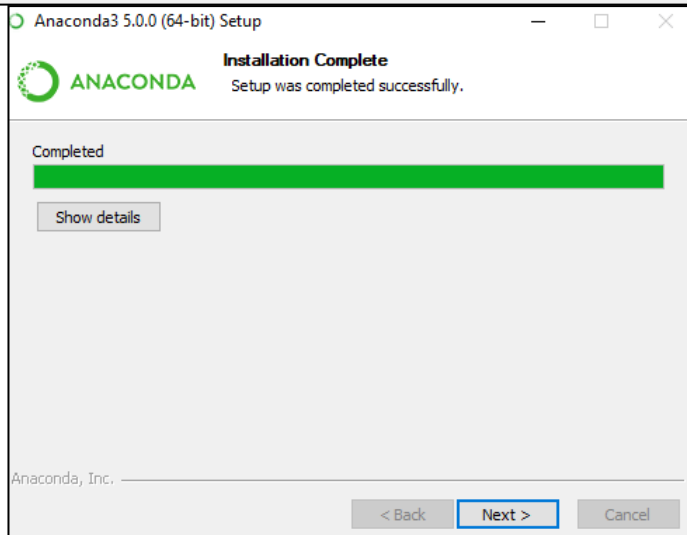
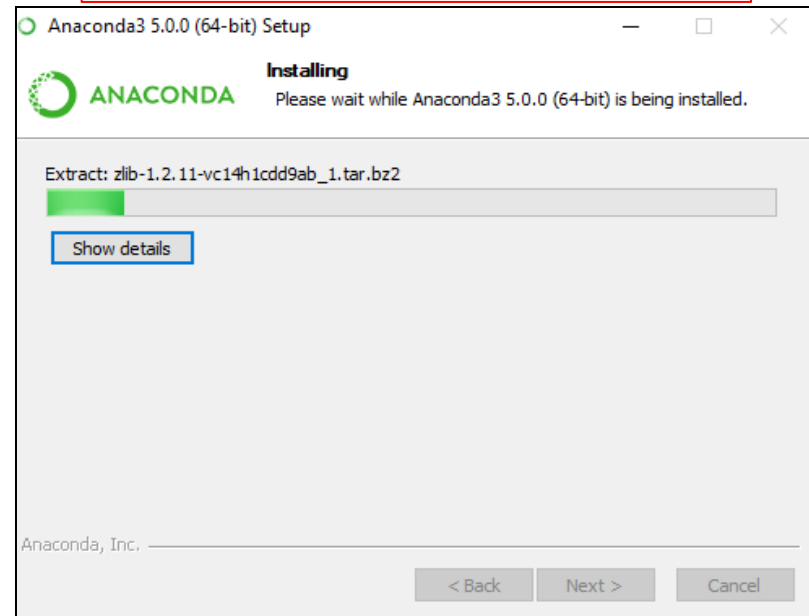
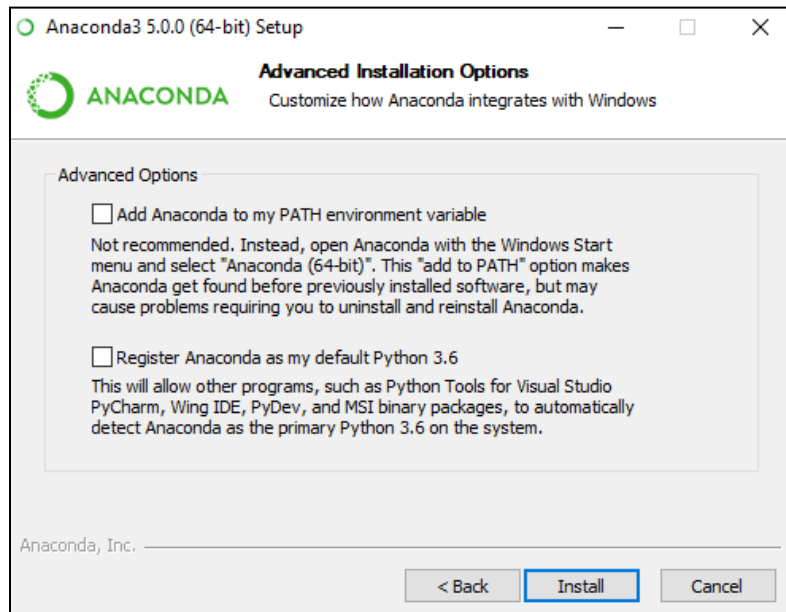


Anaconda3 (64-bit) Installation continued



Anaconda3 (64-bit) Installation continued...

This step takes a very long time!



Test Anaconda

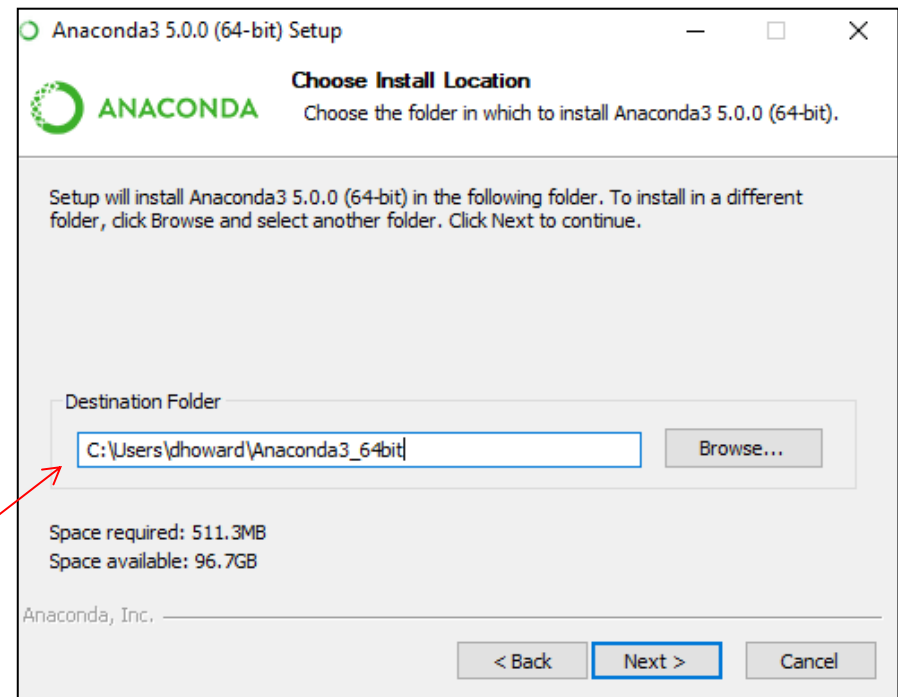
Start up Anaconda Python

- Go to directory where you installed it.

>python

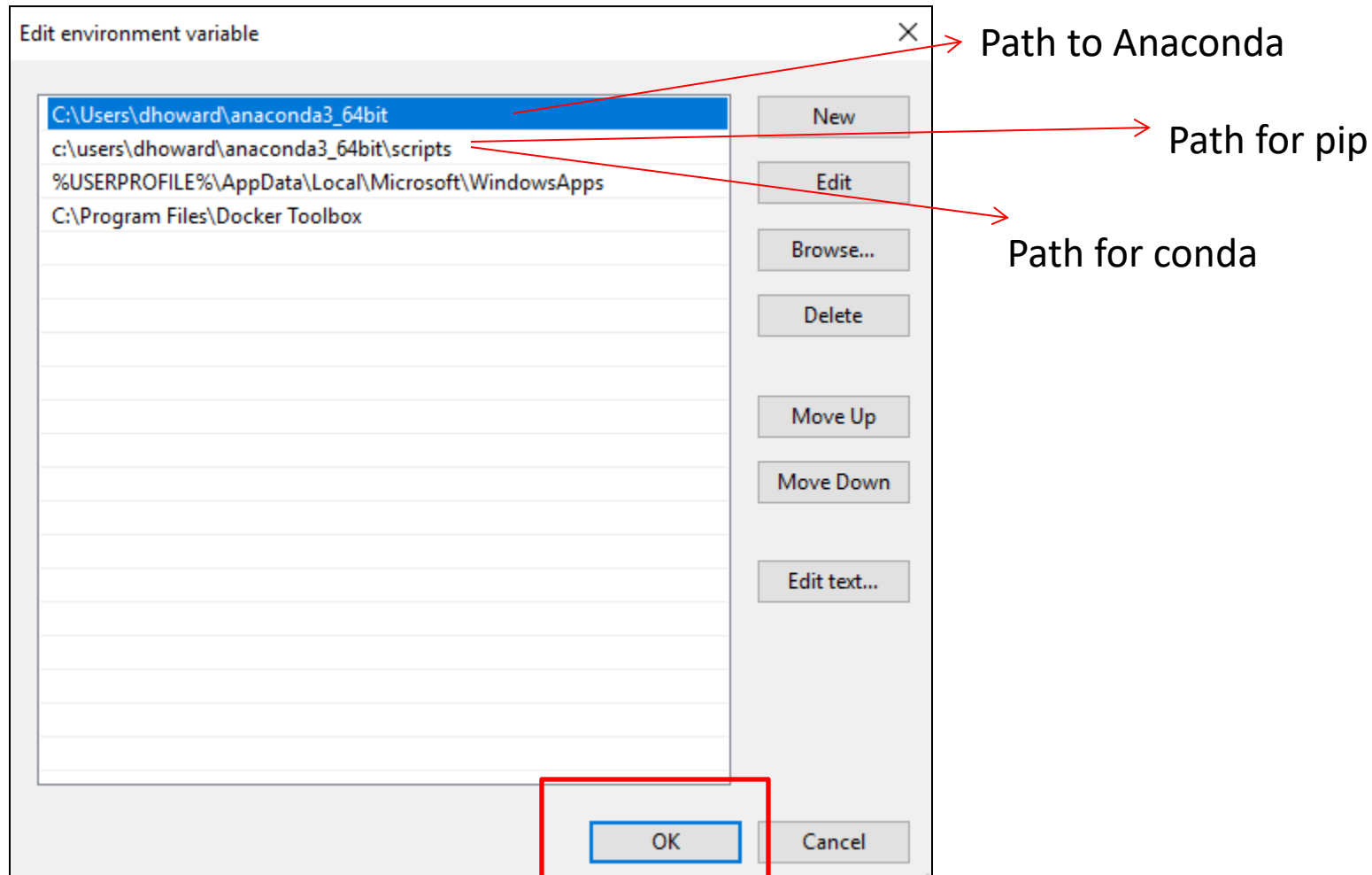
Note: in your Start up Menu you will see the following tools that was installed:

1. Anaconda Notebook
2. Jupyter Notebook
3. Anaconda Prompt



```
c:\Users\dhoward\Anaconda3_64bit>python
Python 3.6.2 |Anaconda, Inc.| (default, Sep 19 2017, 08:03:39) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

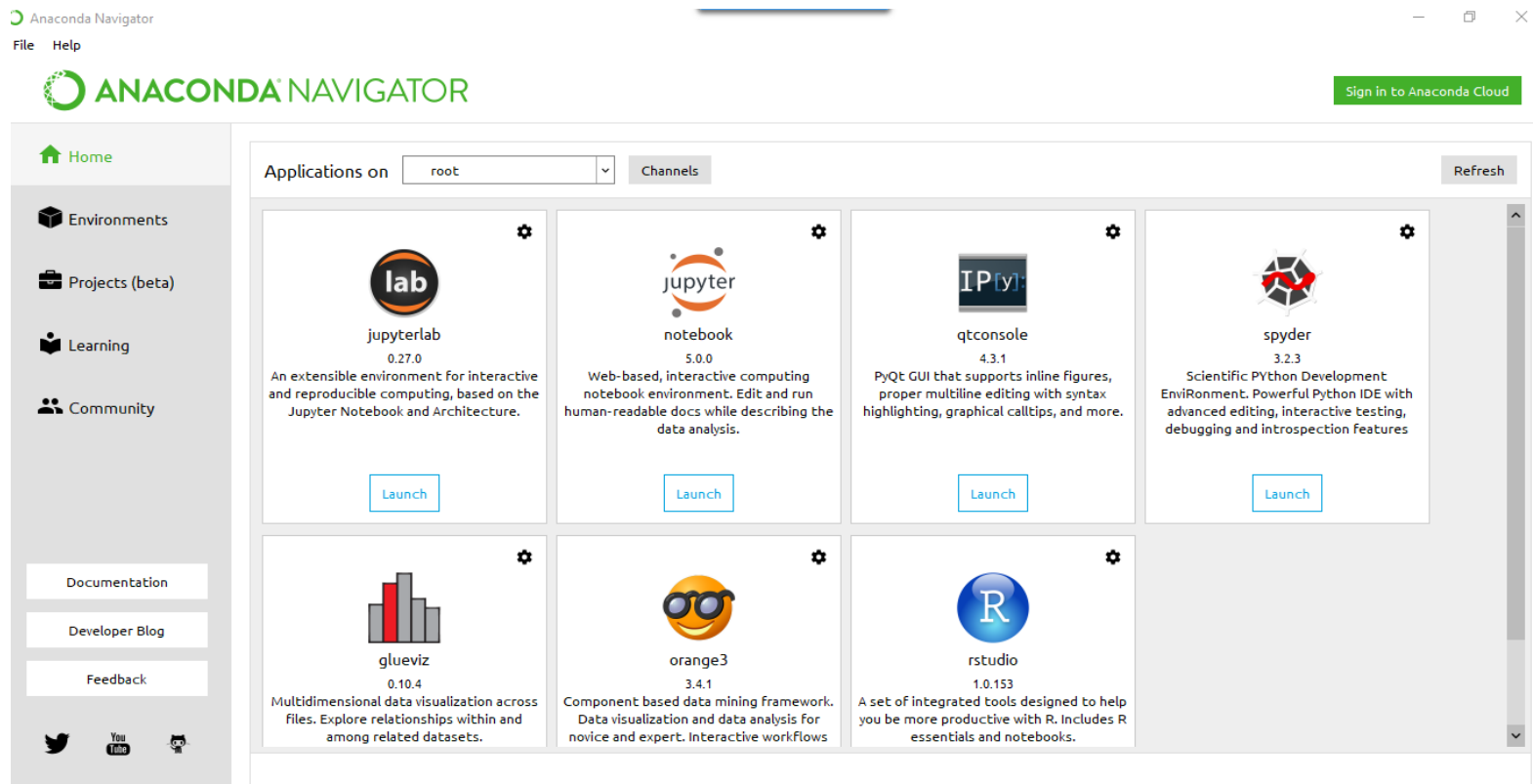
Set up Environment Variable for Anaconda3



Verification of Successful Anaconda Install

<https://docs.anaconda.com/anaconda/install/windows>

- ‘verify it by opening Anaconda Navigator, a program that is included with Anaconda: from your Windows Start menu, select the shortcut Anaconda Navigator. If Navigator opens, you have successfully installed Anaconda.’



Open a new Command Window and Test it out!

Check Python version (note: there are 2 hyphens)

>python --version

```
C:\Users\dhoward>python --version
Python 3.6.2 :: Anaconda, Inc.
```

Where is Python installed?

>where python

C:\Users\dhoward\anaconda3_64\python.exe

Where is pip installed?

>where pip

Where is conda installed?

>where conda

```
C:\Users\dhoward>where python
C:\Users\dhoward\anaconda3_64bit\python.exe
```

```
C:\Users\dhoward>where pip
c:\Python27\Scripts\pip.exe
c:\Users\dhoward\anaconda3_64bit\Scripts\pip.exe
```

```
C:\Users\dhoward>where conda
c:\Users\dhoward\anaconda3_64bit\Scripts\conda.exe
```

How do I start Python Anaconda?

Start the Anaconda Interpreter:

C:\>python

Notice: version 3.6.2 and 64-bit

```
C:\Users\dhoward>python
Python 3.6.2 [Anaconda, Inc.] (default, Sep 19 2017, 08:03:39) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 3+5
8
>>>
```

NLTK Download Page

<https://pypi.python.org/pypi/nltk>

No need to do this as NLTK is pre-installed with Anaconda.

python™

» Package Index > nltk > 3.2.5

PACKAGE INDEX »

- Browse packages
- List trove classifiers
- RSS (latest 40 updates)
- RSS (newest 40 packages)
- Terms of Service
- PyPI Tutorial
- PyPI Security
- PyPI Support
- PyPI Bug Reports
- PyPI Discussion
- PyPI Developer Info

nltk 3.2.5

Natural Language Toolkit

The Natural Language Toolkit (NLTK) is a Python package requires Python 2.7, 3.4, or 3.5.

Author: Steven Bird
Home Page: <http://nltk.org/>
Keywords: NLP, CL, natural language

File	Type	Py Version	Uploaded on	Size
nltk-3.2.5.tar.gz (md5)	Source		2017-09-24	1MB
nltk-3.2.5.win32.exe (md5)	MS Windows installer	any	2017-09-24	1MB

Status
[Nothing to report](#)

Cannot install
Python version -32 required, which was not found in the registry.

nltk-3.2.5.win32.exe
1,799/1,799 KB

arsing,tagging,tokenizing,syntax,linguistics,language,natural

Setup nltk-3.2.5

nltk-3.2.5

Don't do this

NLTK install (Anaconda)

C:\Users\dhoward>where python

C:\Users\dhoward\anaconda3_64bit\python.exe

c:\Users\dhoward>pip install nltk (nltk is already packaged with Anaconda)

Requirement already satisfied: nltk in c:\python27\lib\site-packages

Requirement already satisfied: six in c:\python27\lib\site-packages (from nltk)

Let's check now:

```
c:\Users\dhoward>python
Python 3.6.2 |Anaconda, Inc.| (default, Sep 19 2017, 08:03:39) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>>
```

Jupyter Install on CentOS VM

@639 Much Thanks to Eduardo Morales!

- **Need to plot - Install Jupyter!!!**

The answer to all your plotting needs is Jupyter! (I wish someone had told me before.)

- Jupyter is not in either of the VMs we currently have. You have to install it. Here's how to install it in CentOS:

- Open a Terminal and enter as super-user, then

```
pip install ipython
```

```
python -m pip install --upgrade
```

```
pip python -m pip install jupyter
```

then run it with this:

```
jupyter notebook
```

- However, later on you'll trip with the lack of Panda. So install it once and for all:

```
pip install pandas
```

Python 2.7 Install on CentOs

- You could use your VM for this homework but not necessary.

Python 2.7 will be required to perform this exercise. We will need to install Python 2.7 without disrupting the Python2.6 which is installed on the HDFS system.

Install Python2.7 on CentOS 6.7 (coexisting with Python2.6.6)

```
$ cat /etc/redhat-release
CentOS release 6.7 (Final)

$ sudo yum groupinstall "Development tools"

$ sudo yum install zlib-devel
$ sudo yum install bzip2-devel
$ sudo yum install openssl-devel
$ sudo yum install ncurses-devel
$ sudo yum install sqlite-devel

(in ~cloudera/python2.7)
$ wget --no-check-certificate https://www.python.org/ftp/python/2.7.6/Python-2.7.6.tar.xz
$ tar xf Python-2.7.6.tar.xz
$ cd Python-2.7.6
$ ./configure --prefix=/usr/local
$ make
$ sudo make install (to install in /usr/local/bin)
```

Create a virtual environment for Python 2.7

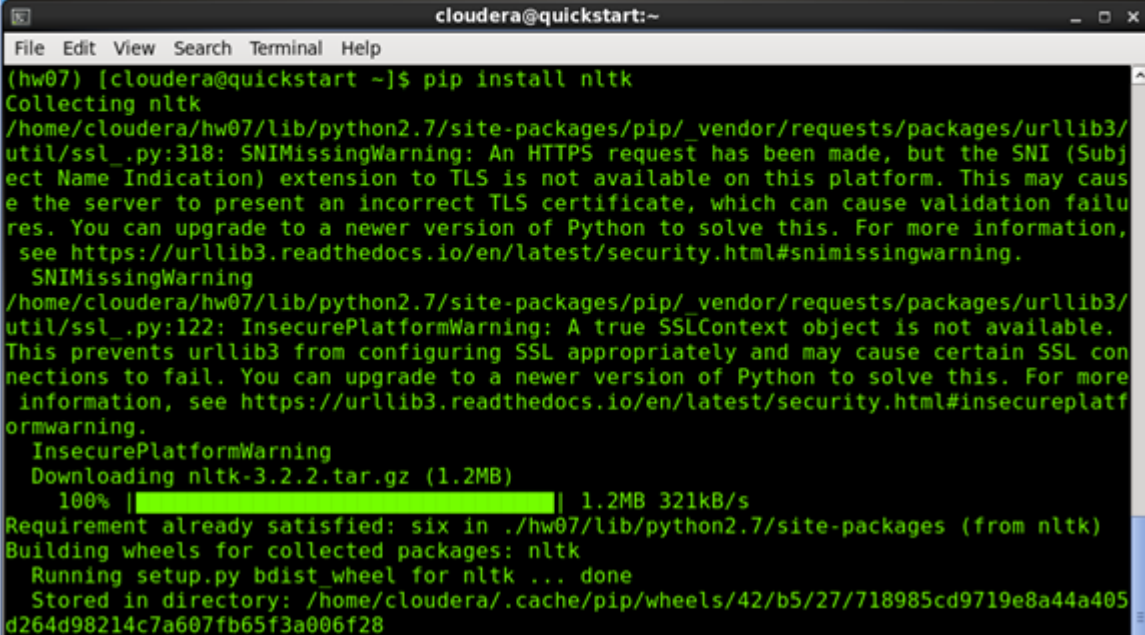
```
$ pip install virtualenv
$ virtualenv -p /usr/local/bin/python2.7 hw07
$ source hw07/bin/activate

$ deactivate (to stop the virtual environment)
```

CentOS NLTK Install - continued

Install `nltk` on the HDFS virtual system

```
$ pip install nltk
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
(hw07) [cloudera@quickstart ~]$ pip install nltk  
Collecting nltk  
/home/cloudera/hw07/lib/python2.7/site-packages/pip/_vendor/requests/packages/urllib3/  
util/ssl_.py:318: SNIMissingWarning: An HTTPS request has been made, but the SNI (Subj  
ect Name Indication) extension to TLS is not available on this platform. This may caus  
e the server to present an incorrect TLS certificate, which can cause validation failu  
res. You can upgrade to a newer version of Python to solve this. For more information,  
see https://urllib3.readthedocs.io/en/latest/security.html#snimissingwarning.  
  SNIMissingWarning  
/home/cloudera/hw07/lib/python2.7/site-packages/pip/_vendor/requests/packages/urllib3/  
util/ssl_.py:122: InsecurePlatformWarning: A true SSLContext object is not available.  
This prevents urllib3 from configuring SSL appropriately and may cause certain SSL con  
nections to fail. You can upgrade to a newer version of Python to solve this. For more  
information, see https://urllib3.readthedocs.io/en/latest/security.html#insecureplatf  
ormwarning.  
  InsecurePlatformWarning  
Downloading nltk-3.2.2.tar.gz (1.2MB)  
100% |████████████████████████████████████████| 1.2MB 321kB/s  
Requirement already satisfied: six in ./hw07/lib/python2.7/site-packages (from nltk)  
Building wheels for collected packages: nltk  
  Running setup.py bdist_wheel for nltk ... done  
  Stored in directory: /home/cloudera/.cache/pip/wheels/42/b5/27/718985cd9719e8a44a405  
d264d98214c7a607fb65f3a006f28
```

Download NLTK books on your CentOS Vm

Download NLTK books

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
(hw07) [cloudera@quickstart ~]$ python  
Python 2.7.6 (default, Mar 19 2017, 12:17:41)  
[GCC 4.4.7 20120313 (Red Hat 4.4.7-17)] on linux2  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import nltk  
>>> nltk.download()  
NLTK Downloader  
-----  
d) Download  l) List  u) Update  c) Config  h) Help  q) Quit  
-----  
Downloader> d book  
Downloading collection u'book'  
|  
| Downloading package abc to /home/cloudera/nltk_data...  
| Unzipping corpora/abc.zip.  
| Downloading package brown to /home/cloudera/nltk_data...  
| Unzipping corpora/brown.zip.  
| Downloading package chat80 to /home/cloudera/nltk_data...  
| Unzipping corpora/chat80.zip.
```

Import the book selection and make some test examinations

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
(hw07) [cloudera@quickstart ~]$ python  
Python 2.7.6 (default, Mar 19 2017, 12:17:41)  
[GCC 4.4.7 20120313 (Red Hat 4.4.7-17)] on linux2  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import nltk  
>>> from nltk.book import *  
*** Introductory Examples for the NLTK Book ***  
Loading text1, ..., text9 and sent1, ..., sent9  
Type the name of the text or sentence to view it.  
Type: 'texts()' or 'sents()' to list the materials.  
text1: Moby Dick by Herman Melville 1851  
text2: Sense and Sensibility by Jane Austen 1811  
text3: The Book of Genesis  
text4: Inaugural Address Corpus  
text5: Chat Corpus  
text6: Monty Python and the Holy Grail  
text7: Wall Street Journal  
text8: Personals Corpus  
text9: The Man Who Was Thursday by G . K . Chesterton 1908  
>>> █
```

Credit:
Stephen Ford

Jupyter Notebook

- Jupyter Notebook comes with Anaconda distribution. You don't need to install anything. Just run the password command (see below)
- Try it out as it has many advantages – saves your work very nicely too!
- FIRST: Run the following to define your password:

```
>jupyter notebook password
```

```
Enter password: **** Verify password: ****
```

To run Jupyter:

Go to your browser and enter:

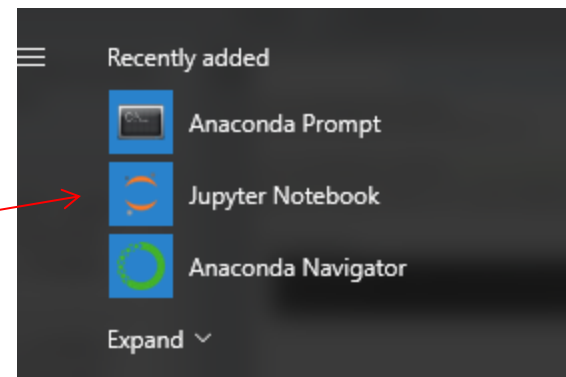
localhost:8888

OR

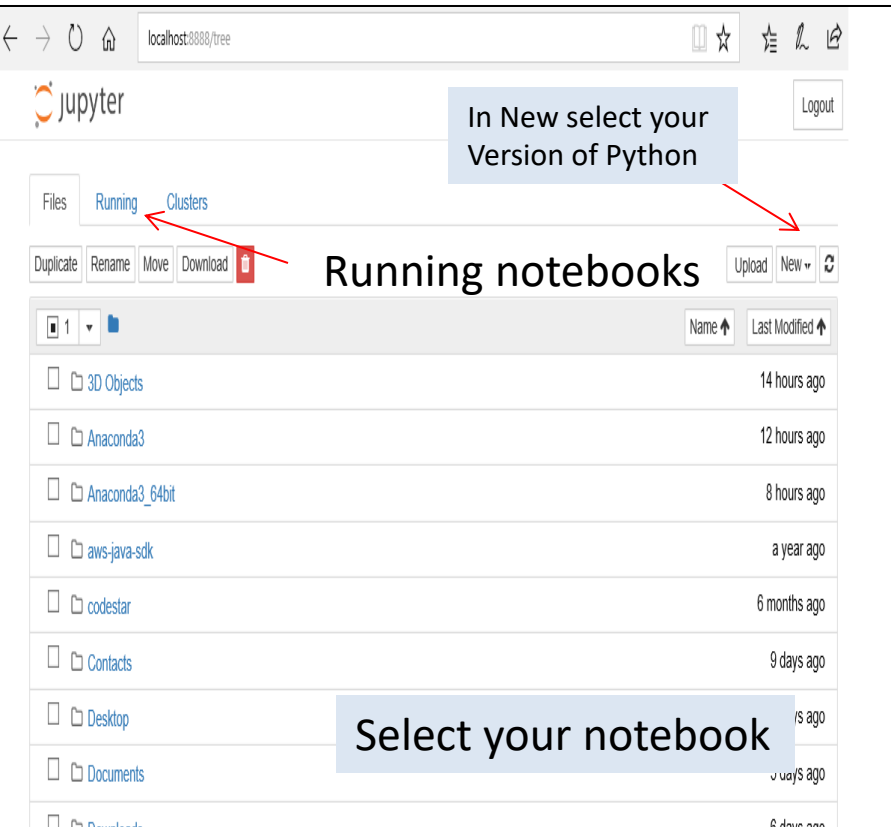
>jupyter notebook → This starts the notebook server from the command line.

```
$ jupyter notebook
[I 08:58:24.417 NotebookApp] Serving notebooks from local directory: /Users/catherine
[I 08:58:24.417 NotebookApp] 0 active kernels
[I 08:58:24.417 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/
[I 08:58:24.417 NotebookApp] Use Control-C to stop this server and shut down all kernels
```

Find the icon in your Start menu

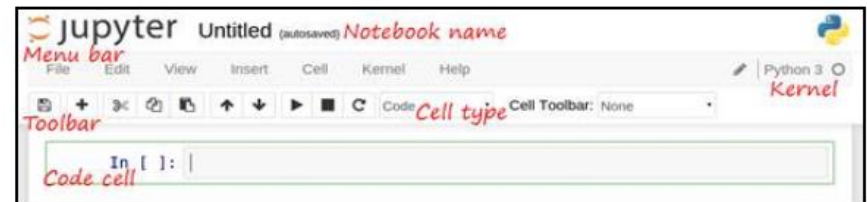


Navigating in Jupyter



- Helpful commands:

<https://datahub.packtpub.com/tutorials/basics-jupyter-notebook-python/>



Ctrl + Enter: run the cell

A new notebook

Shift + Enter: run the cell and select the cell below

Alt + Enter: run the cell and insert a new cell below

Ctrl + S: save the notebook

Edit Mode:

Esc: switch to command mode

Ctrl + Shift + -: split the cell

Command Mode:

Enter: switch to edit mode

Up or *k*: select the previous cell

Down or *j*: select the next cell

y / *m*: change the cell type to code cell/Markdown cell

a / *b*: insert a new cell above/below the current cell

x / *c* / *v*: cut/copy/paste the current cell

dd: delete the current cell

z: undo the last delete operation

Shift + =: merge the cell below

h: display the help menu with the list of keyboard shortcuts

Keyboard Shortcuts Edit Mode

- Select cell, press ESC and h to see this menu

 Keyboard shortcuts ✕

Edit Mode (press Enter to enable)

Tab : code completion or indent	Ctrl-Right : go one word right
Shift-Tab : tooltip	Ctrl-Backspace : delete word before
Ctrl-] : indent	Ctrl-Delete : delete word after
Ctrl-[: dedent	Ctrl-M : command mode
Ctrl-A : select all	Ctrl-Shift-F : open the command palette
Ctrl-Z : undo	Ctrl-Shift-P : open the command palette
Ctrl-Shift-Z : redo	Esc : command mode
Ctrl-Y : redo	Shift-Enter : run cell, select below
Ctrl-Home : go to cell start	Ctrl-Enter : run selected cells
Ctrl-Up : go to cell start	Alt-Enter : run cell, insert below
Ctrl-End : go to cell end	Ctrl-Shift-Minus : split cell
Ctrl-Down : go to cell end	Ctrl-S : Save and Checkpoint

Close

Keyboard Shortcuts Command Mode

- Select cell, press ESC and h to see this menu

Keyboard shortcuts

The Jupyter Notebook has two different keyboard input modes. **Edit mode** allows you to type code/text into a cell and is indicated by a green cell border. **Command mode** binds the keyboard to notebook level commands and is indicated by a grey cell border with a blue left margin.

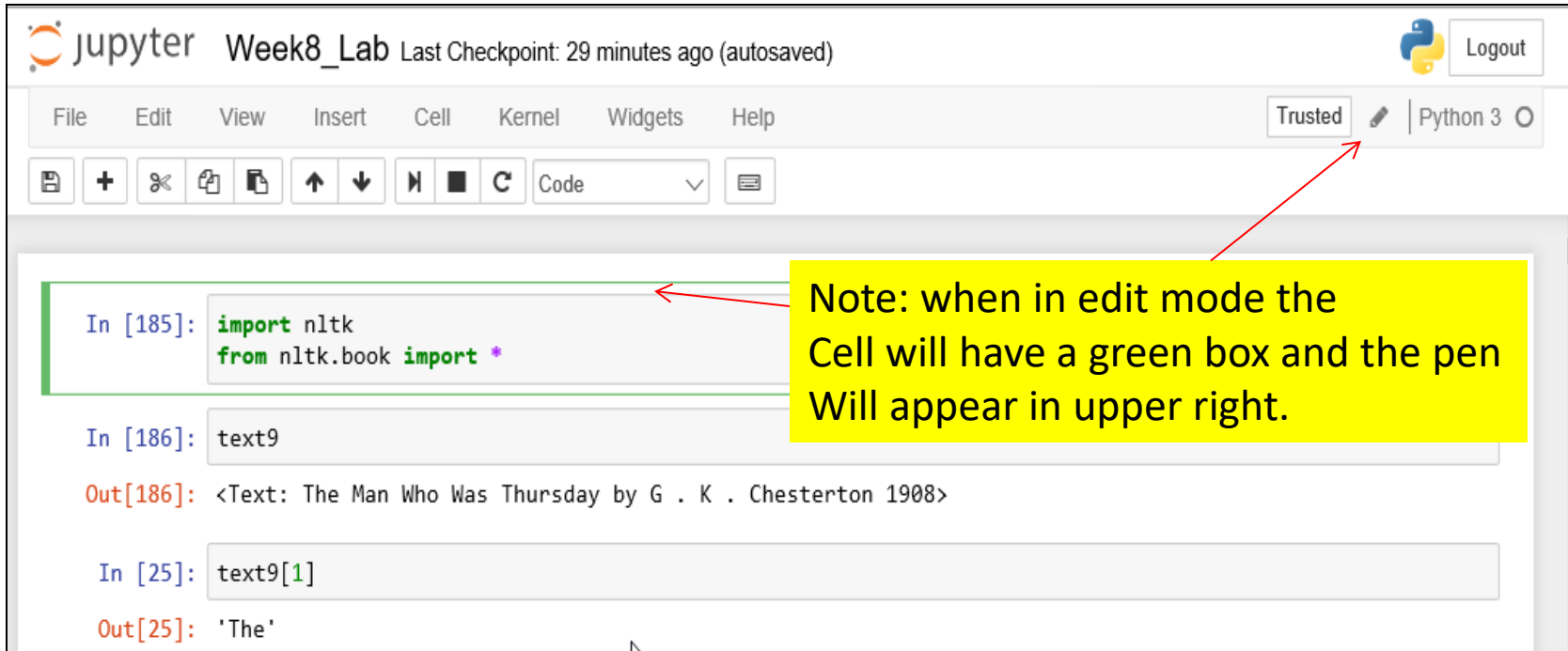
Command Mode (press `Esc` to enable)

Edit Shortcuts

<code>F</code> : find and replace	<code>Shift-Down</code> : extend selected cells below
<code>Ctrl-Shift-F</code> : open the command palette	<code>Shift-J</code> : extend selected cells below
<code>Ctrl-Shift-P</code> : open the command palette	<code>A</code> : insert cell above
<code>Enter</code> : enter edit mode	<code>B</code> : insert cell below
<code>P</code> : open the command palette	<code>X</code> : cut selected cells
<code>Shift-Enter</code> : run cell, select below	<code>C</code> : copy selected cells
<code>Ctrl-Enter</code> : run selected cells	<code>Shift-V</code> : paste cells above

Close

My Jupyter Session



The screenshot shows a Jupyter Notebook interface. At the top, the title bar reads "jupyter Week8_Lab" followed by "Last Checkpoint: 29 minutes ago (autosaved)". On the right, there is a "Logout" button and a Python 3 logo. Below the title bar is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". A toolbar below the menu bar contains icons for saving, adding, deleting, copying, pasting, undo, redo, and a dropdown menu currently set to "Code".

The main area displays three code cells. The first cell, labeled "In [185]:", is in edit mode, indicated by a green border and a green box around the code. The code in this cell is:

```
import nltk
from nltk.book import *
```

A red arrow points from a yellow note box to this cell. The second cell, labeled "In [186]:", contains the code `text9`. Below it, the output is shown: `Out[186]: <Text: The Man Who Was Thursday by G . K . Chesterton 1908>`. The third cell, labeled "In [25]:", contains the code `text9[1]`. Below it, the output is shown: `Out[25]: 'The'`.

In the upper right corner of the interface, there is a "Trusted" button with a pencil icon next to it, and a "Python 3" label with a circular icon. A red arrow points from the yellow note box to the "Trusted" button.

Note: when in edit mode the Cell will have a green box and the pen Will appear in upper right.

NLTK Data

www.nltk.org/nltk_data

NLTK Data

www.nltk.org/nltk_data/

Some of the Corpora and Corpus Samples Distributed with NLTK: For information about downloading and using them, please consult the NLTK

NLTK Corpora

NLTK has built-in support for dozens of corpora and train corpus downloader, >>> nltk.download()

Please consult the README file included with each corp

1. *ACE Named Entity Chunker (Maximum entropy)* [download] id: maxent_ne_chunker; size: 13404747; author: ; co
2. *Australian Broadcasting Commission 2006* [download] id: abc; size: 1487851; author: Australian Broadcas
3. *Alpino Dutch Treebank* [download | source] id: alpino; size: 2797255; author: ; copyright: ; lice
4. *BioCreative Critical Assessment of Information E* id: biocreative_ppi; size: 223566; author: ; copyrig
5. *Brown Corpus* [download | source] id: brown; size: 3314357; author: W. N. Francis and
6. *Brown Corpus (TEI XML Version)* [download | sou] id: brown_tei; size: 8737738; author: W. N. Francis
7. *CESS-CAT Treebank* [download | source] id: cess_cat; size: 5396688; author: ; copyright: ; li Martí, MarionaTaulé, Lluís Márquez, Manuel Bertr http://www.lsi.upc.edu/~mbertran/cess-ece/publicat
8. *CESS-ESP Treebank* [download | source]

Corpus	Compiler	Contents
Brown Corpus	Francis, Kucera	15 genres, 1.15M words, tagged, categorized
CESS Treebanks	CLIC-UB	1M words, tagged and parsed (Catalan, Spanish)
Chat-80 Data Files	Pereira & Warren	World Geographic Database
CMU Pronouncing Dictionary	CMU	127k entries
CoNLL 2000 Chunking Data	CoNLL	270k words, tagged and chunked
CoNLL 2002 Named Entity	CoNLL	700k words, pos- and named-entity-tagged (Dutch, Spanish)
CoNLL 2007 Dependency Treebanks (sel)	CoNLL	150k words, dependency parsed (Basque, Catalan)
Dependency Treebank	Narad	Dependency parsed version of Penn Treebank sample
FrameNet	Fillmore, Baker et al	10k word senses, 170k manually annotated sentences
Floresta Treebank	Diana Santos et al	9k sentences, tagged and parsed (Portuguese)
Gazetteer Lists	Various	Lists of cities and countries
Genesis Corpus	Misc web sources	6 texts, 200k words, 6 languages
Gutenberg (selections)	Hart, Newby, et al	18 texts, 2M words
Inaugural Address Corpus	Cspan	US Presidential Inaugural Addresses (1789-present)
Indian POS-Tagged Corpus	Kumaran et al	60k words, tagged (Bangla, Hindi, Marathi, Telugu)
MacMorpho Corpus	NILC, USP, Brazil	1M words, tagged (Brazilian Portuguese)
Movie Reviews	Pang, Lee	2k movie reviews with sentiment polarity classification
Names Corpus	Kantrowitz, Ross	8k male and female names
NIST 1999 Info Extr (selections)	Garofolo	63k words, newswire and named-entity SGML markup
Nombank	Meyers	115k propositions, 1400 noun frames
NPS Chat Corpus	Forsyth, Martell	10k IM chat posts, POS-tagged and dialogue-act tagged
Open Multilingual WordNet	Bond et al	15 languages, aligned to English WordNet
PP Attachment Corpus	Ratnaparkhi	28k prepositional phrases, tagged as noun or verb modifiers
Proposition Bank	Palmer	113k propositions, 3300 verb frames
Question Classification	Li, Roth	6k questions, categorized
Reuters Corpus	Reuters	1.3M words, 10k news documents, categorized
Roget's Thesaurus	Project Gutenberg	200k words, formatted text
RTE Textual Entailment	Dagan et al	8k sentence pairs, categorized
SEMECOR	Rus, Mihalcea	880k words, part-of-speech and sense tagged
Senseval 2 Corpus	Pedersen	600k words, part-of-speech and sense tagged
SentiWordNet	Esuli, Sebastiani	sentiment scores for 145k WordNet synonym sets
Shakespeare texts (selections)	Bosak	8 books in XML format
State of the Union Corpus	CSPAN	485k words, formatted text
Stopwords Corpus	Porter et al	2,400 stopwords for 11 languages
Swadesh Corpus	Wiktionary	comparative wordlists in 24 languages
Switchboard Corpus (selections)	LDC	36 phonecalls, transcribed, parsed
Univ Decl of Human Rights	United Nations	480k words, 300+ languages
Penn Treebank (selections)	LDC	40k words, tagged and parsed
TIMIT Corpus (selections)	NIST/LDC	audio files and transcripts for 16 speakers
VerbNet 2.1	Palmer et al	5k verbs, hierarchically organized, linked to WordNet
Wordlist Corpus	OpenOffice.org et al	960k words and 20k affixes for 8 languages
WordNet 3.0 (English)	Miller, Fellbaum	145k synonym sets

@Diane Howard

30

Types of Corpus Data Files

1. **Gutenberg Corpus** – electronic text archive from the Project Gutenberg and contains 25,000 free electronic books
2. **Web and Chat Text** - (discussion forums, overheard conversations, movie scripts, personal Advertisements, wine reviews.
3. **Brown Corpus** – 1st million word electronic corpus built by Brown Univ in 1961. Contains 5000 sources categorized by genre (e.g., news, editorials,...).
4. **Reuters Corpus** – contains 10,788 news totaling 1.3 million words with 90 topics.
5. **Inaugural Address Corpus** – 55 U.S. presidential addresses which has a time dimension.
6. **Annotated Text Corporations** – linguistic annotations (named entities, semantic meanings,...)
7. **Other Languages** – (also prints special characters for each language).
8. **Text Corpus Structure** – unstructured, news collections,
9. **Your own Corpus** – your own collection of files can be imported...

1.8 Text Corpus Structure

We have seen a variety of corpus structures so far; these are summarized in [1.3](#). The simplest kind lacks any structure: it is just a collection of texts. Often, texts are grouped into categories that might correspond to genre, source, author, language, etc. Sometimes these categories overlap, notably in the case of topical categories as a text can be relevant to more than one topic. Occasionally, text collections have temporal structure, news collections being the most common example.

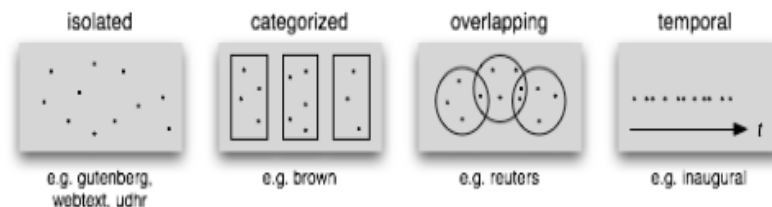


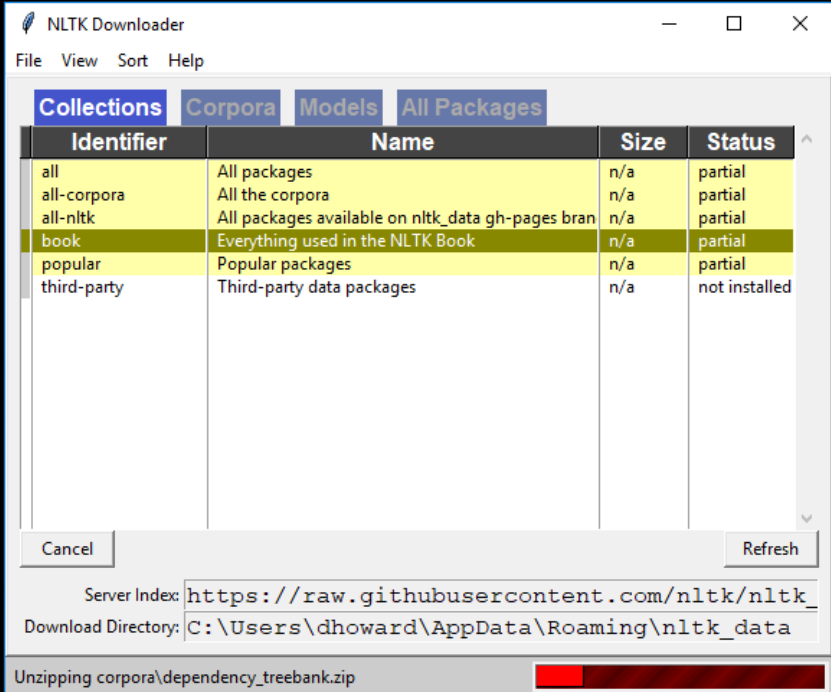
Figure 1.3: Common Structures for Text Corpora: The simplest kind of corpus is a collection of isolated texts with no particular organization; some corpora are structured into categories like genre (Brown Corpus); some categorizations overlap, such as topic categories (Reuters Corpus); other corpora represent language use over time (Inaugural Address Corpus).

Let's get started with NLTK:

- *Import the NLTK datasets.*
- *Select the **book** Collection in the Identifier column*
- *Notice where the download directory stores the text files for you.*

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhoward>python
Python 3.6.2 [Anaconda, Inc.] (default, Sep 20 2017, 13:35:58) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.download()
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
```

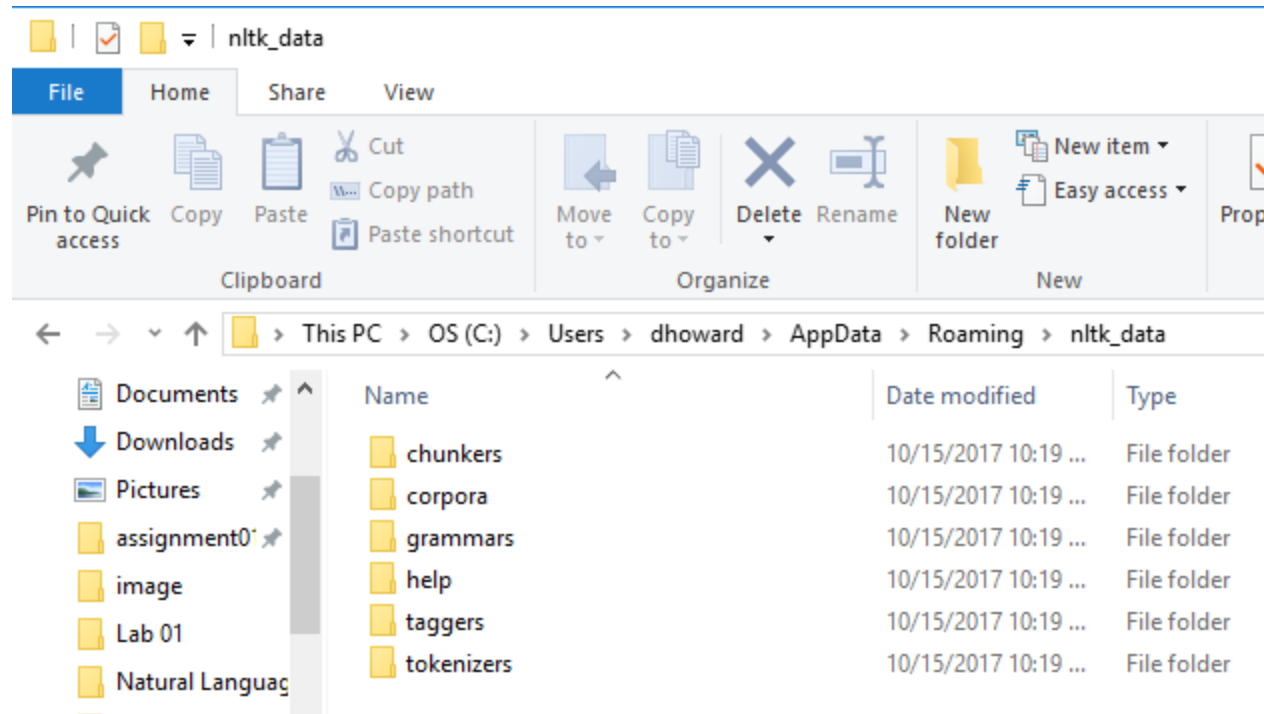


The screenshot shows the NLTK Downloader application window. The 'Collections' tab is active, displaying a table of available collections. The 'book' collection is highlighted in green. Below the table, the 'Server Index' is set to https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml and the 'Download Directory' is `C:\Users\dhoward\AppData\Roaming\nltk_data`. A progress bar at the bottom indicates the download of `corpora\dependency_treebank.zip`.

Identifier	Name	Size	Status
all	All packages	n/a	partial
all-corpora	All the corpora	n/a	partial
all-nltk	All packages available on nltk_data gh-pages branch	n/a	partial
book	Everything used in the NLTK Book	n/a	partial
popular	Popular packages	n/a	partial
third-party	Third-party data packages	n/a	not installed

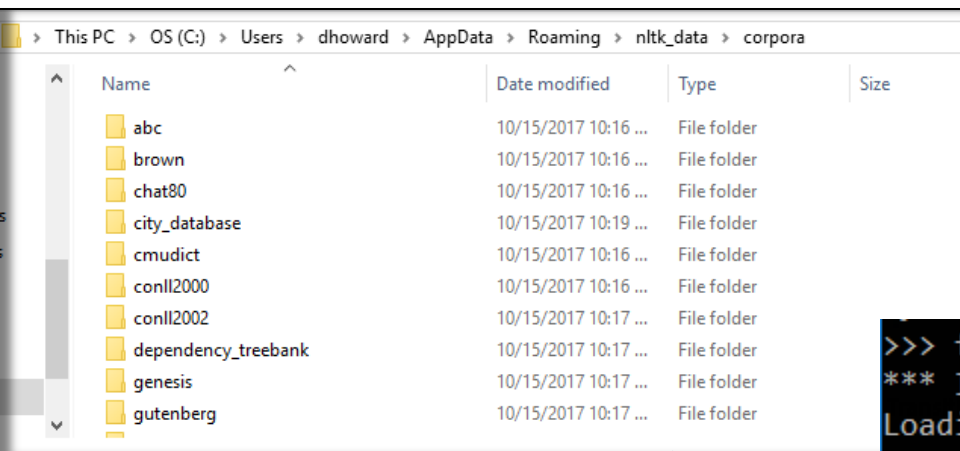
Examine our Book directory

C:\Users\dhoward\AppData\Roaming\nltk_data

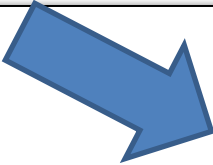


Importing data

- from nltk.book import *




Name	Date modified	Type	Size
abc	10/15/2017 10:16 ...	File folder	
brown	10/15/2017 10:16 ...	File folder	
chat80	10/15/2017 10:16 ...	File folder	
city_database	10/15/2017 10:19 ...	File folder	
cmudict	10/15/2017 10:16 ...	File folder	
conll2000	10/15/2017 10:16 ...	File folder	
conll2002	10/15/2017 10:17 ...	File folder	
dependency_treebank	10/15/2017 10:17 ...	File folder	
genesis	10/15/2017 10:17 ...	File folder	
gutenberg	10/15/2017 10:17 ...	File folder	



```
>>> from nltk.book import *
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
```

How to access data

```
>>> from nltk.book import *
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
>>>
>>> text9
<Text: The Man Who Was Thursday by G . K . Chesterton 1908>
>>> text4
<Text: Inaugural Address Corpus>
```



'text' in Python is a list of words (i.e,
'a sequence of words and punctuations')

NLTK Functions for extracting words, categories, sentences, fileids

Table 1.3:

Basic Corpus Functionality defined in NLTK: more documentation can be found using `help(nltk.corpus.reader)` and by reading the online Corpus HOWTO at <http://nltk.org/howto>.

Example	Description
<code>fileids()</code>	the files of the corpus
<code>fileids([categories])</code>	the files of the corpus corresponding to these categories
<code>categories()</code>	the categories of the corpus
<code>categories([fileids])</code>	the categories of the corpus corresponding to these files
<code>raw()</code>	the raw content of the corpus
<code>raw(fileids=[f1,f2,f3])</code>	the raw content of the specified files
<code>raw(categories=[c1,c2])</code>	the raw content of the specified categories
<code>words()</code>	the words of the whole corpus
<code>words(fileids=[f1,f2,f3])</code>	the words of the specified fileids
<code>words(categories=[c1,c2])</code>	the words of the specified categories
<code>sents()</code>	the sentences of the whole corpus
<code>sents(fileids=[f1,f2,f3])</code>	the sentences of the specified fileids
<code>sents(categories=[c1,c2])</code>	the sentences of the specified categories
<code>abspath(fileid)</code>	the location of the given file on disk
<code>encoding(fileid)</code>	the encoding of the given file
<code>open(fileid)</code>	open a stream for the given file
<code>root</code>	if the path is relative, the root of the corpus
<code>readme()</code>	the content of the README file

```

d>>> words=gb.words("austen-sense.txt")
r>>> words[1:25]
['Sense', 'and', 'Sensibility', 'by', 'Jane', 'Austen', '1811', ''], 'CHAPTER', '1', 'The', 'family', 'of', 'Dashwood', 'and', 'had', 'long', 'been', 'settled', 'in', 'Sussex', '.', 'Their', 'estate', 'was']
>>> raw=gb.raw('austen-sense.txt')
>>> raw[1:25]
'Sense and Sensibility by'
>>> words=gb.words("austen-sense.txt")
>>> words[1:25]
['Sense', 'and', 'Sensibility', 'by', 'Jane', 'Austen', '1811', ''], 'CHAPTER', '1', 'The', 'family', 'of', 'Dashwood', 'and', 'had', 'long', 'been', 'settled', 'in', 'Sussex', '.', 'Their', 'estate', 'was']
>>> sents=gb.sents("austen-sense.txt")
>>> sents[1:25]
[['CHAPTER', '1'], ['The', 'family', 'of', 'Dashwood', 'had', 'long', 'been', 'settled', 'in', 'Sussex', '.'], ['The', 'estate', 'was', 'large', 'and', 'their', 'residence', 'was', 'at', 'Norland', 'Park', 'in', 'the', 'centre', 'of', 'their', 'property', 'where', 'for', 'many', 'generations', 'they', 'had', 'lived', 'in']]

```

Access Gutenberg Corpus Books

- ‘Project Gutenberg contains thousands of books, it represents established literature’
- It is a great resource for a small sampling from Gutenberg.

```
# import gutenberg texts
```

```
>>> from nltk.corpus import gutenberg as gb
```

```
>>> gb.fileids()
```

```
['austen-emma.txt', 'austen-persuasion.txt',  
'austen-sense.txt', 'bible-kjv.txt',  
'blake-poems.txt', 'bryant-stories.txt',  
'burgess-busterbrown.txt', 'carroll-alice.txt',  
'chesterton-ball.txt', 'chesterton-brown.txt',  
'chesterton-thursday.txt', 'edgeworth-parents.txt',  
'melville-moby_dick.txt', 'milton-paradise.txt',  
'shakespeare-caesar.txt', 'shakespeare-hamlet.txt',  
'shakespeare-macbeth.txt', 'whitman-leaves.txt']
```

alias – use these for shortening names

These are unique file identifiers for the 18 Gutenberg Corpus text files.

<http://www.nltk.org/book/ch02.html>

Text manipulation examples

```
>>> text9
```

```
<Text: The Man Who Was Thursday by G . K . Chesterton 1908>
```

```
>>> text9[1]
```

Note: indexes start at zero

```
'The'
```

```
>>>a=text9
```

```
>>>a[0]
```

```
>>>'['
```

```
>>>a[15]
```

```
>>>'Edmund'
```

```
>>> text9[1:13] → slicing (sublists)
```

```
['The', 'Man', 'Who', 'Was', 'Thursday', 'by', 'G', '!', 'K', '!', 'Chesterton', '1908']
```

```
>>> text9.index('Was')
```

```
4
```

```
>>> a=text9[1:13] → store sublist in 'a' to manipulate each field in the list
```

```
>>> print(a)
```

```
['The', 'Man', 'Who', 'Was', 'Thursday', 'by', 'G', '!', 'K', '!', '!', 'Chesterton', '1908']
```

```
>>> len(text7) → length of the entire text
```

```
100676
```

```
>>> len(text9)
```

```
69213
```

Jupyter example

```
In [186]: 1 text9
Out[186]:
Type: Text
String form: <Text: The Man Who Was Thursday by G . K . Chesterton 1908>
Length: 69213
File: c:\users\dhoward\anaconda3 64bit\lib\site-packages\nltk\text.py

In [25]:
Out[25]:
```

Jupyter example

```
In [155]: 1 len(set(text7))
Out[155]:
Type: Text
String form: <Text: Wall Street Journal>
Length: 100676
File: c:\users\dhoward\anaconda3 64bit\lib\site-packages\nltk\text.py

In [45]:
```

List Manipulations

- **Lists** = collection of data; a sequence of items; groups together other values (strings, ints, ...)
- comma-separated values (items) between square brackets.
- a=text9

<https://docs.python.org/3.4/library/functions.html>

- Some Methods:

append

clear

count

extend

insert

...

reverse

sort

List Comprehensions

- ‘concise way to create new list where each element is the result of some operations applied to each member of another sequence or iterable’
- ‘consists of brackets containing an expression followed by a for clause, then zero or more for or if clauses’
- $\{w \mid w \in V \ \& \ P(w)\}$ Property P . “The set of all w such that w is an element of a Vocabulary and w has Property P.”
- **Python format:**
- $[w \text{ for } w \text{ in } V \text{ if } p(w)] \rightarrow \text{output is a list}$

```
>>> squares = []
>>> for x in range(10):
...     squares.append(x**2)
...
>>> squares
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
squares = [x**2 for x in range(10)]
```

<https://docs.python.org/2/tutorial/datastructures.html>

@Diane Howard

String Manipulation Methods

```
>>> query = 'Hi from Diane!'
>>> query[8]
'D'
>>> query[2:]
' from Diane!'
>>> query + ' Have a great day!'
'Hi from Diane! Have a great day!'
```

```
1 query = 'Hi from Diane!'
```

```
Type:      str
String form: Hi from Diane!
Length:    14
Docstring:
```

```
for fileid in gb.fileids():
...     filelist.append(fileid)
...     fdist=nlTK.FreqDist(w.lower() for w in
gb.words(fileid))
...     print(fdist.most_common(50))
```

```
1 for fileid in gb.fileids():
2     num_chars = len(gb.get(fileid))
```

```
Type:      str
String form: whitman-leaves.txt
Length:    18
Docstring:
```

- String Methods:

capitalize

center

count

decode

encode

endswith

find

index

isdigit

...

lower

...

upper

Difference between Lists & Strings

- Strings & lists are both **sequences**.
- Both can be indexed , sliced and joined by concatenation (operators: +, -_.
- BUT you CANNOT join strings and lists.
- strings are **immutable** — you can't change a string once you create it
- strings are **immutable** — you can't change a string once you have create it
- Be careful of recognizing strings vs. lists.

```
>>> beatles[0] = "John Lennon"  
>>> del beatles[-1]  
>>> beatles ['John Lennon', 'Paul', 'George']
```

List

```
>>> query[0] = 'F'  
Traceback (most recent call last): File "<stdin>", line 1, in ?  
TypeError: object does not support item assignment
```

String

<http://www.nltk.org/book/ch03.html>

Frequency Distribution

- Used to determine frequency of words in a text.
Note: words must be tokenized. Sentences must be tokenized to words, etc.

```
>>>import nltk
>>>from nltk.corpus import gutenberg as gb
>>> for fileid in gb.fileids():
...     filelist.append(fileid)
...     fdist=nltk.FreqDist(w.lower() for w in gb.words(fileid))
...     print(fdist.most_common(50))
```

Frequency Distribution Functions

- These functions (FreqDist, CondFreqDist) describe the frequency (count) of each vocabulary item in the text.

```
>>> text4
```

```
<Text: Inaugural Address Corpus>
```

```
>>> fdist1=FreqDist(text4)
```

```
>>> fdist1 or use print(fdist1)
```

```
<FreqDist with 9754 samples and 145735 outcomes>
```

```
>>> fdist1.most_common(5) → prints word and frequency  
[('the', 9281), ('of', 6970), (',', 6840), ('and', 4991), ('.', 4676)]
```

Conditional Frequency Distribution

- Can combine 2 or more Frequency Distributions

**Create a table displaying relative frequencies with “modals”
(can, could, may, might, will, would and should)
are used in 18 texts provided by NLTK in the
extract from Gutenberg Corpus**

What is a modal? A helping verb

How do we define modals in Python? Lists!

```
>>> modals=['can','could','may','might','will','would','should']
>>> modals[0]
'can'
>>> modals[2:4]
['may', 'might']
>>> m=modals
>>> m[2:4]
['may', 'might']
```

```
import nltk
from nltk.corpus import gutenberg as gb

print("")
print("Relative frequencies of modals in 18 Gutenberg Corpus")
print("")
titles = gb.fileids()
modals = ['can', 'could', 'may', 'might', 'will', 'should']
cfd = nltk.ConditionalFreqDist(
    (text, word)
    for text in gb.fileids()
    for word in gb.words(text))
cfd.tabulate(conditions=titles, samples=modals)
```

Relative frequencies of modals in 18 Gutenberg Corpus						
	can	could	may	might	will	should
austen-emma.txt	270	825	213	322	559	366
austen-persuasion.txt	100	444	87	166	162	185
austen-sense.txt	206	568	169	215	354	228
bible-kjv.txt	213	165	1024	475	3807	768
blake-poems.txt	20	3	5	2	3	6
bryant-stories.txt	75	154	18	23	144	38
burgess-busterbrown.txt	23	56	3	17	19	13
carroll-alice.txt	57	73	11	28	24	27
chesterton-ball.txt	131	117	90	69	198	75
chesterton-brown.txt	126	170	47	71	111	56
chesterton-thursday.txt	117	148	56	71	109	54
edgeworth-parents.txt	340	420	160	127	517	271
melville-moby_dick.txt	220	215	230	183	379	181
milton-paradise.txt	107	62	116	98	161	55
shakespeare-caesar.txt	16	18	35	12	129	38
shakespeare-hamlet.txt	33	26	56	28	131	52
shakespeare-macbeth.txt	21	15	30	5	62	41
whitman-leaves.txt	88	49	85	26	261	42

Pandas

- Pandas = powerful Python data analysis toolkit. Data is stored in Data Frames.
- Install pandas using conda:

```
C:\Users\dhoward>conda install pandas
Fetching package metadata .....
Solving package specifications: .

Package plan for installation in environment c:\users\dhoward\anaconda3_64bit:

The following packages will be UPDATED:

    conda: 4.3.27-py36hcbae3bd_0 --> 4.3.30-py36h7e176b0_0

Proceed ([y]/n)? y

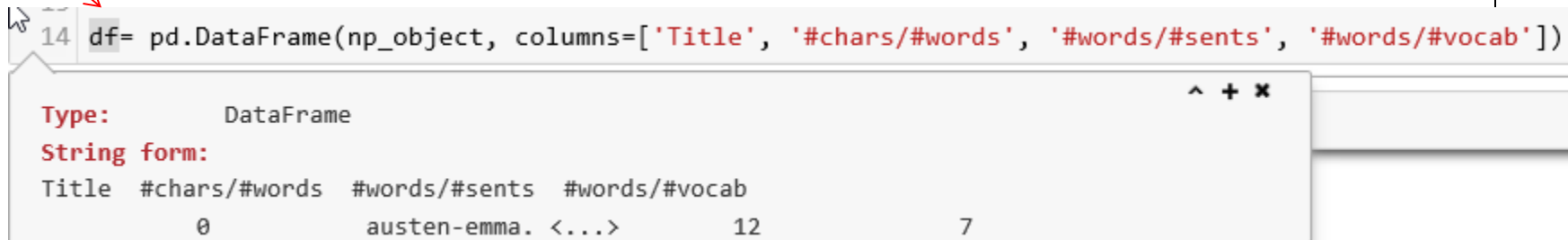
conda-4.3.30-p 100% |#####| Time: 0:00:00  2.48 MB/s
```

Data Frame

```
np_object=[]
for fileid in gb.fileids():
    num_chars = len(gb.raw(fileid))
    num_words = len(gb.words(fileid))
    num_sents = len(gb.sents(fileid))
    num_vocab = len(set(w.lower() for w in gb.words(fileid)))
    x= [fileid, round(num_chars/num_words), round(num_words/num_sents), round(num_words/num_vocab)
]
np_object.append(x) --
```

PUT DATA in DataFrame and fancy up columns and row headings

```
df= pd.DataFrame(np_object, columns=['Title', '#chars/#words', '#words/#sents', '#words/#vocab'])
```



The screenshot shows a Jupyter Notebook cell with the following code:

```
14 df= pd.DataFrame(np_object, columns=['Title', '#chars/#words', '#words/#sents', '#words/#vocab'])
```

A red arrow points from the `df` variable in the code above to the DataFrame preview below.

Type: DataFrame

String form:

Title	#chars/#words	#words/#sents	#words/#vocab
0	austen-emma. <...>	12	7

Parts of Speech

- 'A **part-of-speech** tagger, or **POS**-tagger - processes a sequence of words, and attaches a **part of speech** tag to each word'
- text-to-speech processors use this methodology.

Universal Part-of-Speech Tagset

Tag	Meaning	English Examples
ADJ	adjective	<i>new, good, high, special, big, local</i>
ADP	adposition	<i>on, of, at, with, by, into, under</i>
ADV	adverb	<i>really, already, still, early, now</i>
CONJ	conjunction	<i>and, or, but, if, while, although</i>
DET	determiner, article	<i>the, a, some, most, every, no, which</i>
NOUN	noun	<i>year, home, costs, time, Africa</i>
NUM	numeral	<i>twenty-four, fourth, 1991, 14:24</i>
PRT	particle	<i>at, on, out, over per, that, up, with</i>
PRON	pronoun	<i>he, their, her, its, my, I, us</i>
VERB	verb	<i>is, say, told, given, playing, would</i>
.	punctuation marks	<i>. , ; !</i>
X	other	<i>ersatz, esprit, dunno, gr8, univeristy</i>

Breaking down documents, sentences, phrases into parts of speech

1. Tokenize to words.
2. Tag Data – Determines parts of speech

```
>>>mysentence = "See Spot Run"
```

```
>>>tokenized_text = nltk.word_tokenize(mysentence)
```

```
>>>tagged_text = nltk.pos_tag(tokenized_text)
```

```
>>>print(tagged_text)
```

3. Define your Grammar for each word. Simplified categories:

Breaking down a sentence into parts of speech

See Spot run.

corenlp.run

Stanford CoreNLP

Text to annotate: See Spot run.

Annotations: parts-of-speech X named entities X dependency parse X openie X named entities (regexner) X

Language: English

Submit

Part-of-Speech:

1 See Spot run .

Named Entity Recognition:

1 See Spot run .



Helpful References I used

- <http://www.nltk.org/book>
- <https://www.digitalocean.com/community/tutorials/understanding-lists-in-python-3>
- The_Quick_Python_Book_Third_Edition_v5_MEAP.pdf
- Plus many other URLs sprinkled throughout.

BACKUP SLIDES

What do we have on Cloudera?

The screenshot displays a Linux desktop environment with a blue background. The top panel shows the 'Applications' menu, 'Places', and 'System' status. The desktop contains icons for 'Computer', 'cloudera's Home', 'Configure Kerberos', 'Eclipse', and 'Launch Cloudera Enterprise (trial)'. A web browser window is open to the 'Cloudera LIVE' portal, showing a 'Welcome to Your Cl' message and a table with columns 'Your Cluster' and 'Node'. The table lists 'Manager Node'. A terminal window titled 'cloudera@quickstart:~' is open, showing the following commands and output:

```
[cloudera@quickstart ~]$ python --version
Python 2.6.6
[cloudera@quickstart ~]$ python
Python 2.6.6 (r266:84292, Jul 23 2015, 15:22:56)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-11)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named nltk
>>>
```