

Python setup

Jim Bagrow

STAT/CS 287 — Data Science 1

Data Science 1 - Python setup

Outline:

1. Where to get Python
2. Setting up a work environment
 1. writing code with Spyder
 2. installing packages with Anaconda
3. Running code
 1. IPython—Interactive Python
4. Getting help

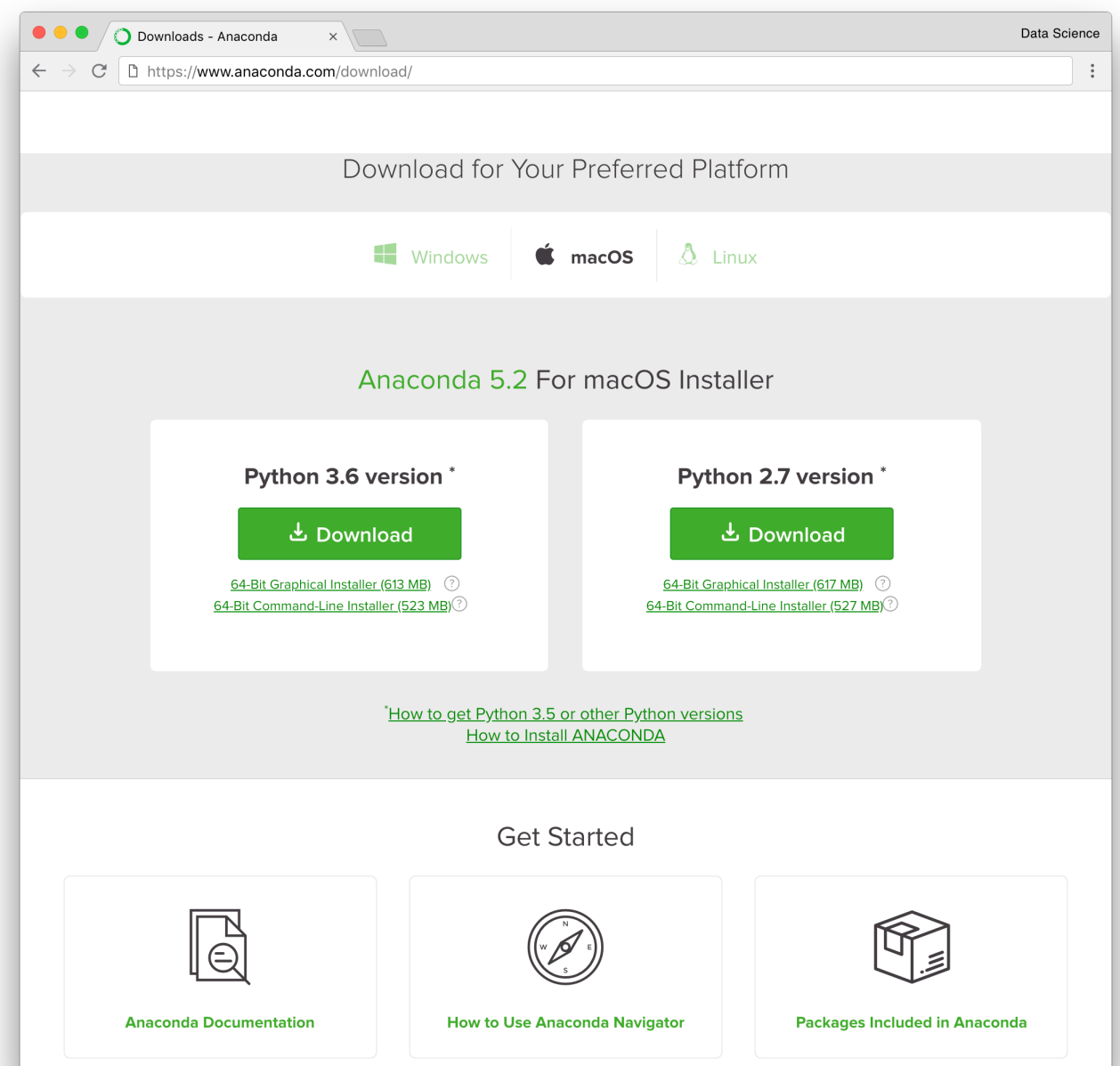
1. Where to get Python

Where to get Python

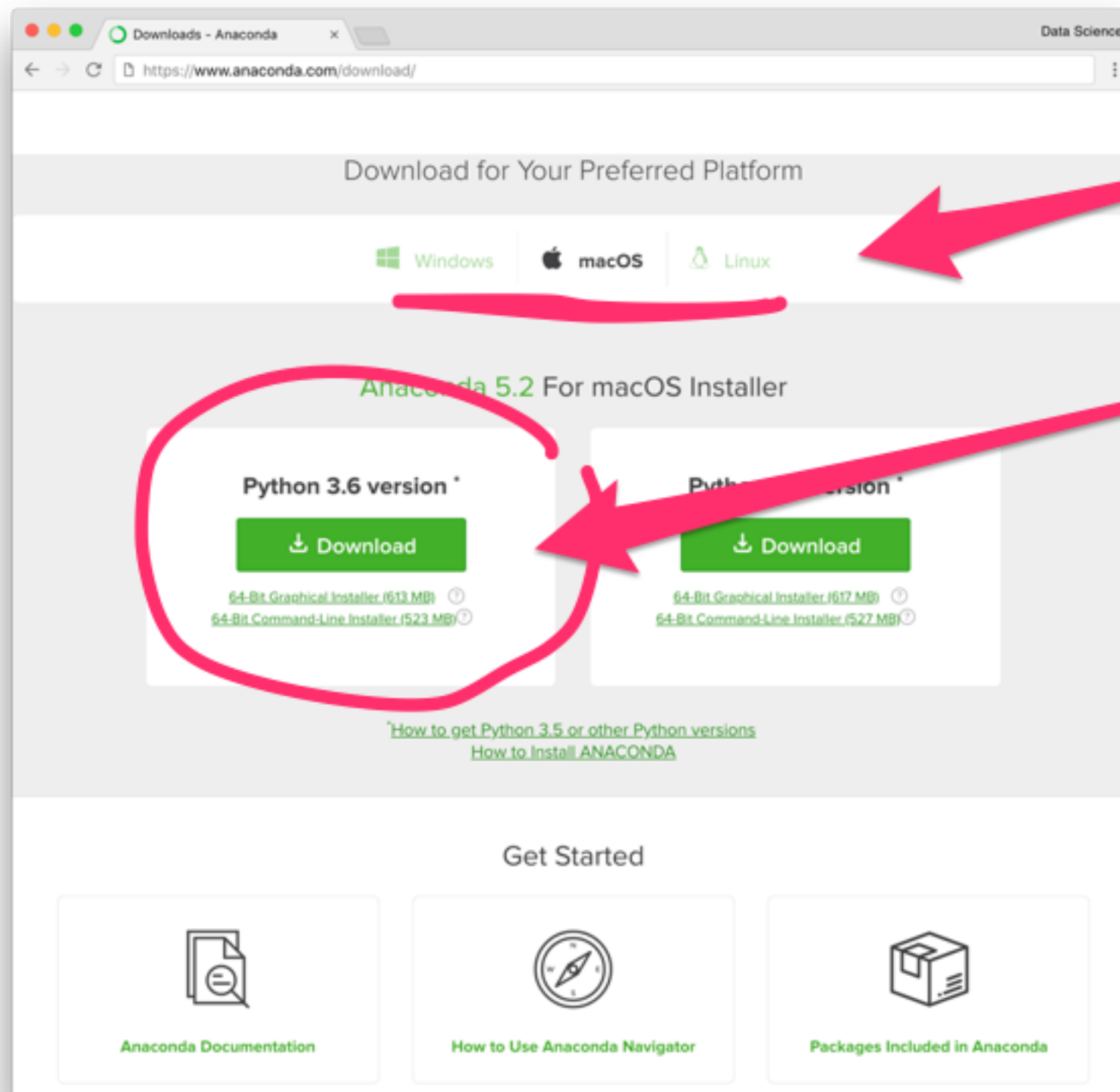
<https://www.anaconda.com/downloads>



Anaconda is a free, cross-platform Python distribution that includes crucial Python libraries such as [Numpy](#) and [Scipy](#)



Where to get Python



Select your platform

Select Python 3.x

(Python 2.x code will not be accepted.)

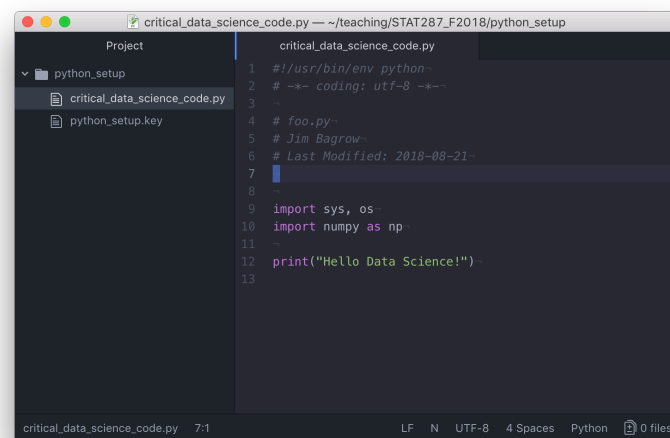
(Macs and Linux machines come with a version of Python already, but it's best to install Anaconda anyway in order to get all the packages set up for you.)

2. Setting up a work environment

Setting up a work environment

Python requires a tool to **create Python code** and a tool to **execute Python code**

Often these are separate apps such as a **text editor** and a **terminal**:

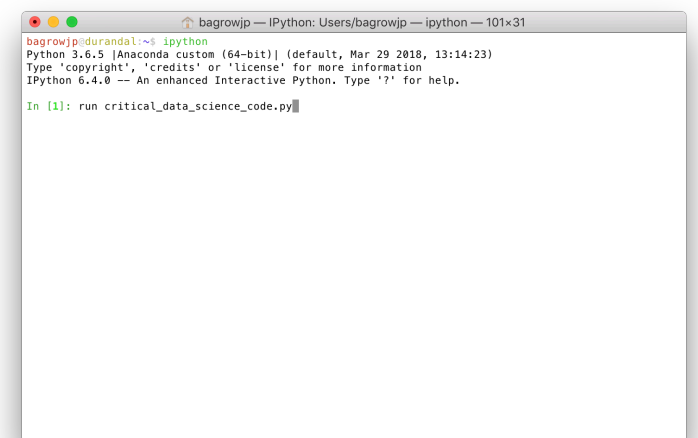
A screenshot of a text editor window titled 'critical_data_science_code.py'. The editor shows a Python script with the following content:

```
1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3
4 # foo.py
5 # Jim Bagrow
6 # Last Modified: 2018-08-21
7
8
9 import sys, os
10 import numpy as np
11
12 print("Hello Data Science!")
13
```

The left sidebar shows a project structure with 'python_setup' containing 'critical_data_science_code.py' and 'python_setup.key'.

text editor

+

A screenshot of a terminal window titled 'bagrowjp — IPython: Users/bagrowjp — ipython — 101x31'. The terminal shows the command 'ipython' being executed, followed by the IPython prompt 'In [1]:'. The user has entered 'run critical_data_science_code.py', which has been executed, resulting in the output 'Hello Data Science!'.

terminal

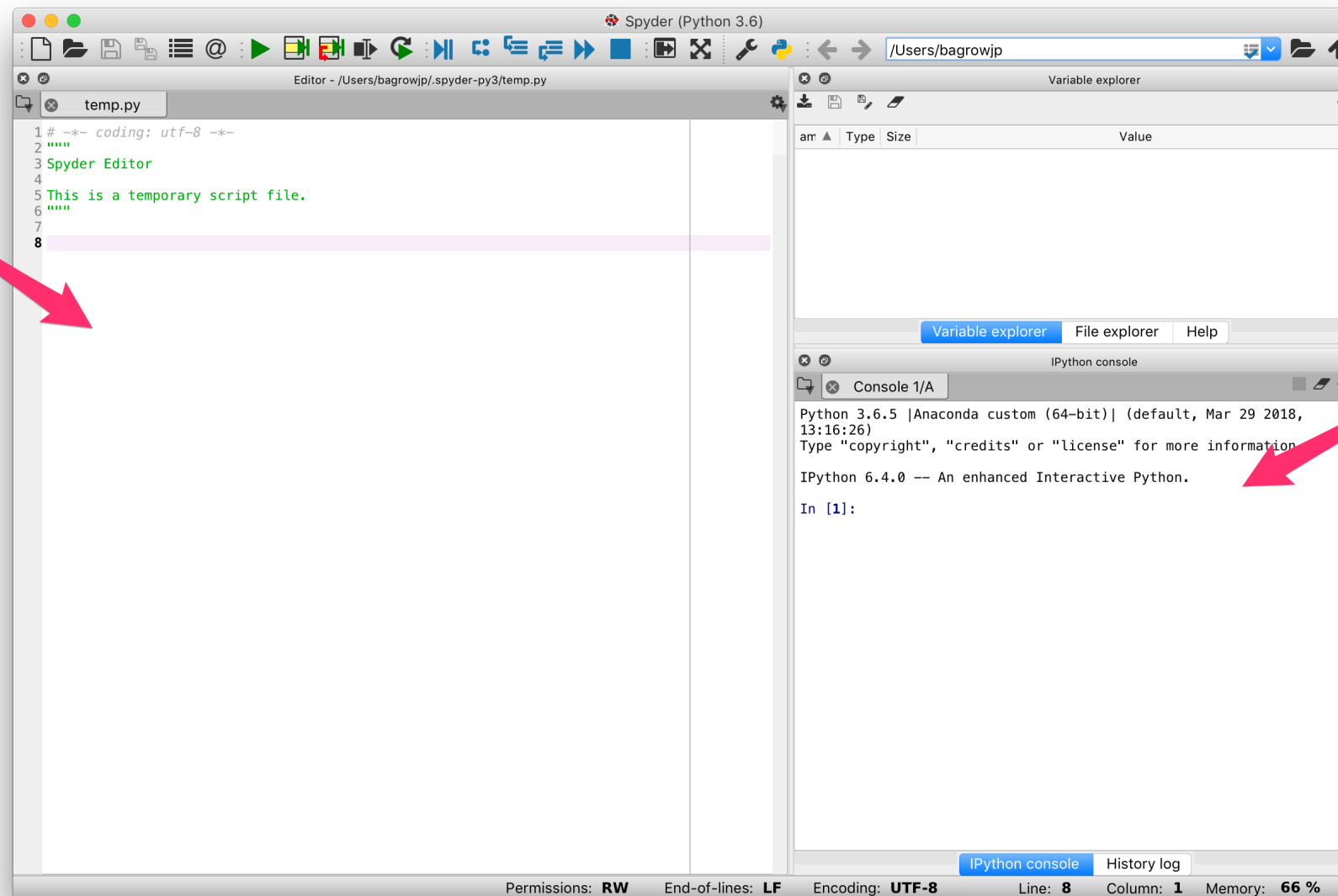
But apps exist that **contain both** 



Spyder - Python work environment

(Scientific PYthon Development EnviRonment)

text editor to
create code



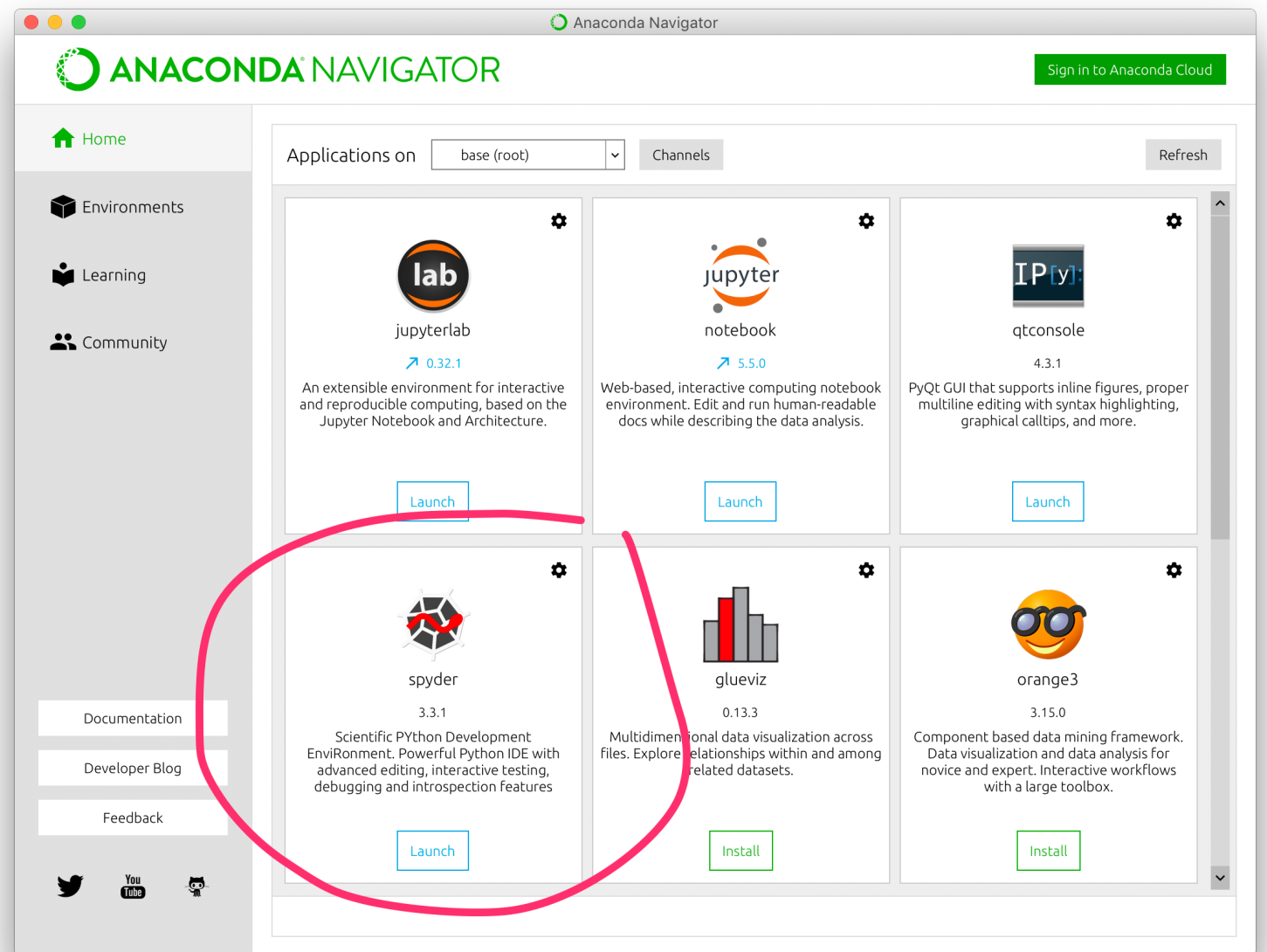
console (terminal)
to run code

You are **STRONGLY encouraged** to do all your work within the free Spyder app. You can set up your own tools, but you are responsible for ensuring they function correctly (see Appendix).

Use Spyder!

Launching Spyder

1. Open the **Anaconda Navigator** app that installs with Anaconda Python
2. Click **Launch** on the Spyder pane

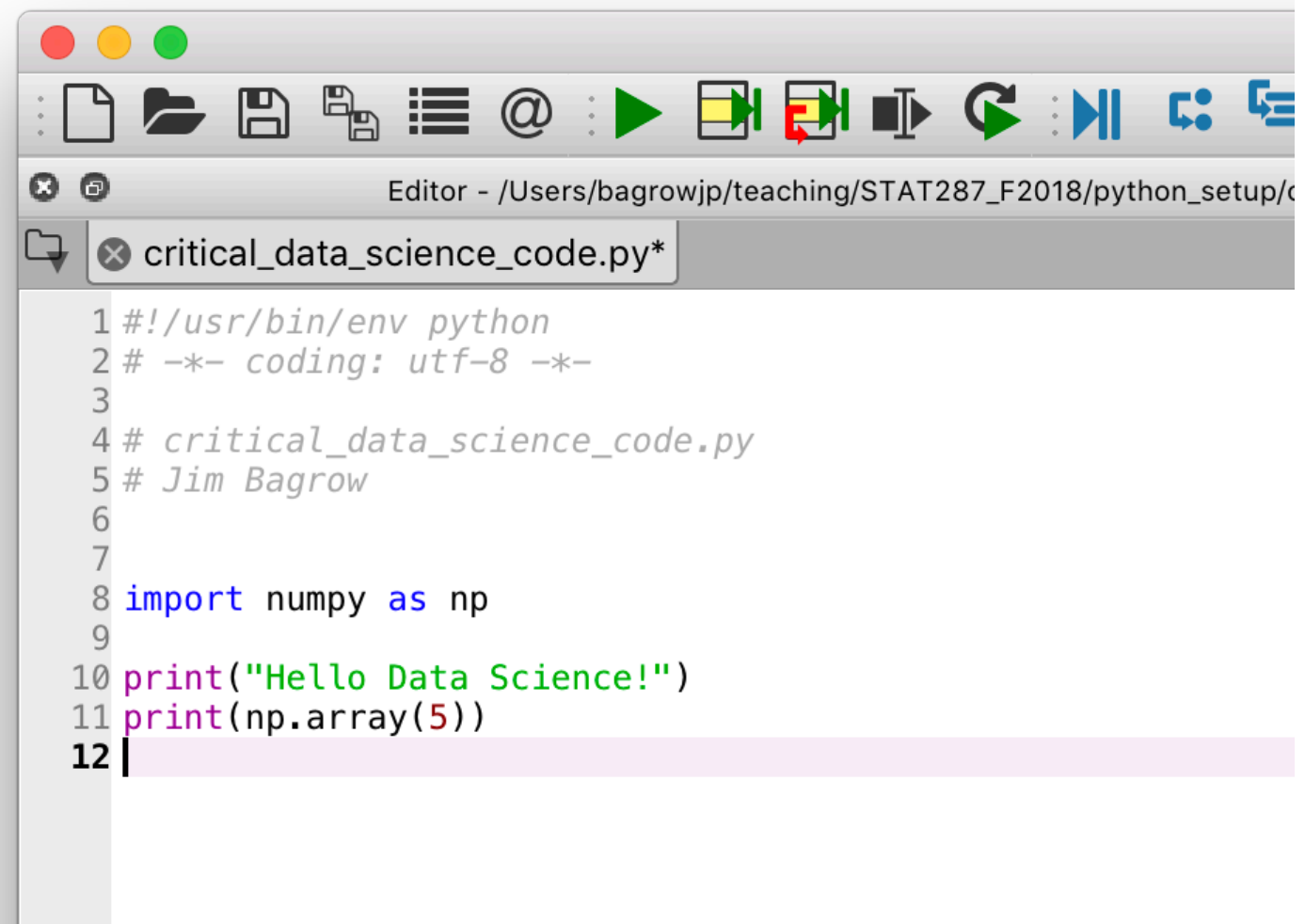


<https://docs.anaconda.com/anaconda/navigator/>

Writing code with Spyder

Write your code in the Spyder editor:

This creates a .py script file



The screenshot shows the Spyder Python IDE interface. The top toolbar contains icons for file operations (new, open, save, save as), editing (undo, redo, cut, copy, paste), and execution (run, step through, step over, step under, interrupt). The editor window title is "Editor - /Users/bagrowjp/teaching/STAT287_F2018/python_setup/c". The file name in the tab is "critical_data_science_code.py*". The code in the editor is as follows:

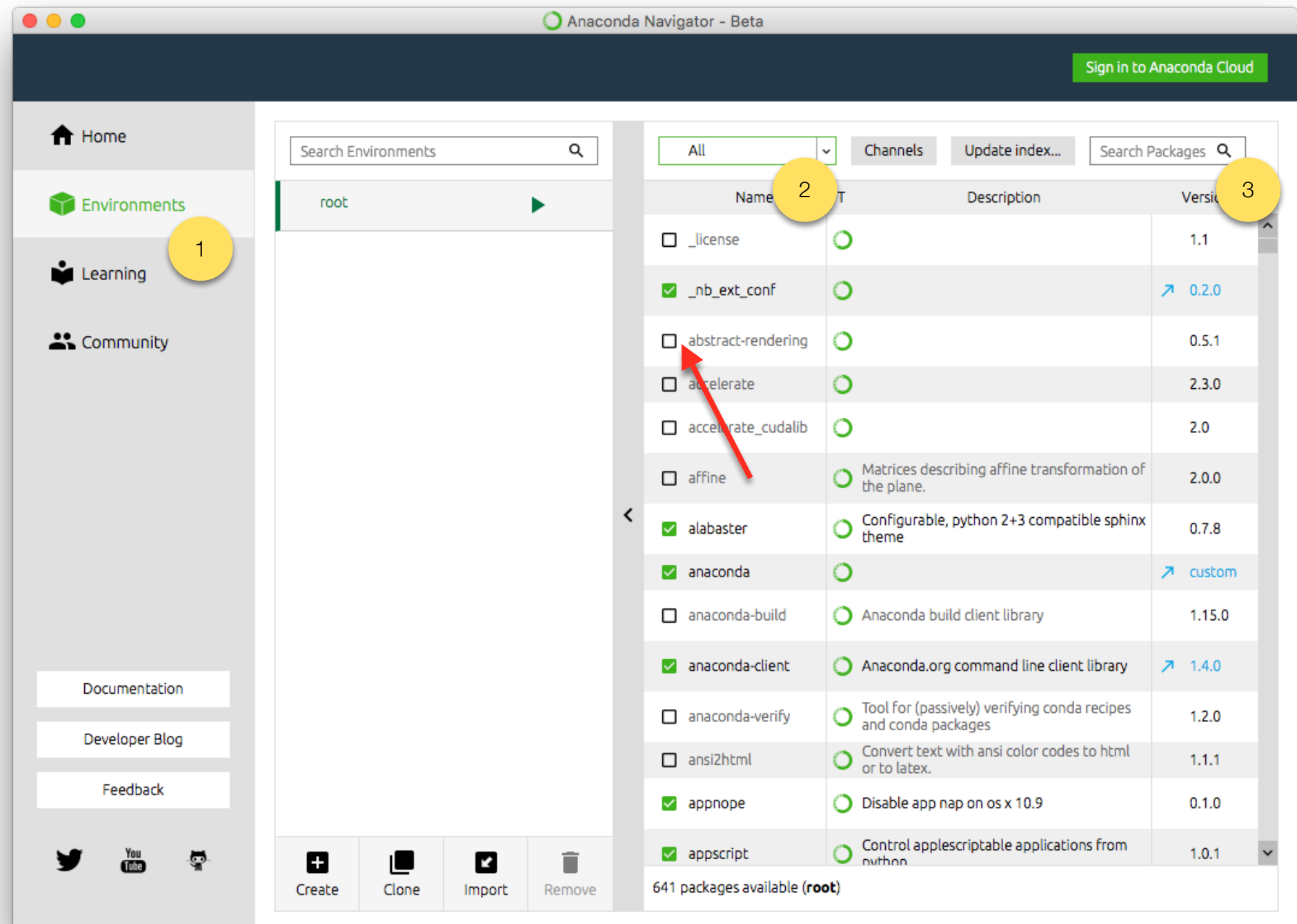
```
1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3
4 # critical_data_science_code.py
5 # Jim Bagrow
6
7
8 import numpy as np
9
10 print("Hello Data Science!")
11 print(np.array(5))
12 |
```

Installing packages with Anaconda

If necessary, to install packages from the **Navigator**:

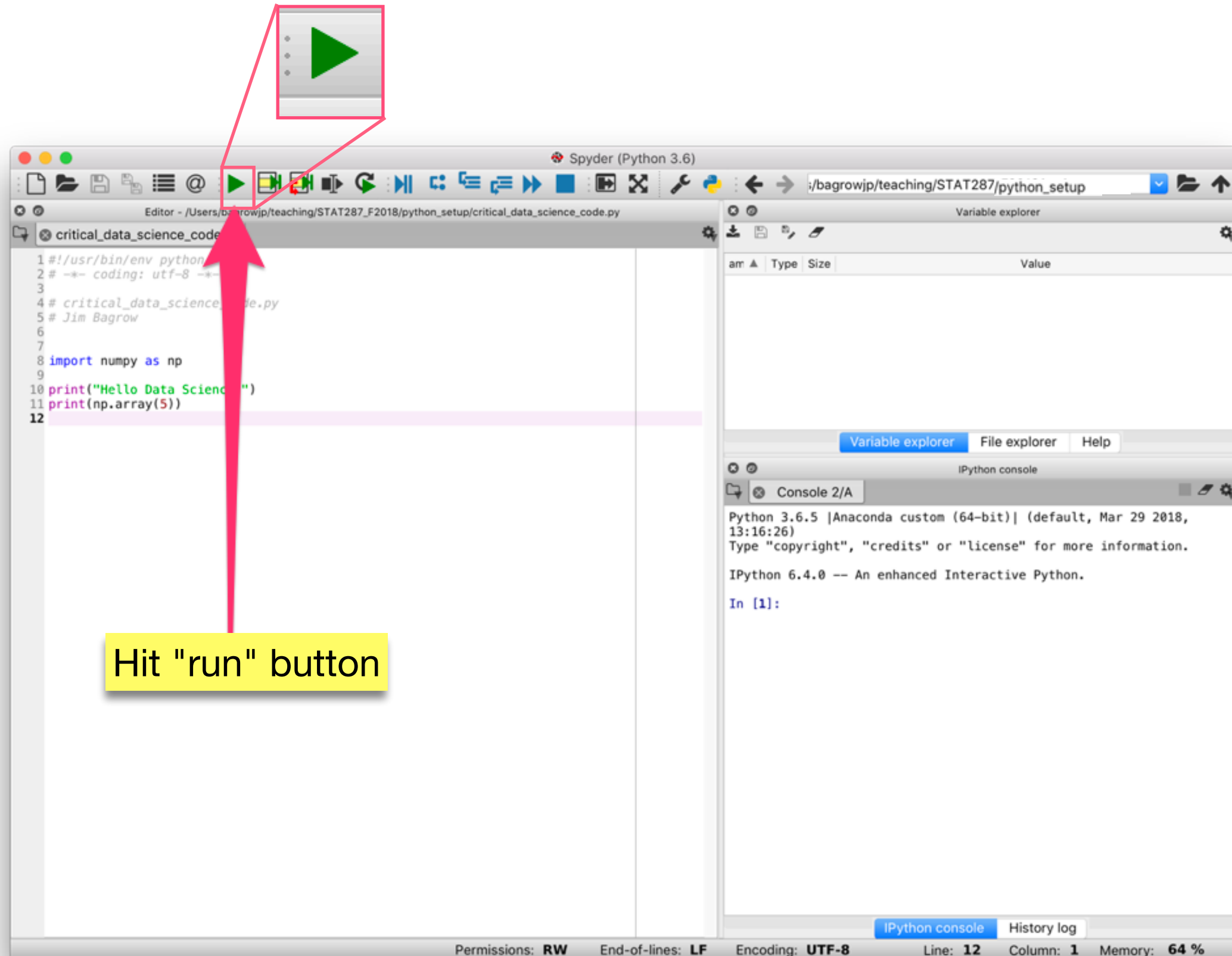
- (1) Go to environments
- (2) Select “All”
- (3) Search for a package
→ Check the package's checkbox

Lastly, click the new **apply** button that appears and follow the resulting dialog box to finish the installation.



3. Running code

Running code with Spyder



Running code with Spyder

The screenshot displays the Spyder Python IDE interface. The main editor window shows a Python script named `critical_data_science_code.py` with the following code:

```
1#!/usr/bin/env python
2# -*- coding: utf-8 -*-
3
4# critical_data_science_code.py
5# Jim Bagrow
6
7
8import numpy as np
9
10print("Hello Data Science!")
11print(np.array(5))
12
```

The IPython console on the right shows the execution of the script, outputting:

```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

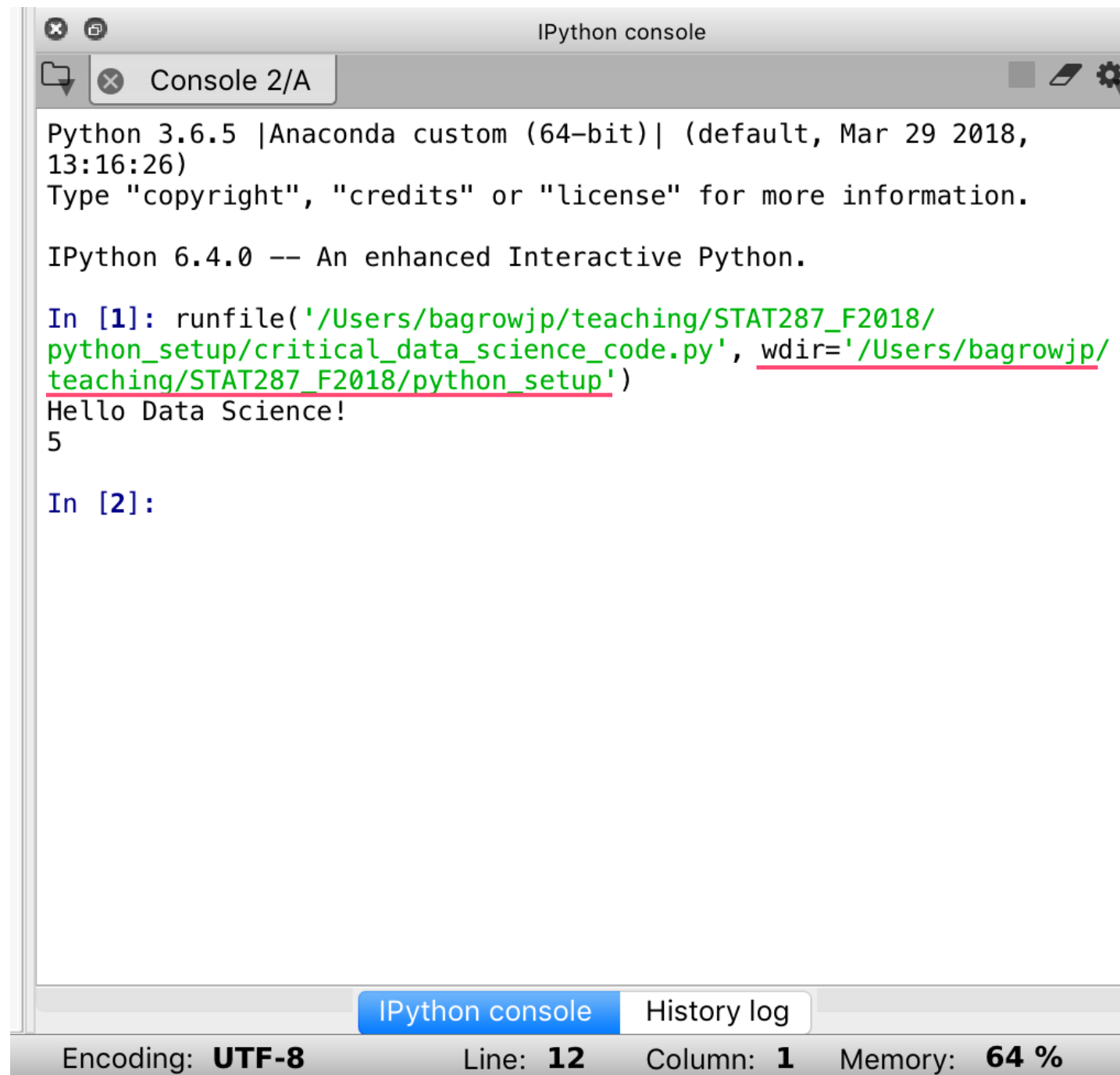
In [1]: runfile('/Users/bagrowjp/teaching/STAT287_F2018/python_setup/critical_data_science_code.py', wdir='/Users/bagrowjp/teaching/STAT287_F2018/python_setup')
Hello Data Science!
5

In [2]:
```

A yellow callout box with a red arrow pointing to the IPython console contains the text: "and any output appears here!".

At the bottom of the window, the status bar shows: Permissions: **RW** End-of-lines: **LF** Encoding: **UTF-8** Line: **12** Column: **1** Memory: **64 %**

Running code with Spyder




```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/bagrowjp/teaching/STAT287_F2018/python_setup/critical_data_science_code.py', wdir='/Users/bagrowjp/teaching/STAT287_F2018/python_setup')
Hello Data Science!
5

In [2]:
```

The screenshot shows the Spyder IPython console window. The title bar reads 'IPython console'. The console output shows the Python version (3.6.5), Anaconda version (64-bit), and IPython version (6.4.0). The first input prompt 'In [1]:' shows a call to `runfile()` with two arguments: a file path and a `wdir` parameter. The file path is `'/Users/bagrowjp/teaching/STAT287_F2018/python_setup/critical_data_science_code.py'` and the `wdir` parameter is `'/Users/bagrowjp/teaching/STAT287_F2018/python_setup'`. The output of the script is 'Hello Data Science!' followed by the number '5'. The second input prompt 'In [2]:' is shown. The status bar at the bottom indicates 'Encoding: UTF-8', 'Line: 12', 'Column: 1', and 'Memory: 64 %'.

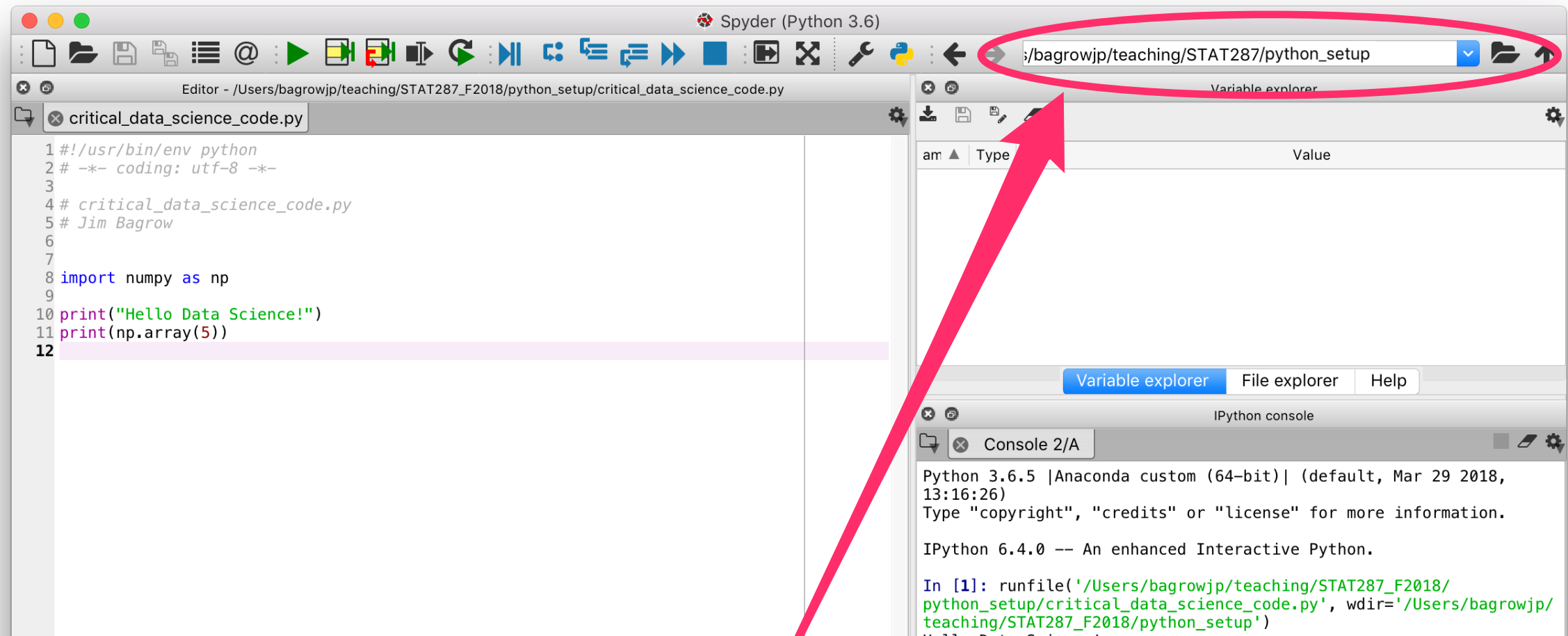


`wdir` = "working directory" the location where the code runs.

Make sure this is the location where you saved your .py script (this should be the **enclosing folder** of any assignments)

Any files you read or write inside your .py script must have locations specified *relative* to `wdir`

Running code with Spyder

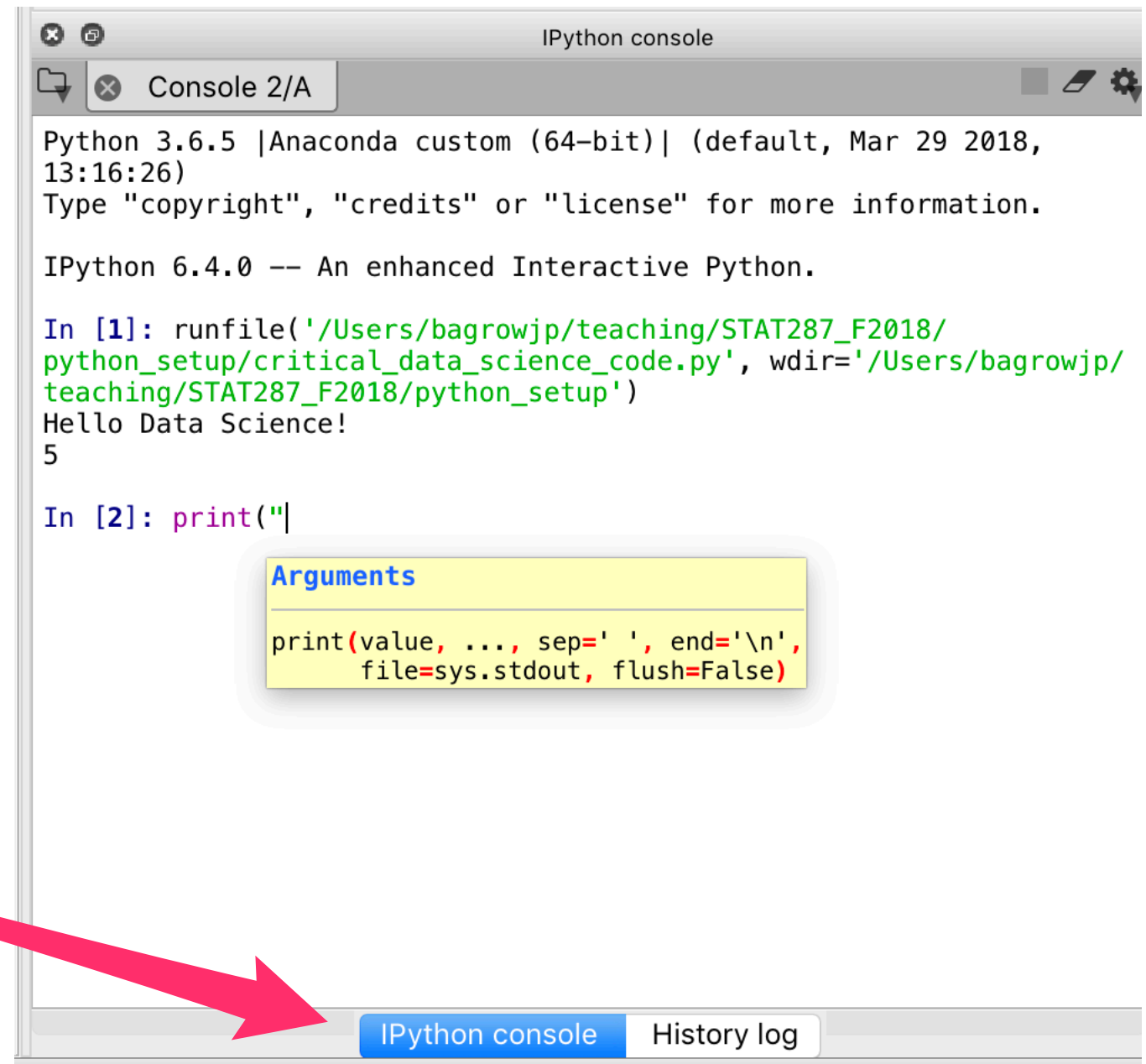


The current working directory is also listed here

Interactive Python

The console can also run **individual python commands** typed directly into it

This console is running IPython, a very handy command line tool for using Python *interactively*!



```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/bagrowjp/teaching/STAT287_F2018/python_setup/critical_data_science_code.py', wdir='/Users/bagrowjp/teaching/STAT287_F2018/python_setup')
Hello Data Science!
5

In [2]: print("|
```

Arguments

```
print(value, ..., sep=' ', end='\n',
      file=sys.stdout, flush=False)
```

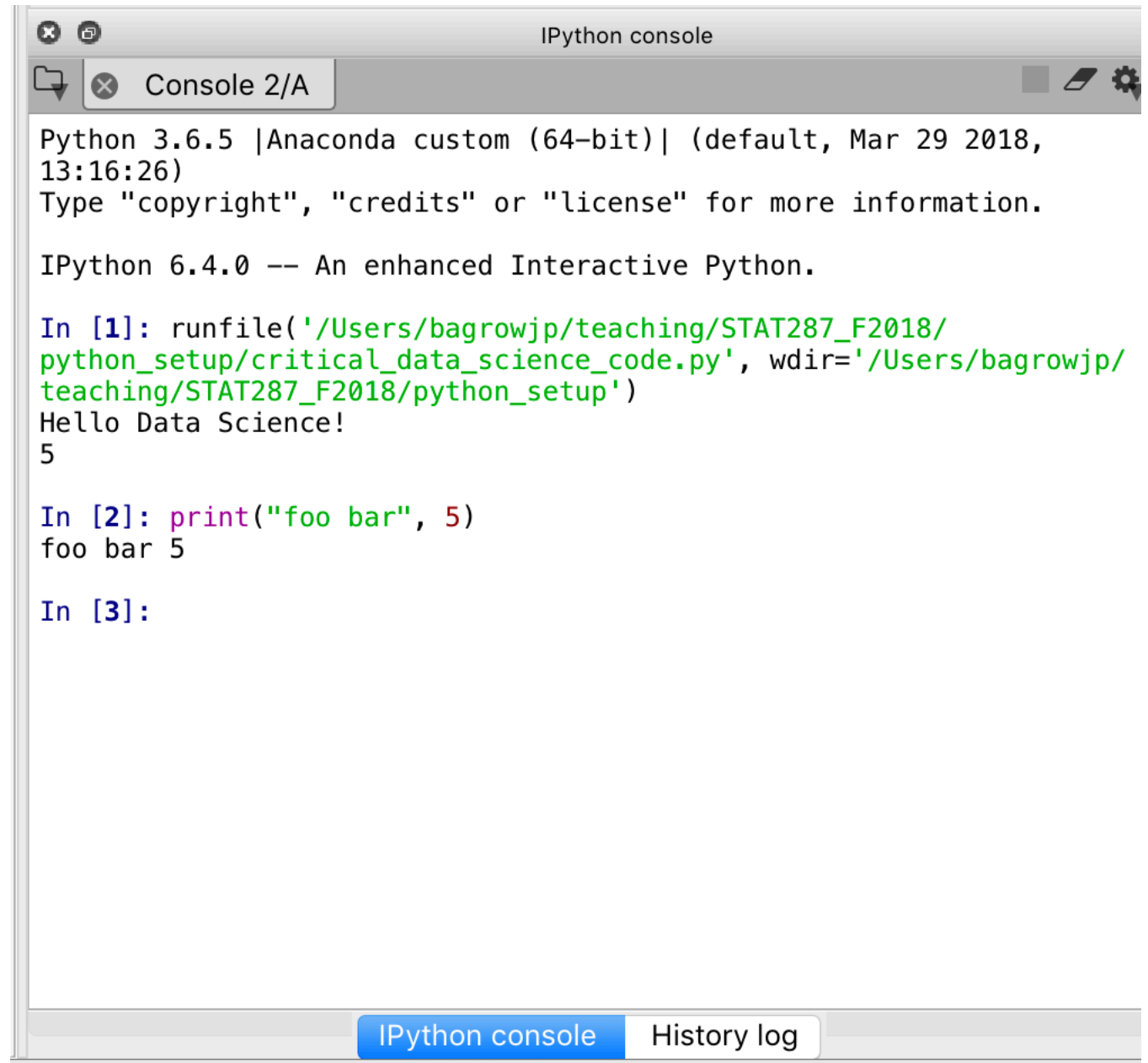
IPython console History log

Interactive Python

The console can also run **individual python commands** typed directly into it

Just type your command and press **enter/return** to run it

Running small snippets of your code interactively and examining the results is one of the **most important techniques** you have for writing correct code, especially as you build longer scripts



```
IPython console
Console 2/A
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/bagrowjp/teaching/STAT287_F2018/python_setup/critical_data_science_code.py', wdir='/Users/bagrowjp/teaching/STAT287_F2018/python_setup')
Hello Data Science!
5

In [2]: print("foo bar", 5)
foo bar 5

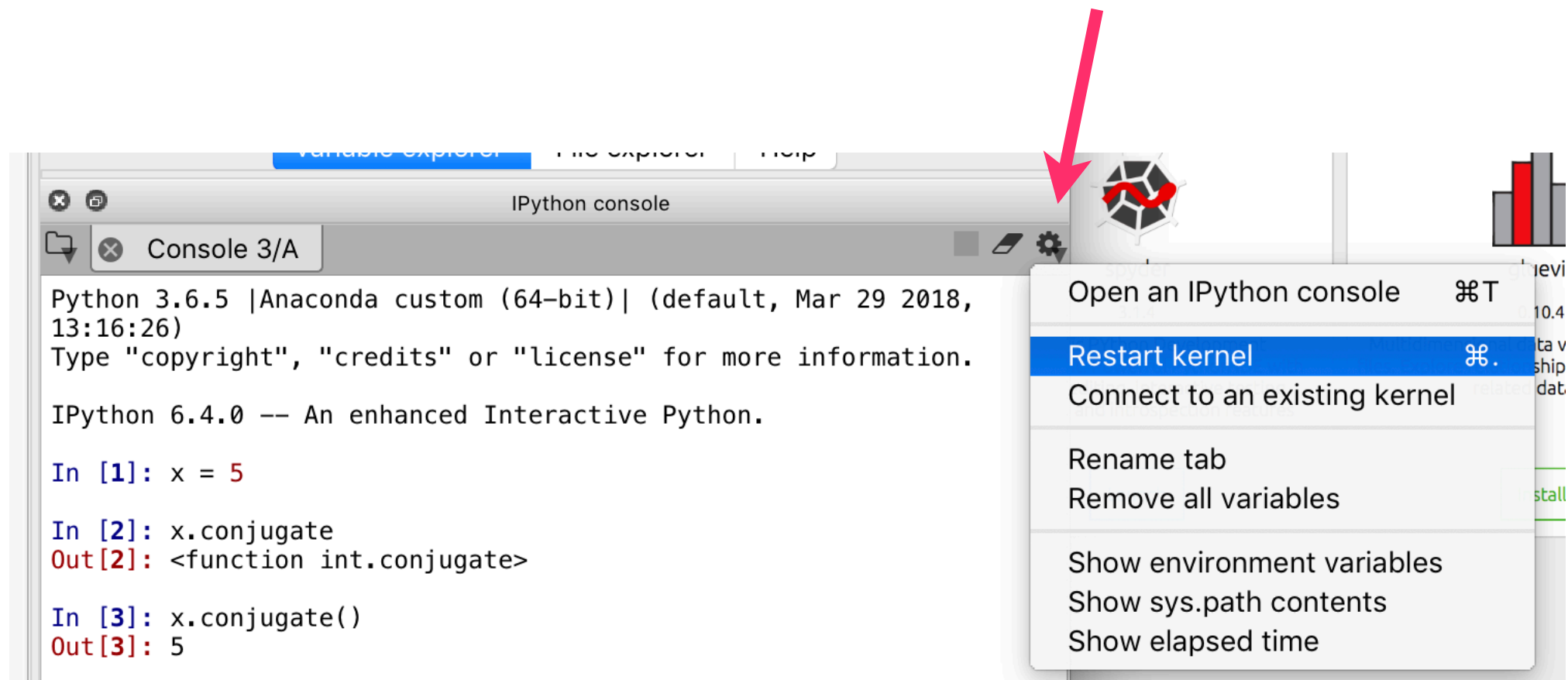
In [3]:
```

IPython console History log

Side note: killing the console

Sometimes, you need to **restart Python**, such as if you have created an endless loop

The "**gear**" button in the upper right of the console lets you do this, just click and select "Restart kernel"

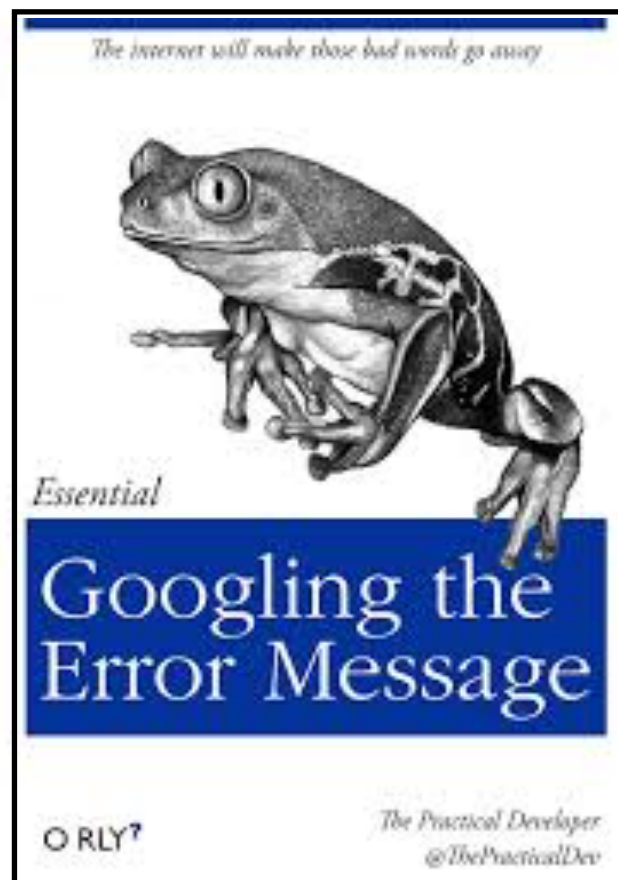


4. Getting help

(besides office hours, asking friends, etc.)

Getting help

Many online tutorials exist for Python, and websites such as Stack Overflow have tons of answered questions. *Googling the error message* is also helpful!



- Put only the **relevant part** of the error message into Google
- Remove **local information**, like the name of your laptop (for example)
- Use quotes "" to search for **exact strings** as needed

Getting help

Many online tutorials exist for Python, and websites such as Stack Overflow have tons of answered questions. *Googling the error message* is also helpful!

The IPython console provides documentation access as well:

```

Variable explorer  File explorer  Help
IPython console
Console 2/A
IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/bagrowjp/teaching/STAT287_F2018/
python_setup/critical_data_science_code.py', wdir='/Users/bagrowjp/
teaching/STAT287_F2018/python_setup')
Hello Data Science!
5

In [2]: print("foo bar", 5)
foo bar 5

In [3]: print?
Docstring:
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.
Optional keyword arguments:
file: a file-like object (stream); defaults to the current
sys.stdout.
sep:   string inserted between values, default a space.
end:   string appended after the last value, default a newline.
flush: whether to forcibly flush the stream.
Type:  builtin_function_or_method

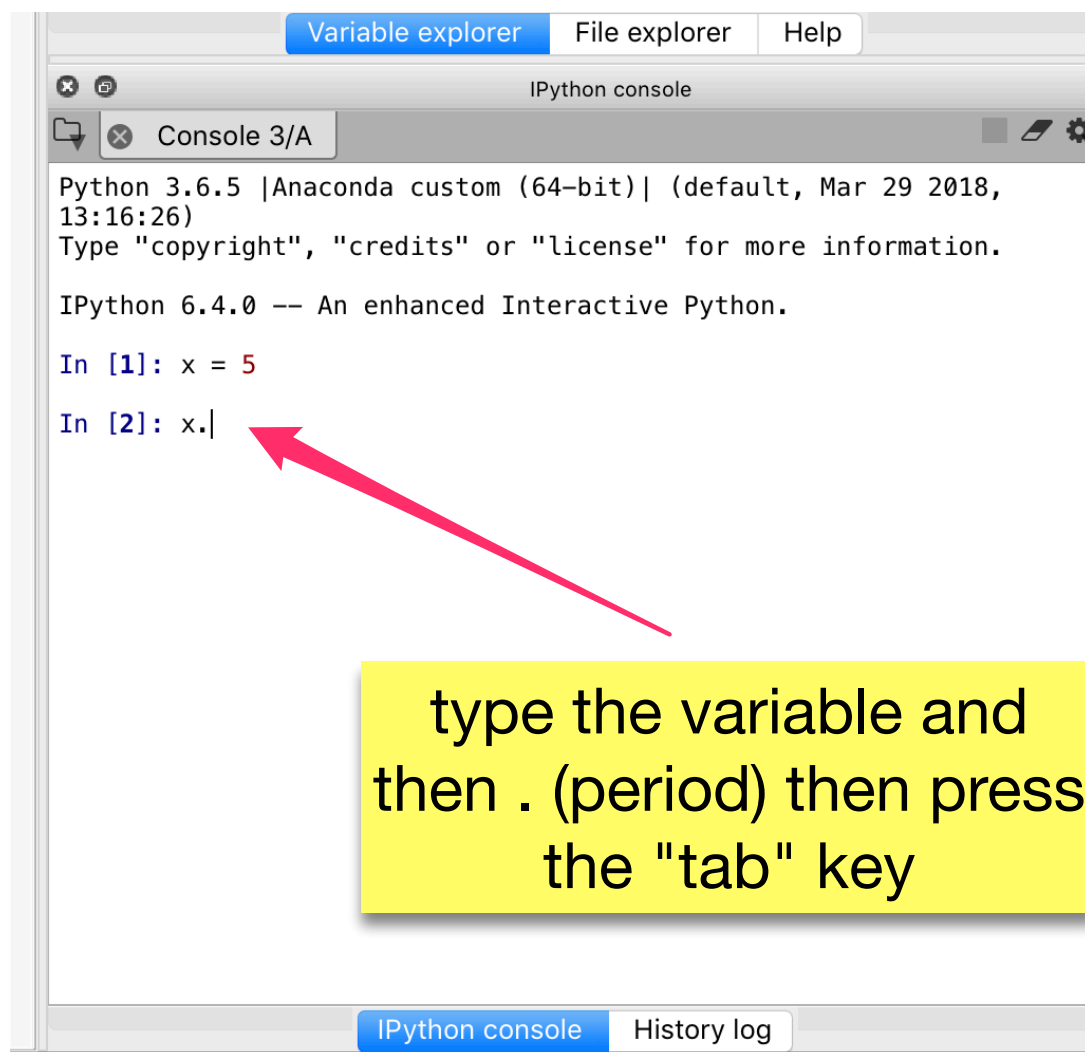
In [4]:

```

Just put a "?" after a variable, method, or function (ex: `print?`) and the docstring will be displayed

Getting help

IPython also helps you explore new code with **TAB COMPLETION**

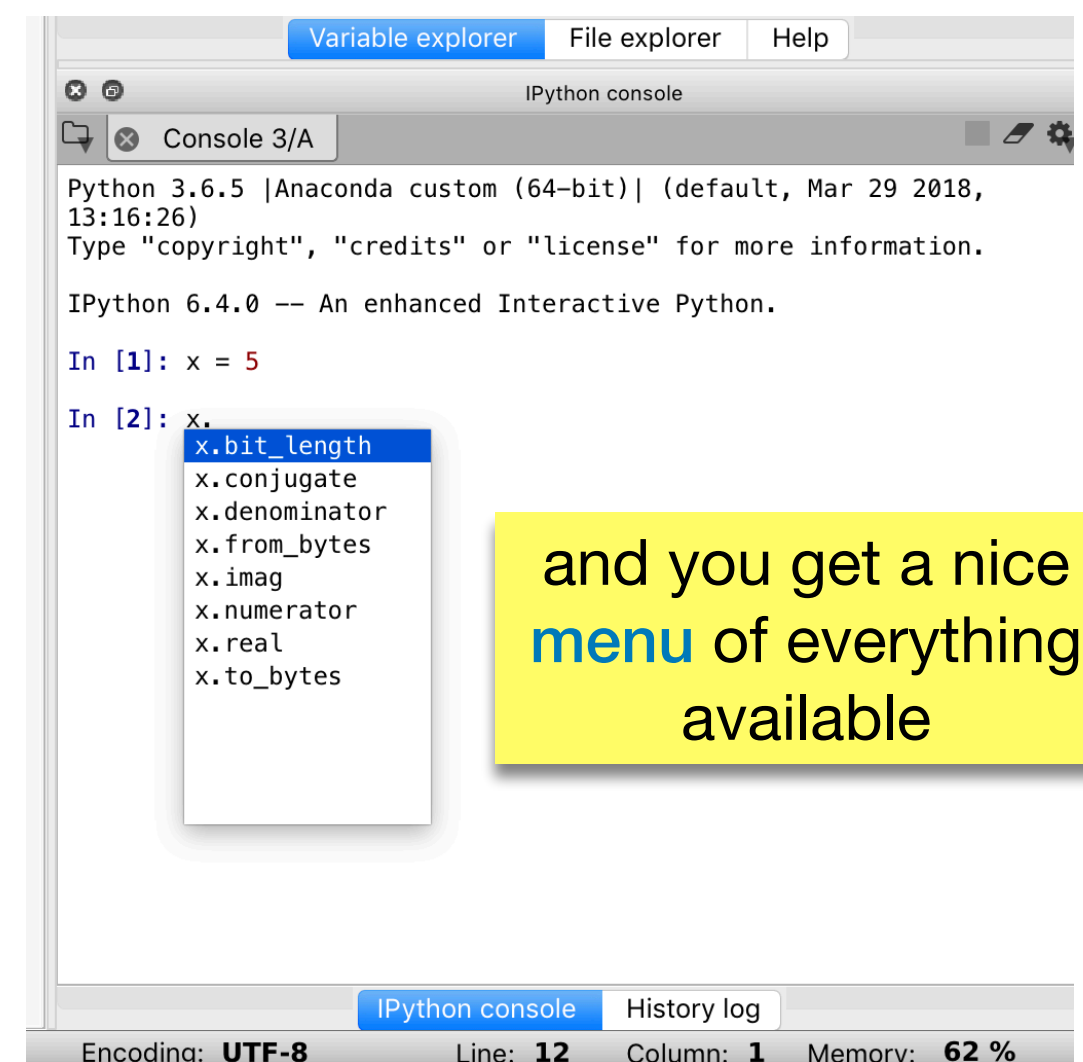


The screenshot shows an IPython console window with tabs for 'Variable explorer', 'File explorer', and 'Help'. The console output includes the Python version (3.6.5), Anaconda version (64-bit), and IPython version (6.4.0). The user has entered 'In [1]: x = 5' and 'In [2]: x.' with a pink arrow pointing to the period. A yellow box at the bottom contains the text: 'type the variable and then . (period) then press the "tab" key'.

```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: x = 5
In [2]: x.
```



The screenshot shows the same IPython console window after pressing the 'tab' key. A dropdown menu is visible, listing attributes: 'x.bit_length', 'x.conjugate', 'x.denominator', 'x.from_bytes', 'x.imag', 'x.numerator', 'x.real', and 'x.to_bytes'. A yellow box to the right contains the text: 'and you get a nice menu of everything available'. The status bar at the bottom shows 'Encoding: UTF-8', 'Line: 12', 'Column: 1', and 'Memory: 62 %'.

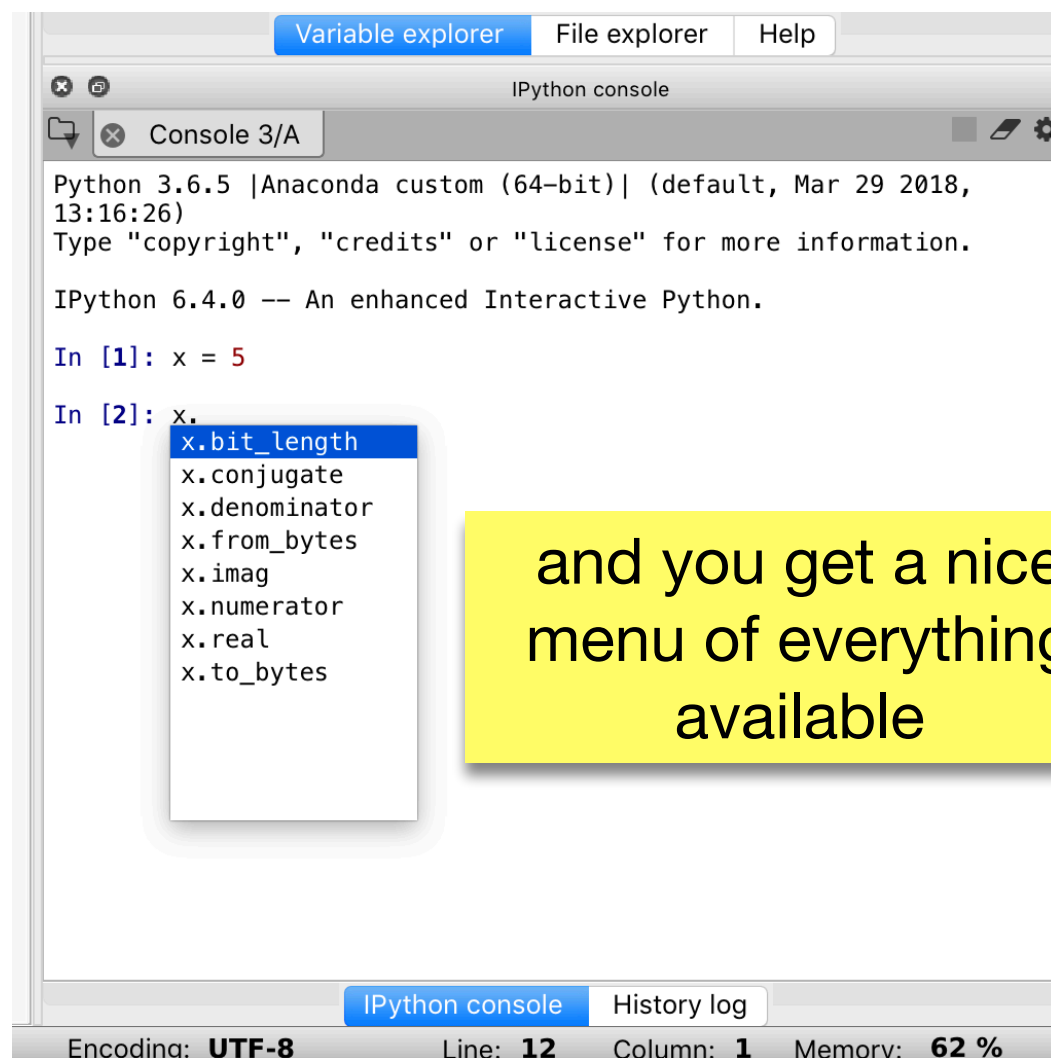
```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: x = 5
In [2]: x.
x.bit_length
x.conjugate
x.denominator
x.from_bytes
x.imag
x.numerator
x.real
x.to_bytes
```

Getting help

IPython also helps you explore new code with **TAB COMPLETION**



The screenshot shows the IPython console interface. At the top, there are tabs for 'Variable explorer', 'File explorer', and 'Help'. Below these is a tab for 'Console 3/A'. The console displays the following text:

```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

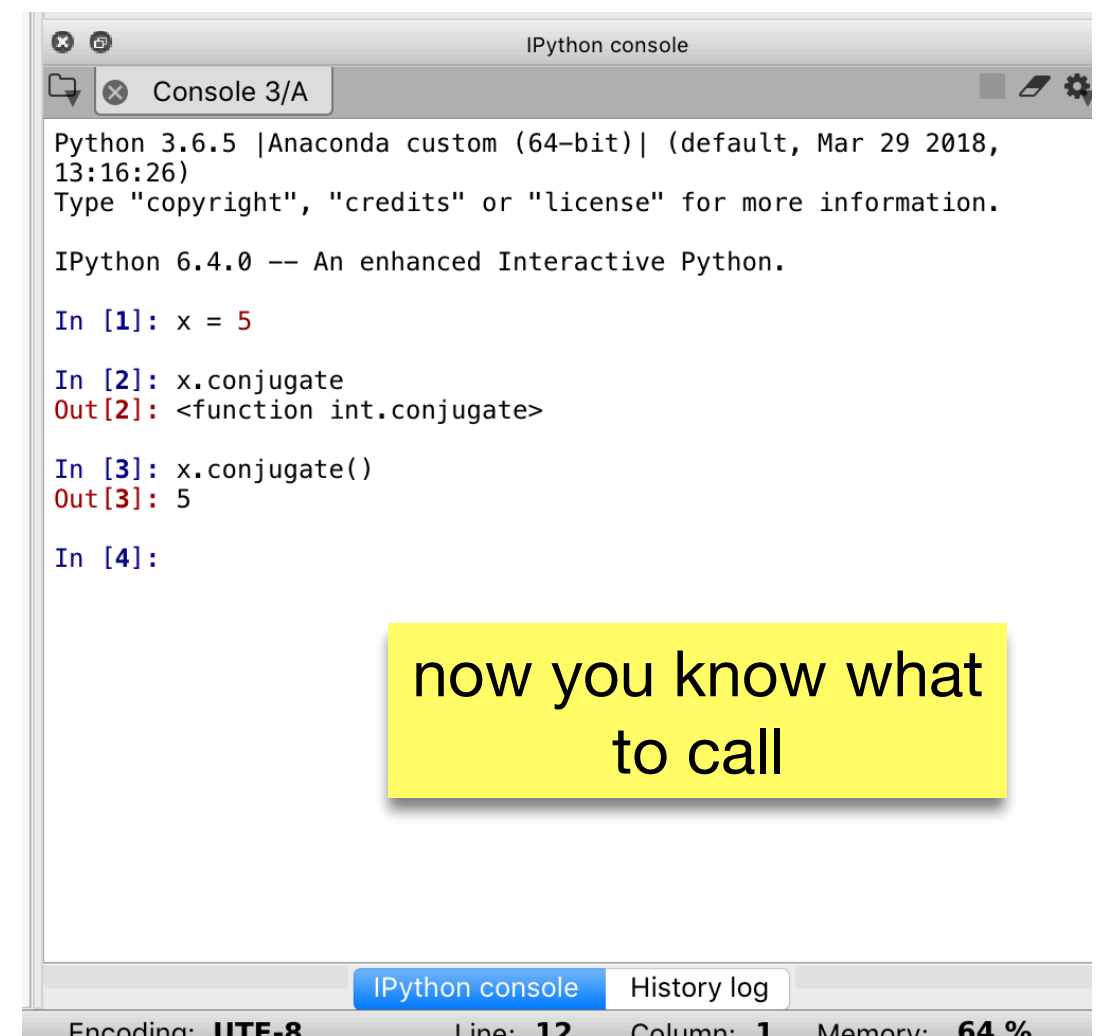
In [1]: x = 5
In [2]: x.
```

A dropdown menu is visible below the prompt 'In [2]: x.', showing a list of attributes and methods for the integer object 'x':

- x.bit_length
- x.conjugate
- x.denominator
- x.from_bytes
- x.imag
- x.numerator
- x.real
- x.to_bytes

A yellow callout box with the text 'and you get a nice menu of everything available' points to this dropdown menu.

At the bottom of the console window, there are tabs for 'IPython console' and 'History log'. The status bar at the very bottom shows 'Encoding: UTF-8', 'Line: 12', 'Column: 1', and 'Memory: 62 %'.



The screenshot shows the IPython console interface. At the top, there are tabs for 'Console 3/A' and 'Help'. The console displays the following text:

```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: x = 5
In [2]: x.conjugate
Out[2]: <function int.conjugate>

In [3]: x.conjugate()
Out[3]: 5

In [4]:
```

A yellow callout box with the text 'now you know what to call' points to the output of the third command, '5'.

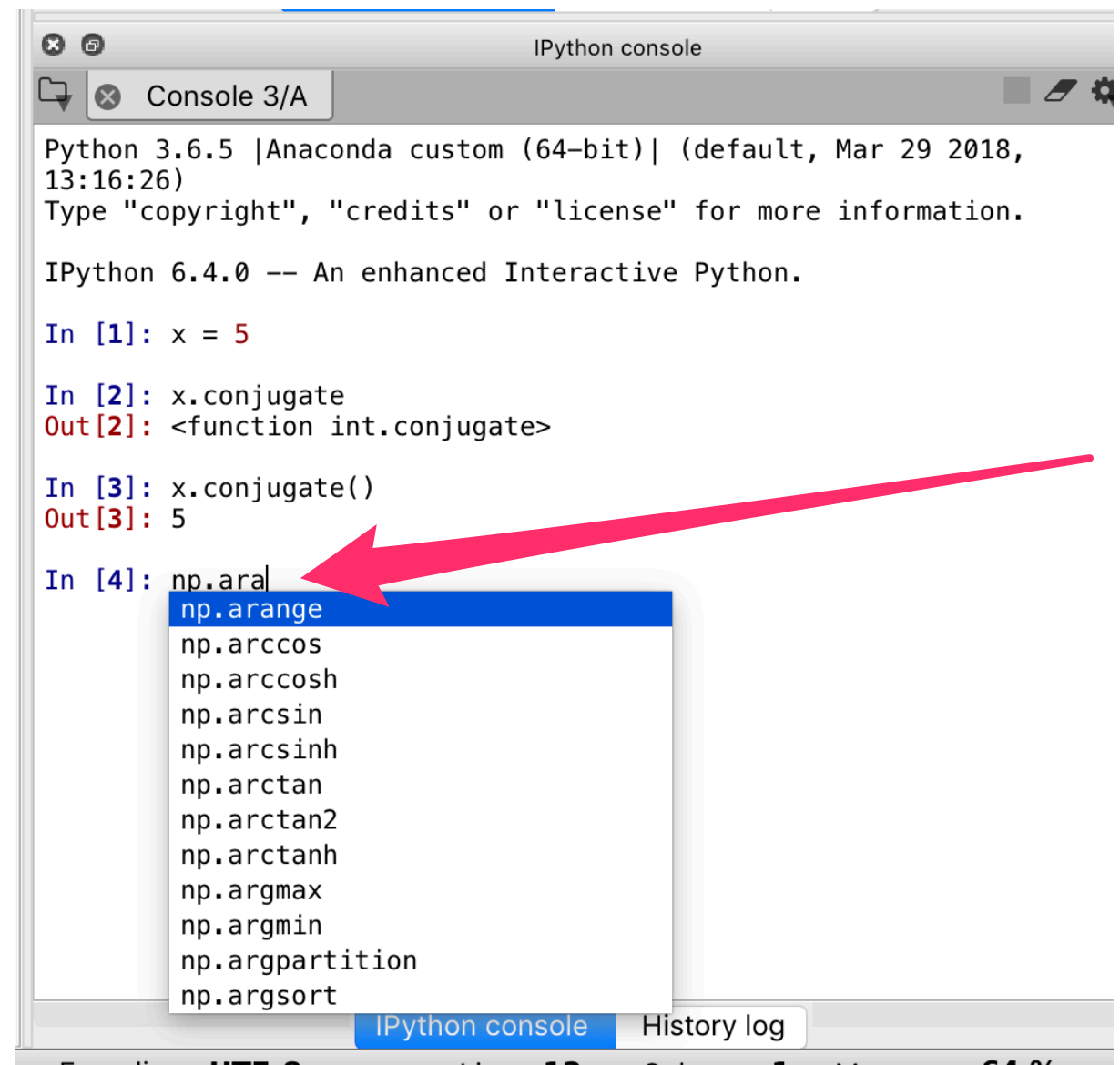
At the bottom of the console window, there are tabs for 'IPython console' and 'History log'. The status bar at the very bottom shows 'Encoding: UTF-8', 'Line: 12', 'Column: 1', and 'Memory: 64 %'.

Getting help

IPython also helps you explore new code with **TAB COMPLETION**

Sometimes there are **very big completion menus**—narrow your search by typing the **first few letters** of what you are looking for, either before or after you press "tab", and you will jump right there:

And: press "tab" a second time to put in the selected completion automatically (saves typing time for long names)



```
Python 3.6.5 |Anaconda custom (64-bit)| (default, Mar 29 2018, 13:16:26)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

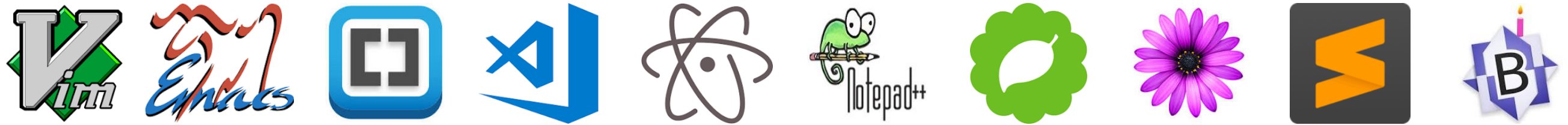
In [1]: x = 5

In [2]: x.conjugate
Out[2]: <function int.conjugate>

In [3]: x.conjugate()
Out[3]: 5

In [4]: np.aral
np.arange
np.arccos
np.arccosh
np.arcsin
np.arcsinh
np.arctan
np.arctan2
np.arctanh
np.argmax
np.argmin
np.argpartition
np.argsort
```

Appendix: set up your source code editor



If you really want to "roll your own" work environment for the course, **and have heeded the warning about being on your own to support it**, then you need to **configure your text editor** when working with Python code you will submit for grading.

Whitespace is a [significant component of Python source code](#), with indentation defining blocks. One can indent a line of code by inserting spaces (using space bar) or tabs (using tab key). Python 3 will not allow mixtures of tabs and spaces.

—Required—

For all code in this course, configure your text editor so that **pressing the tab key once is exactly the same as pressing the spacebar four times**.

- Using spaces for tabs is known as "hard tabs"
- How to set up 4-space hard tabs depends on what editor you are using, which *you will need to figure out*
- Submit only code with 4-space indents, do not submit code containing tab characters

Confused? Don't worry, just use Spyder—it's all set up!