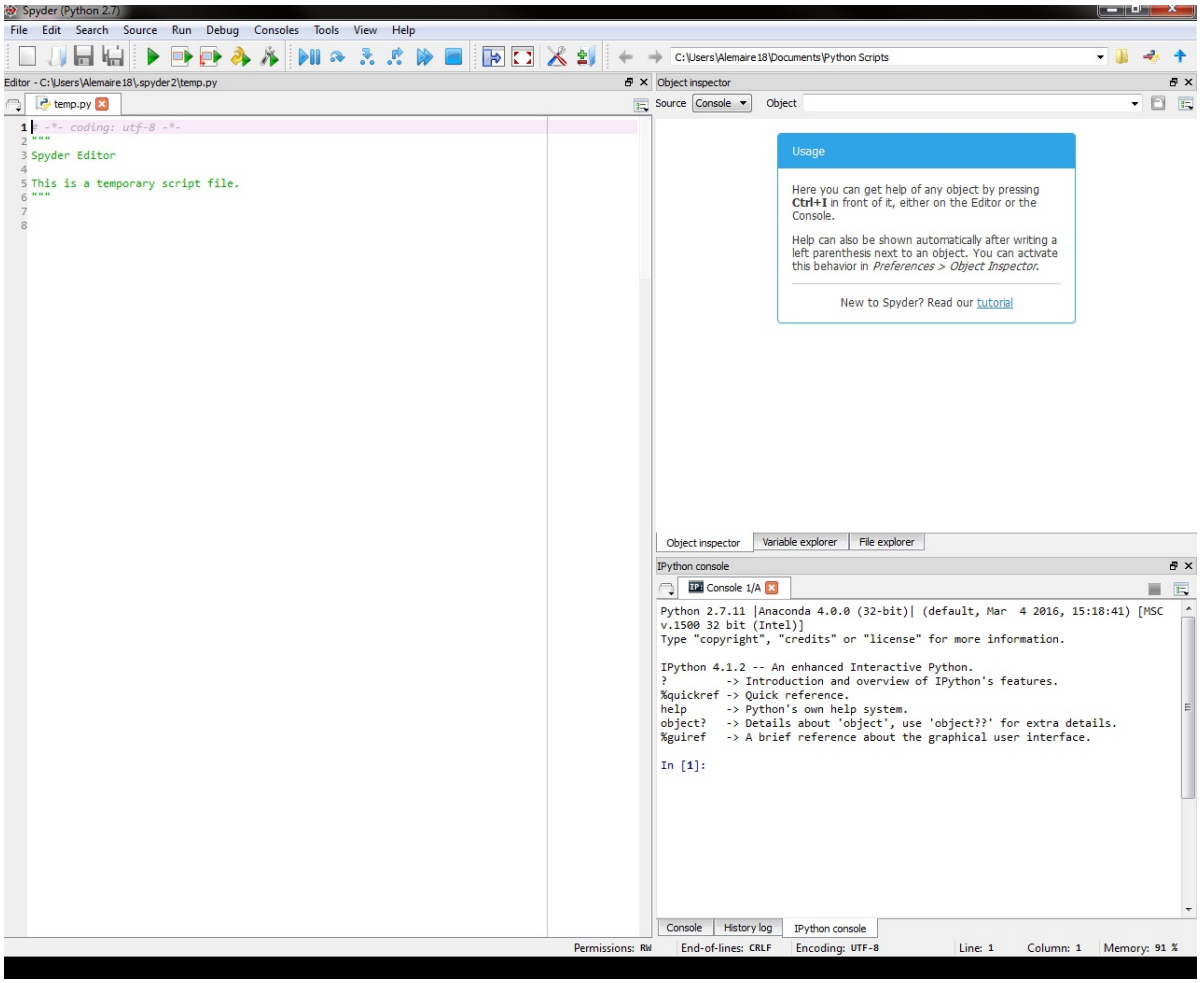
Introduction to python 2018

In this tutorial, I'll go over the basic steps to install python anaconda. This steps will be crucial and can save you some time.

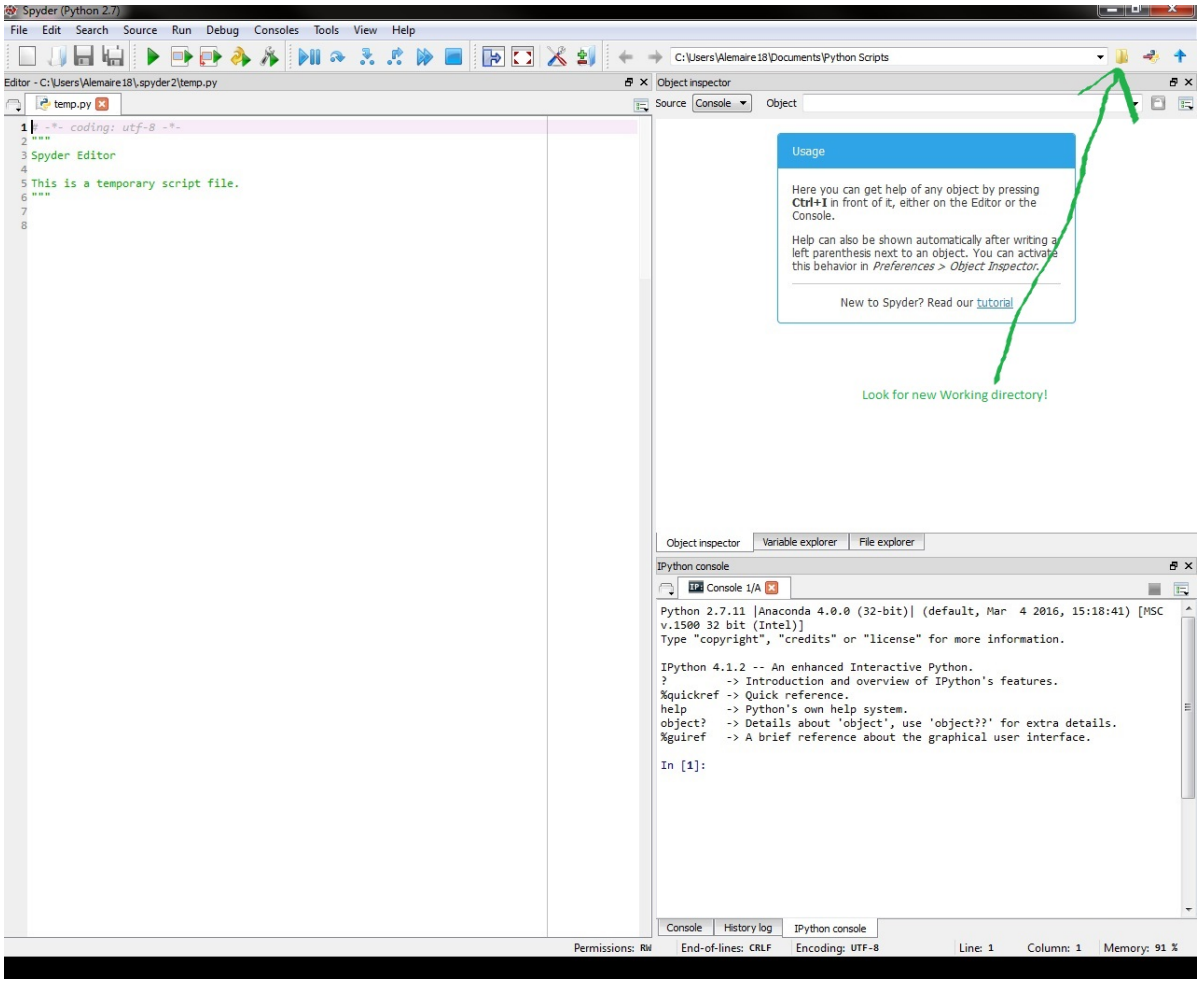
To do this homework, you need to download python anaconda 2.7 using the following link https://www.continuum.io/downloads (https://www.continuum.io/downloads) . Installing this process may take some time.

Once this process is completed you'll be looking for the program spyder in your computer. Although this process might be different for Mac a quick search for anaconda navigator or spyder should bring up to the following prompt.

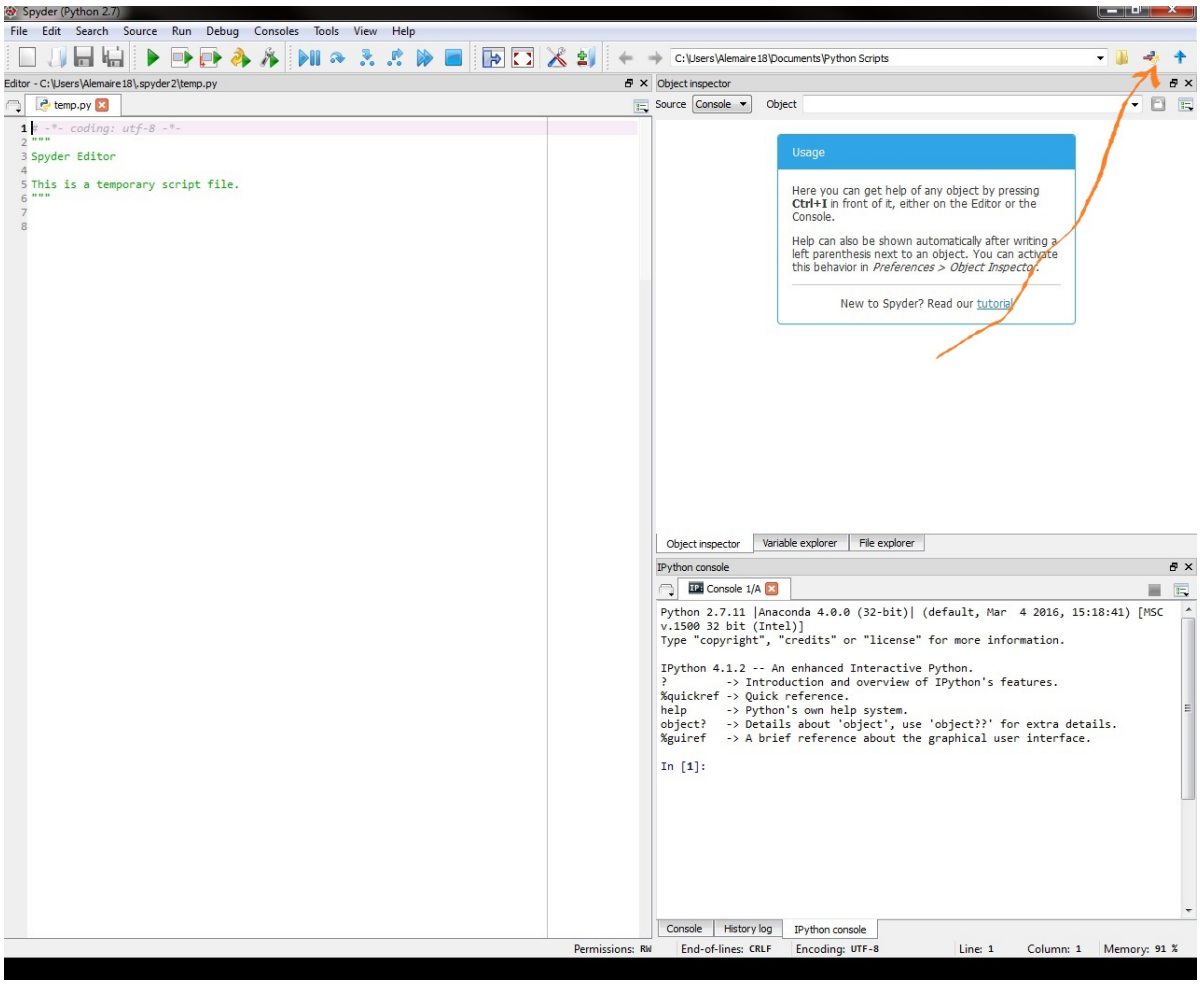


Changing Working Directory

In order to read the files for the homework, you'll need to save all the files given to you(yelp.stop, yelp_homework.csv, etc.) in a directory. To change your working directory you need to change the address bar in the top right of your page by clicking on the "yellow folder" and go to the location you saved all the files provided. you are looking at the green arrow



Once you have found the folder where you put all your files, you'll need to set it as your current directory to do so you'll need to now click in the python icon (the blue and yellow icon) next to the yellow folder. See the orange arrow.


Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Tools View Help
C:\Users\Alemaire18\Documents\Python Scripts
Editor - C:\Users\Alemaire18\.spyder2\temp.py
temp.py
-*- coding: utf-8 -*-
"""
Spyder Editor
This is a temporary script file.
"""
Object inspector
Source Console Object
Usage
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.
Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Object Inspector.
New to Spyder? Read our tutorial
Object inspector Variable explorer File explorer
IPython console
Console 1/A
Python 2.7.11 |Anaconda 4.0.0 (32-bit)| (default, Mar 4 2016, 15:18:41) [MSC v.1500 32 bit (Intel)]
Type "copyright", "credits" or "license" for more information.
IPython 4.1.2 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.
%guiref -> A brief reference about the graphical user interface.
In [1]:
Console History log IPython console
Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 1 Column: 1 Memory: 91 %

Success?

To confirm that you have succeeded, then click on the "File Explorer" to the right of the screen. If you see the files that are in your directory then you can proceed.

Getting Started with Python Spyder

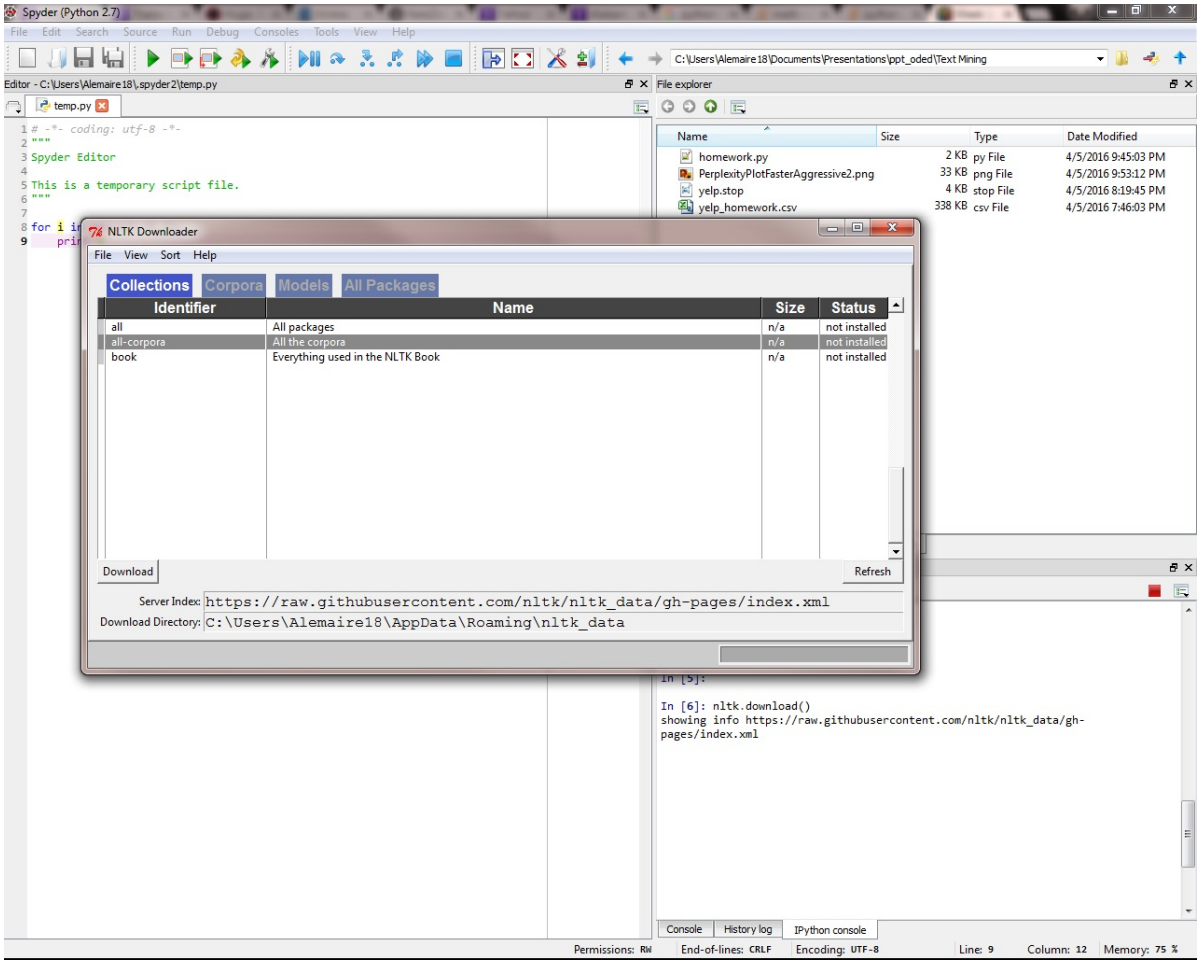
Python spyder should be similar to R-studio at this point. The console should be at the bottom right of the screen with the text editor to the right.

Installing stopwords

Before trying all the codes in the text_analysis.html supplement you need to download the nltk stopwords corpus. This corpora, body of text, contains all off-the-shell stopwords that you need to go over the supplement. Without this you'll get an error when you input the command **stopwords.words("english")**.

To install the stopwords, got to the console and type the commands

nltk.download()


Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Tools View Help
C:\Users\Alemaire18\Documents\Presentations\ppt_oded\Text Mining
Editor - C:\Users\Alemaire18\.spyder2\temp.py
temp.py
-*- coding: utf-8 -*-
"""
Spyder Editor
This is a temporary script file.
"""
for i in
print
File explorer
Name Size Type Date Modified
homework.py 2 KB py File 4/5/2016 9:45:03 PM
PerplexityPlotFasterAggressive2.png 33 KB png File 4/5/2016 9:53:12 PM
yelp.stop 4 KB stop File 4/5/2016 8:19:45 PM
yelp_homework.csv 338 KB csv File 4/5/2016 7:46:03 PM
NLTK Downloader
File View Sort Help
Collections Corpora Models All Packages
Identifier Name Size Status
all All packages n/a not installed
all-corpora All the corpora n/a not installed
book Everything used in the NLTK Book n/a not installed
Download
Refresh
Server Index: https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
Download Directory: C:\Users\Alemaire18\AppData\Roaming\nltk_data
In [5]:
In [6]: nltk.download()
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
Console History log IPython console
Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 9 Column: 12 Memory: 75 %

You should see a new prompt (see picture above). click in the collections tab and at this point, you have 2 options either select all or all-corpora and then hit the download command.

I would suggest downloading the all options as you'll not have to this step ever again. But if memory is a issue for your computer then by all means go with all-corpora.

- **Disclaimer**:This might take a while!

To go over all the material in the class supplement, one can simply copy and paste all the commands and update the file name etc and one should be good.

Python Basics

In order to help you spend the least amount of time possible going over coding issues. I will give you some basic commands that might be useful for the homework. This commands will go from most basics to intermediate.

In [1]:

```
 a = 1 # variable decleration
```

In [2]:

```
a = 'b' ## setting a to a string
```

In [3]:

```
1+1, 2-1, 2**3, ## basic math operation in python add, substract exponent
```

Out[3]:

```
(2, 1, 8)
```

In [4]:

```
2/3 ## Division...this is bad  because it returns
```

Out[4]:

```
0
```

To remedy this issue and return decimal value one of the elements have to be a fraction

In [5]:

```
2/3.0
```

Out[5]:

```
0.6666666666666666
```

In [6]:

```
2.0/3 # good!
```

Out[6]:

```
0.6666666666666666
```

Reading the stop words

In python there are multiple ways to read the data, but for this homework I suggest using the pandas, read_csv commands.

Now i'll read the yelp.stop files, I will name the only column in the file as "word" and return all of this as a list. which is similar to the R vector data structure.

In [7]:

```
import pandas as pd
stop = pd.read_csv("yelp.stop", names =["words"])
```

In [8]:

```
stop
```

Out[8]:

	words
0	the
1	and
2	to
3	of
4	is
5	for
6	it
7	in
8	was
9	this
10	but
11	my
12	with
13	that
14	on
15	have
16	they
17	you
18	place
19	not
20	good
21	had
22	are
23	food
24	so
25	at
26	be

	words
27	great
28	there
29	we
...	...
695	whole
696	whom
697	whose
698	why
699	will
700	willing
701	wish
702	with
703	within
704	without
705	won't
706	wonder
707	would
708	would
709	wouldn't
710	x
711	y
712	yes
713	yet
714	you
715	you'd
716	you'll
717	you're
718	you've
719	your

	words
720	yours
721	yourself
722	yourselves
723	z
724	zero

725 rows × 1 columns

Although this is nice, we NEED the stopwords to be in a list format. Thus to convert the pandas dataframe to list, enter the following commands.

In [9]:

```
stop['words'].values.tolist()
```

Out[9]:

```
['the',
 'and',
 'to',
 'of',
 'is',
 'for',
 'it',
 'in',
 'was',
 'this',
 'but',
 'my',
 'with',
 'that',
 'on',
 'have',
 'they',
 'you',
 'place',
 'not',
 'good',
 'had',
 'are',
 'food',
 'so',
 'at',
 'be',
 'great',
 'there',
 'we',
 'were',
 'like',
 'if',
 'here',
 'all',
 'very',
 'out',
 'just',
 'as',
 'get',
 'one',
 'me',
 'service',
 'go',
 'or',
 'time',
 'back',
 'from',
 'when',
```

```
'their',
'up',
'really',
'about',
'some',
'an',
'will',
'been',
'would',
"it's",
'what',
'can',
'more',
'which',
'only',
'also',
'our',
"don't",
'by',
'your',
'too',
'other',
'no',
'love',
'nice',
'even',
'has',
'best',
'well',
'because',
'little',
"i'm",
'do',
'than',
'friendly',
'always',
'try',
"i've",
'got',
'them',
'much',
'after',
'staff',
'first',
'went',
'us',
'pretty',
'menu',
'never',
'restaurant',
'people',
```

```
'ordered',
'could',
'make',
'way',
'know',
'over',
'going',
'order',
"didn't",
'better',
'did',
'am',
'think',
'come',
'off',
'came',
'again',
'how',
'then',
'who',
'few',
'right',
'made',
'definitely',
'any',
'now',
'want',
'say',
'see',
'two',
'night',
'fresh',
'down',
'he',
'while',
'experience',
'sure',
'eat',
'before',
'new',
'still',
'since',
'around',
'lunch',
'take',
'next',
'ever',
'she',
'delicious',
'bar',
'chicken',
```

```
'her',
'wait',
'where',
'said',
'a',
"a's",
'able',
'about',
'above',
'according',
'accordingly',
'across',
'actually',
'after',
'afterwards',
'again',
'against',
"ain't",
'all',
'allow',
'allows',
'almost',
'alone',
'along',
'already',
'also',
'although',
'always',
'am',
'among',
'amongst',
'an',
'and',
'another',
'any',
'anybody',
'anyhow',
'anyone',
'anything',
'anyway',
'anyways',
'anywhere',
'apart',
'appear',
'appreciate',
'appropriate',
'are',
"aren't",
'around',
'as',
'aside',
```

```
'ask',
'asking',
'associated',
'at',
'available',
'away',
'awfully',
'b',
'be',
'became',
'because',
'become',
'becomes',
'becoming',
'been',
'before',
'beforehand',
'behind',
'being',
'believe',
'below',
'beside',
'besides',
'best',
'better',
'between',
'beyond',
'both',
'brief',
'but',
'by',
'c',
"c'mon",
"c's",
'came',
'can',
"can't",
'cannot',
'cant',
'cause',
'causes',
'certain',
'certainly',
'changes',
'clearly',
'co',
'com',
'come',
'comes',
'concerning',
'consequently',
```

```
'consider',
'considering',
'contain',
'containing',
'contains',
'corresponding',
'could',
"couldn't",
'course',
'currently',
'd',
'definitely',
'described',
'despite',
'did',
"didn't",
'different',
'do',
'does',
"doesn't",
'doing',
"don't",
'done',
'down',
'downwards',
'during',
'e',
'each',
'edu',
'eg',
'eight',
'either',
'else',
'elsewhere',
'enough',
'entirely',
'especially',
'et',
'etc',
'even',
'ever',
'every',
'everybody',
'everyone',
'everything',
'everywhere',
'ex',
'exactly',
'example',
'except',
'f',
```

```
'far',
'few',
'fifth',
'first',
'five',
'followed',
'following',
'follows',
'for',
'former',
'formerly',
'forth',
'four',
'from',
'further',
'furthermore',
'g',
'get',
'gets',
'getting',
'given',
'gives',
'go',
'goes',
'going',
'gone',
'got',
'gotten',
'greetings',
'h',
'had',
"hadn't",
'happens',
'hardly',
'has',
"hasn't",
'have',
"haven't",
'having',
'he',
"he's",
'hello',
'help',
'hence',
'her',
'here',
"here's",
'hereafter',
'hereby',
'herein',
'hereupon',
```

```
'hers',
'herself',
'hi',
'him',
'himself',
'his',
'hither',
'hopefully',
'how',
'howbeit',
'however',
'i',
"i'd",
"i'll",
"i'm",
"i've",
'ie',
'if',
'ignored',
'immediate',
'in',
'inasmuch',
'inc',
'indeed',
'indicate',
'indicated',
'indicates',
'inner',
'insofar',
'instead',
'into',
'inward',
'is',
"isn't",
'it',
"it'd",
"it'll",
"it's",
'its',
'itself',
'j',
'just',
'k',
'keep',
'keeps',
'kept',
'know',
'knows',
'known',
'l',
'last',
```

```
'lately',
'later',
'latter',
'latterly',
'least',
'less',
'lest',
'let',
"let's",
'like',
'liked',
'likely',
'little',
'look',
'looking',
'looks',
'ltd',
'm',
'mainly',
'many',
'may',
'maybe',
'me',
'mean',
'meanwhile',
'merely',
'might',
'more',
'moreover',
'most',
'mostly',
'much',
'must',
'my',
'myself',
'n',
'name',
'namely',
'nd',
'near',
'nearly',
'necessary',
'need',
'needs',
'neither',
'never',
'nevertheless',
'new',
'next',
'nine',
'no',
```

```
'nobody',
'non',
'none',
'noone',
'nor',
'normally',
'not',
'nothing',
'novel',
'now',
'nowhere',
'o',
'obviously',
'of',
'off',
'often',
'oh',
'ok',
'okay',
'old',
'on',
'once',
'one',
'ones',
'only',
'onto',
'or',
'other',
'others',
'otherwise',
'ought',
'our',
'ours',
'ourselves',
'out',
'outside',
'over',
'overall',
'own',
'p',
'particular',
'particularly',
'per',
'perhaps',
'placed',
'please',
'plus',
'possible',
'presumably',
'probably',
'provides',
```

```
'q',
'que',
'quite',
'qv',
'r',
'rather',
'rd',
're',
'really',
'reasonably',
'regarding',
'regardless',
'regards',
'relatively',
'respectively',
'right',
's',
'said',
'same',
'saw',
'say',
'saying',
'says',
'second',
'secondly',
'see',
'seeing',
'seem',
'seemed',
'seeming',
'seems',
'seen',
'self',
'selves',
'sensible',
'sent',
'serious',
'seriously',
'seven',
'several',
'shall',
'she',
'should',
"shouldn't",
'since',
'six',
'so',
'some',
'somebody',
'somehow',
'someone',
```

```
'something',
'sometime',
'sometimes',
'somewhat',
'somewhere',
'soon',
'sorry',
'specified',
'specify',
'specifying',
'still',
'sub',
'such',
'sup',
'sure',
't',
"t's",
'take',
'taken',
'tell',
'tends',
'th',
'than',
'thank',
'thanks',
'thanx',
'that',
"that's",
'thats',
'the',
'their',
'theirs',
'them',
'themselves',
'then',
'thence',
'there',
"there's",
'thereafter',
'thereby',
'therefore',
'therein',
'theres',
'thereupon',
'these',
'they',
"they'd",
"they'll",
"they're",
"they've",
'think',
```

```
'third',
'this',
'thorough',
'thoroughly',
'those',
'though',
'three',
'through',
'throughout',
'thru',
'thus',
'to',
'together',
'too',
'took',
'toward',
'towards',
'tried',
'tries',
'truly',
'try',
'trying',
'twice',
'two',
'u',
'un',
'under',
'unfortunately',
'unless',
'unlikely',
'until',
'unto',
'up',
'upon',
'us',
'use',
'used',
'useful',
'uses',
'using',
'usually',
'uucp',
'v',
'value',
'various',
'via',
'viz',
'vs',
'w',
'want',
'wants',
```

```
'was',
"wasn't",
'way',
'we',
"we'd",
"we'll",
"we're",
"we've",
'welcome',
'well',
'went',
'were',
"weren't",
'what',
"what's",
'whatever',
'when',
'whence',
'whenever',
'where',
"where's",
'whereafter',
'whereas',
'whereby',
'wherein',
'whereupon',
'wherever',
'whether',
'which',
'while',
'whither',
'who',
"who's",
'whoever',
'whole',
'whom',
'whose',
'why',
'will',
'willing',
'wish',
'with',
'within',
'without',
"won't",
'wonder',
'would',
'would',
"wouldn't",
'x',
'y',
```

```
 'yes',
 'yet',
 'you',
 "you'd",
 "you'll",
 "you're",
 "you've",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'z',
 'zero']
```

for-Loop in python

In python, in order to do a for loop of increment i, one only need the range function to iterate over the number. The syntax for range is range(beginning,end, increment). Example

In [10]:

```
for number in range(0,10,3):
    print number
```

```
0
3
6
9
```

another example this time squaring the output and over a longer range

In [11]:

```
for number in range(5,25,5):
    print number**2
```

```
25
100
225
400
```

Always Remember to indent everything that is in the for loop

In [12]:

```
data = pd.DataFrame()
data["X-axis"] =range(-10,10,1)
data["Y-axis"]= data['X-axis']**2
data
```

Out[12]:

	X-axis	Y-axis
0	-10	100
1	-9	81
2	-8	64
3	-7	49
4	-6	36
5	-5	25
6	-4	16
7	-3	9
8	-2	4
9	-1	1
10	0	0
11	1	1
12	2	4
13	3	9
14	4	16
15	5	25
16	6	36
17	7	49
18	8	64
19	9	81

Since now we have our data, We will need to import our plotting package in python this is called matplotlib. NOTES the line matplotlib inline was used only to insert the picture in this document.

In [13]:

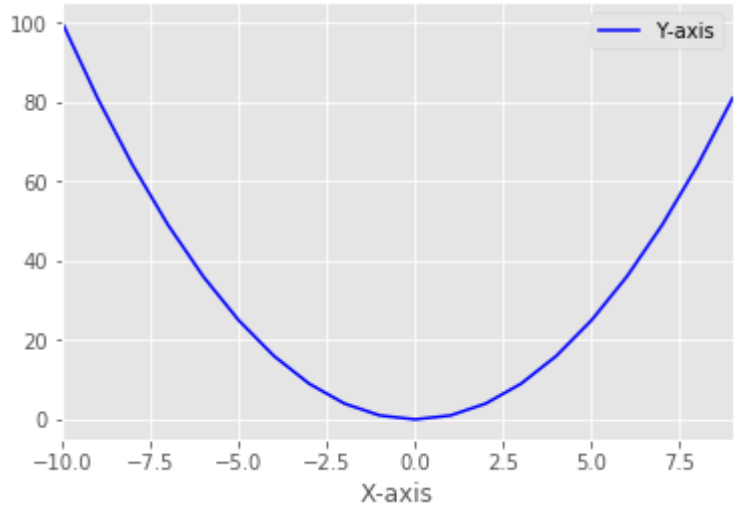
```
from matplotlib import pyplot as plt
import matplotlib
%matplotlib inline
matplotlib.style.use('ggplot') # use ggplot format
```

In [14]:

```
data.plot(x= "X-axis", y ="Y-axis", color = 'b')
```

Out[14]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f66e2caf450>
```



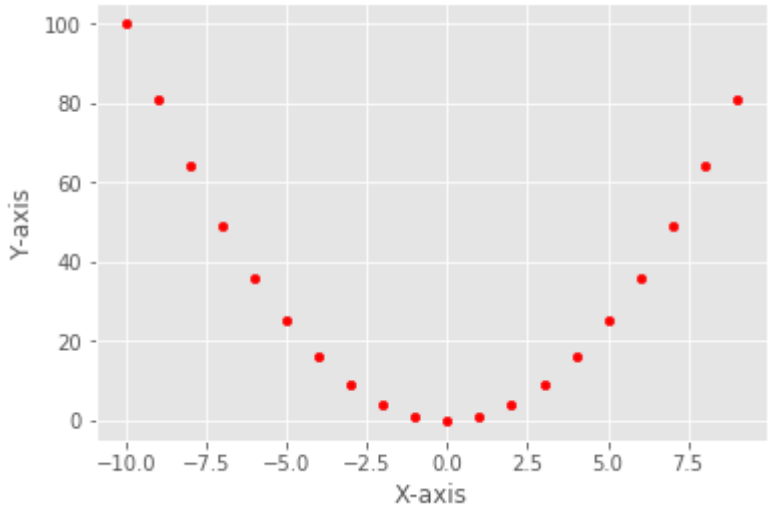
what if you wanted a scatter plot??

In [15]:

```
data.plot(kind ='scatter', x="X-axis", y= "Y-axis", color = 'r')
```

Out[15]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f66e2ba5e50>
```



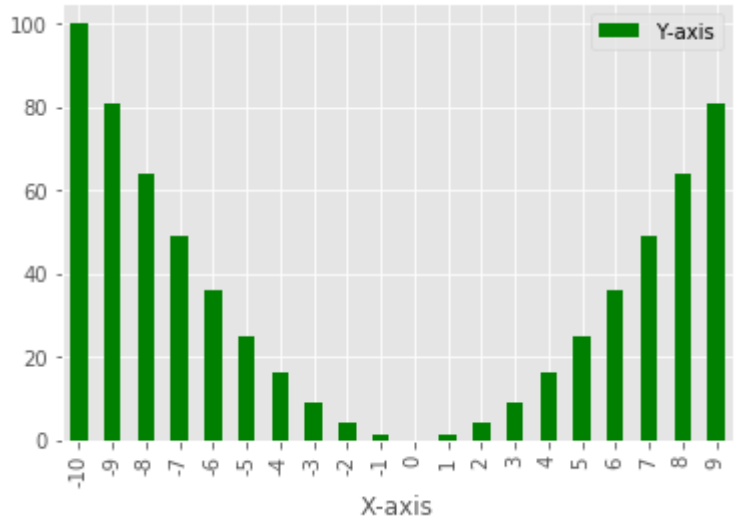
What about bar plot??

In [16]:

```
data.plot(kind ='bar', x = 'X-axis', y = 'Y-axis', color = 'g')
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f66df2875d0>
```



Nice huh?.. When doing this in spyder it is IMPORTANT to call plt.show() to display the graph after running the command above.

Conclusion

I hope this tutorial is helpful and will reduce the barrier to entry to python for you.

In [ ]:

```

```