PASS SQLSATURDAY

SQL SATURDAY PHILADELPHIA (835)

MAY THE 4TH BE WITH YOU!

Code Like a Snake Charmer

Introduction to Python!

Jamey Johnston, Sr. Data Scientist


QUESTION #2
What did you study to go in your field?
Please silence cell phones

Jamey Johnston

Sr. Data Scientist

/jameyj

@STATCowboy

Project Coach Texas A&M Analytics

Education

Texas A&M - MS in Analytics

LSU - BS in Spatial Analysis

Photographer

http://jamey.photos

Blog

http://STATCowboy.com

Code

https://github.com/STATCowboy/SnakeCharmer-Intro

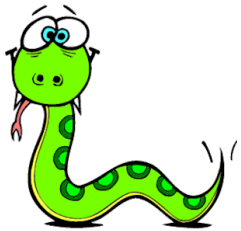
Agenda

- Introduction to Python
- Anaconda / IDEs
- Comments, Numbers and Strings
- Lists, Tuples and Dictionaries
- Pandas
- Control Flows
- Functions
- Packages
- Python and Microsoft
- Demos

Source: https://www.python.org/community/logos/

Introduction to Python

Why Python?



- Expansive Open Source Library of Data Science Tools (Giant Ecosystem)
- Easy language for new programmers
- Microsoft Support in tools like Azure Machine Learning, SQL Server 2017, Microsoft Machine Learning Server
- You can code on a Raspberry Pi (Who doesn't like Pi!)
- One of the most popular program languages (IEEE/GitHub ranked Python #3 in 2016)
- Interpreted language, saves you time, no compilation and linking is necessary

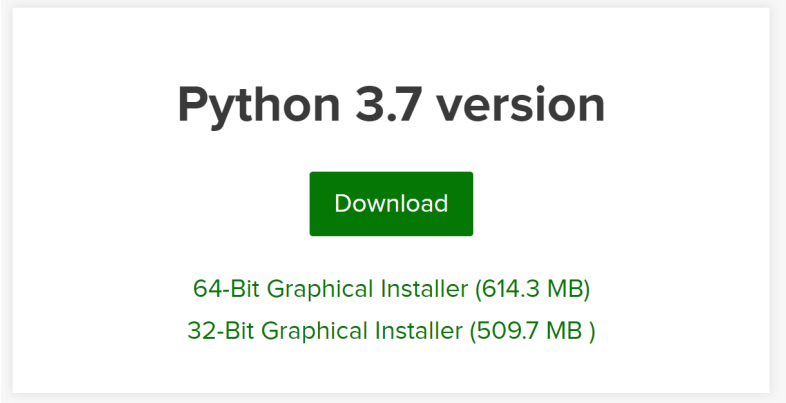
Anaconda

Source: http://www.anaconda.com

Anaconda

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

Download the 64-bit Python 3.7 version (still can setup Python 2.7 environments)

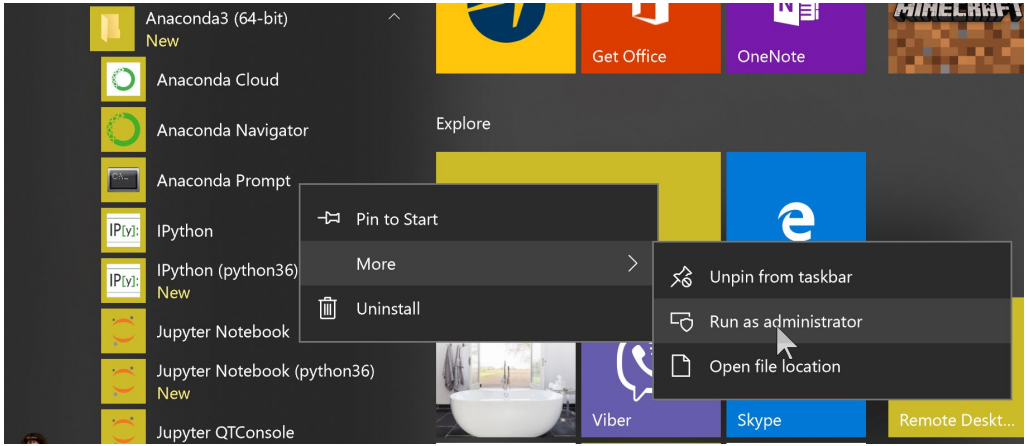


Anaconda

Conda

Open Source Package Management System and Environment Management System

Launch the “Anaconda Prompt” as Administrator to Manage Anaconda Environment



Anaconda

Conda Commands

- Upgrade All Packages
  - `conda update --all`
- Setup New Environment (e.g. Python 3.7)
  1. `conda create --name python37 python=3.7`
  2. `activate python37`
  3. Install Packages (few examples below)
     1. `conda install seaborn`
     2. `conda install spyder`
     3. `conda install jupyter`
- Setup a Python 2.7 Environment: Use above steps and change 36 to 27 and 3.7 to 2.7

Anaconda

Conda Commands

- List Environments
  - `conda env list`
  - * indicates active environment
- List Packages in Environment
  - `conda list`
- Remove an Environment
  - `conda env remove --name deleteme`
- Update Package
  - `conda update PACKAGENAME`

https://conda.io/docs/_downloads/conda-cheatsheet.pdf

Conda

Demo

Python IDE

PyCharm

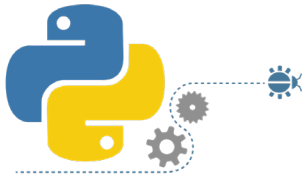
https://www.jetbrains.com/pycharm/

Spyder

Included in Anaconda Distribution

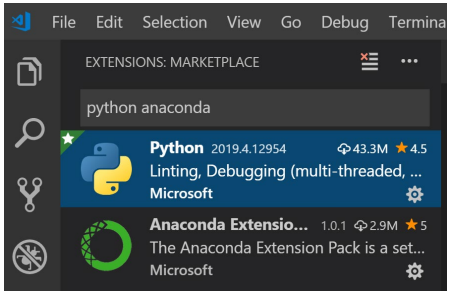
Visual Studio Code

https://code.visualstudio.com/docs/languages/python
https://code.visualstudio.com/docs/python/python-tutorial
https://marketplace.visualstudio.com/items?itemName=ms-python.python



Visual Studio Code

VS Code Python Shortcuts



https://code.visualstudio.com/docs/python/python-tutorial

- Command Palette (CP) – `Ctl+Shift+P`
- Select Python Interpreter (in CP) – `Python: Select Interpreter`
- Run Selection/Line in Python Terminal – `Shift+Enter`
- Install pylint for Highlighting Syntax – `conda install pylint (run in all env)`
- Install Python and Anaconda Extensions

PyCharm

PyCharm Shortcuts

https://www.jetbrains.com/help/pycharm/2016.1/keyboard-shortcuts-you-cannot-miss.html
https://www.jetbrains.com/help/pycharm/keyboard-shortcuts-by-category.html

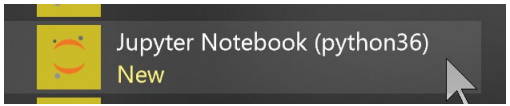
- Run – `Alt+Shift+F10`
- Run Selection / Current Line – `Alt+Shift+E`
- Comment / Uncomment Code – `Ctrl+Slash / Ctl+Shift+Slash`
- Invoke Code Completion – `Ctl+Space`
- Indent / Un-indent (selection of code) – `Tab / Ctl+Tab`

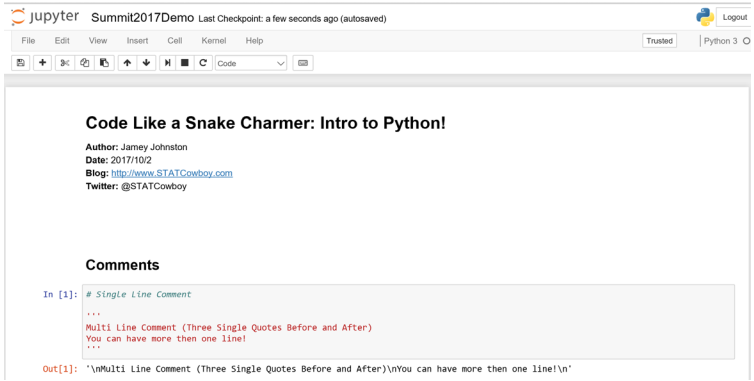
Jupyter Notebooks

Computer Code and Rich Text

http://jupyter-notebook.readthedocs.io/en/latest/

- Activate desired environment first
- Then to Start a Notebook – `jupyter notebook`





IDE / Tools

Demo

Comments

Single Line Comment

- Pound Sign/Hash is used for single line comments

```
# Single Line Comment
```

Multi-Line Comment

‘ ‘ ‘- Three single-quotes before and after the comments

```
'''
Multi Line Comment (Three Single Quotes Before and After)
You can have more then one line!
'''
```

Numbers

Operators “+, -, * and / “ as you would expect!

```
taxRate = 8.25 / 100
price = 100
tax = price * taxRate
finalPrice = price + tax
print('Tax: ${:,.2f}'.format(tax))
print('Final Price: ${:,.2f}'.format(finalPrice))

Tax: $8.25
Final Price: $108.25
```

Strings

single quotes ('...') or double quotes ("...")

```
simpleString = 'This is a simple string!'
print(simpleString)
simpleStringDouble = “This is a simple string!”
print(simpleStringDouble)
This is a simple string!
This is a simple string!
```

Strings

Escape with "\"

```
print('Isn\'t Pass Summit Awesome')
Isn't Pass Summit Awesome
```

Strings

Span String Literals Multiple Lines

```
print("""\
Usage: magicSummitPass [OPTIONS]
     -h                         Display this usage message
     -S year                    Magically get me into Summit that year for free!
""")
```

Strings

Repeat Strings with "*" and Concatenate with "+"

```
espn = 3*'duh '+' (we still wish MJ was playing!)  '+3*'duh '
print(espn)

duh duh duh  (we still wish MJ was playing!)  duh duh duh
```

Strings

Slicing/Indices on Strings

Positive indexes start at 0 and Negative start with -1

```
passSummit = 'PASS Summit 2017'
```

```
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| P | A | S | S |   | S | u | m | m | i | t |   | 2 | 0 | 1 | 7 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
-16 -15 -14 -13 -12 -11 -10  -9  -8  -7  -6  -5  -4  -3  -2  -1
```

Strings

Important Notes

Strings are Immutable (i.e. you can't change them)

`len()` – will return the length of the string

Basics

Demo

Lists

Compound Data Type

Used to group values together.

Comma-separated values/items enclosed by square brackets.

List can contain different types of data but usually they contain the same types.

```
myList = [1,2,3,4]
```

Lists

Slice and Index List

```
myList[0]
myList[-3:] # slicing returns a new list
```

Concatenate Lists

```
myNewList = myList + [5,6,7,9]
```

Lists

List are mutable (you can change them!)

```
myNewList[7] = 8
```

Append to a List

```
myNewList.append(9)
```

Lists

Replace a slice (even with a different size)

```
myNewList[2:4] = [1,1]
```

Length of list

```
len(myNewList)
```

Tuples

Number of Values Separated by Commas

```
t = 'PASS', 'Summit', '2017'
```

Tuples may be Nested

```
nt = t, ('is', 'awesome', '!')
```

Tuples

Tuples are Immutable

```
t[2] = '2018’  # Will throw an error!
```

Dictionaries

Unordered key/value pairs

```
yearBirth = {'jamey': 1974, 'melanie': 1975, 'jeanna': 1989, ‘robyn': 1979}
```

Delete item in Dictionary

```
del yearBirth['robyn']
```

Dictionaries

List Keys (unordered)

```
list(yearBirth.keys())
```

List Keys (sorted/ordered)

```
sorted(yearBirth.keys())
```

Pandas

Series and DataFrame

Labeled Array Data Structures

Input/output Tools (CSV, Excel, ODBC)

http://pandas.pydata.org/pandas-docs/stable/10min.html

Pandas

DataFrame

Import Pandas and Read CSV

```
import pandas as pd
baseball = pd.read_csv('baseball.csv', sep=',', encoding='UTF-8')
```

Print header of pandas DataFrame

```
baseball.head()
```

Print tail of pandas DataFrame

```
baseball.tail(3)
```

Pandas

DataFrame

Describe DataFrame

```
baseball.describe()
```

Sort by Column

```
baseball.sort_values(by='Attendance')
```

Select one Column

```
baseball[['Team']]
```

Pandas

DataFrame

Group By

```
baseballMean = baseball.groupby('Team').mean()
print(baseballMean.sort_values(by='Attendance')[['Attendance']])
```

```
           Attendance
Team
Royals    17597.812500
Phillies  20484.825000
Reds      23108.587500
Cubs      34575.037037
```

Data Structure

Demo

Control Flows

Indention

Indention is used to indicate the scope of a block of code (like { … } in other languages)
Blank lines do not affect indention, Same as Comments on a line by themselves

Word of CAUTION: Turn OFF Tabs!!!

If you copy and paste from the internet you indentions will more than likely be Tabs!

Python cares a great deal about indention! You will get “indention errors” if not right.

Control Flows

Conditionals / Comparisons

PYTHON CODE	RESULT
==	Equal To
!=	Not Equal To
<	Less Than
<=	Less Than or Equal To
>	Greater Than
>=	Greater Than or Equal To

Control Flows

if … elif … else

```
n = 5
m = 10
if n < 10 and m <  10:
    print('n and m are single digit numbers!')
elif n >= 10 and m < 10:
    print('n is a big number and m is a single digit number!')
elif n < 10 and m >= 10:
    print('n is a single digit number and m is a big number!')
else:
    print('n and m are big number!')
```

Control Flows

IN Operator on List

```
if 2 in [1, 2, 3, 4]:
    print('Found it!')
else:
    print('Keep looking!')
```

Control Flows

for Loops

```
for i in [1, 2, 3, 4]:
    print(i)

wordList = ['Jamey', 'Melanie', 'Stefanie', 'Robyn']
for word in wordList:
    print('Family member name:', word)
```

Control Flows

Range Function

```
r = range(5)
print(r)
for num in r:
    print(r[num])
```

Control Flows

Loop over two or more lists

```
questions = ['name', 'birth year', 'occupation']
answers = ['Jamey Johnston', '1974', 'Data Scientist']
for q, a in zip(questions, answers):
    print('What is your {0}?  It is {1}.'.format(q, a))
```

Control Flows

Retrieve Key/Value of List in Loop, Sorted by Key

```
yearBirth = {'jamey': 1974, 'melanie': 1975, 'jeanna': 1989}
for k, v in sorted(yearBirth.items()):
    print(k, 'was born in the year ', v)
```

Control Flows

break, continue and else

```
for n in range(2, 10):
    for x in range(2, n):
        if n % x == 0:
            print(n, 'equals', x, '*', n//x)
            break
    else:
        # loop fell through without finding a factor
        print(n, 'is a prime number’)
```

https://docs.python.org/3/tutorial/controlflow.html#break-and-continue-statements-and-else-clauses-on-loops

Control Flows

break and continue … try and except

```
while True:
    txt = input('Enter number (integers only!):')
    try:
        integer = int(txt)
    except:
        print('Please enter integer only!')
        continue
    print('You entered the integer,', integer)
    break
print('Done!')
```

Control Flows

while Loops

```
num = 0
while num < 10:
    print(num)
    num = num+1
```

Functions

Simple Function

```
# NOTE: non-default parameters must be first!
def greetSummit(year, name=None):
    if name is not None:
        print('Welcome to PASS Summit ', year, ', ', name, '!', sep='')
    else:
        print('Welcome to PASS Summit ', year, '!', sep='')

greetSummit(2017)
greetSummit(2017, 'Jamey')
```

Control Flows

Demo

Packages

pip

PyPA recommended tool for installing Python packages

Some packages are not in the conda repository (e.g. latest tensorflow packages)

```
pip install --ignore-installed --upgrade tensorflow-gpu
```

conda

Anaconda Distribution package manager (Use conda if using Anaconda)

```
conda install pyodbc
```

Packages

Import Module from Package

Import sys and show Python version/distribution

```
import sys
sys.version
```

PYODBC/Pandas Example

```
import pyodbc
import pandas.io.sql as psql
```

Packages

Popular Packages

PACKAGE	DETAILS
pandas	High performance, easy use data structures and analysis (DataFrames)
pyodbc	Open Source Python Module for ODBC data sources
matplotlib	2D Plotting library
scikit-learn	Simple tool for data mining and data analysis / statistics
numpy	N-dimensional arrays, linear algebra, random numbers
SciPy	Math, Stats, Science and Engineering package

Packages

Demo

Python and Microsoft SQL Server 2017

sp_execute_external_script

Executes Python via T-SQL in MSSQL 2017

Install Machine Learning Services (In-Database)

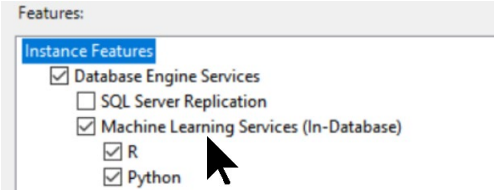
Anaconda Distribution installed with MLS

New revoscalepy library – scale and performance

Executes outside the SQL Server process

Data returned as a pandas data frame

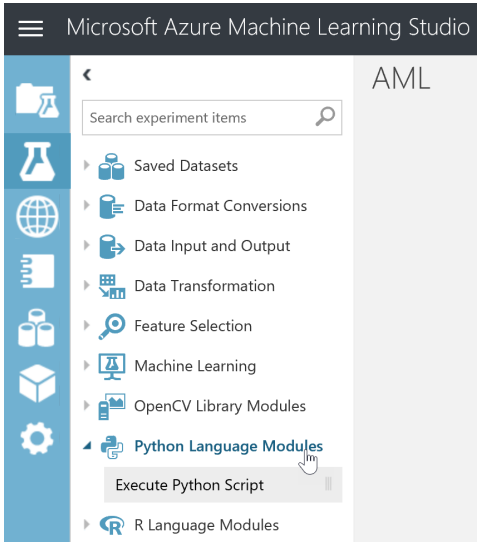
Also, supports R

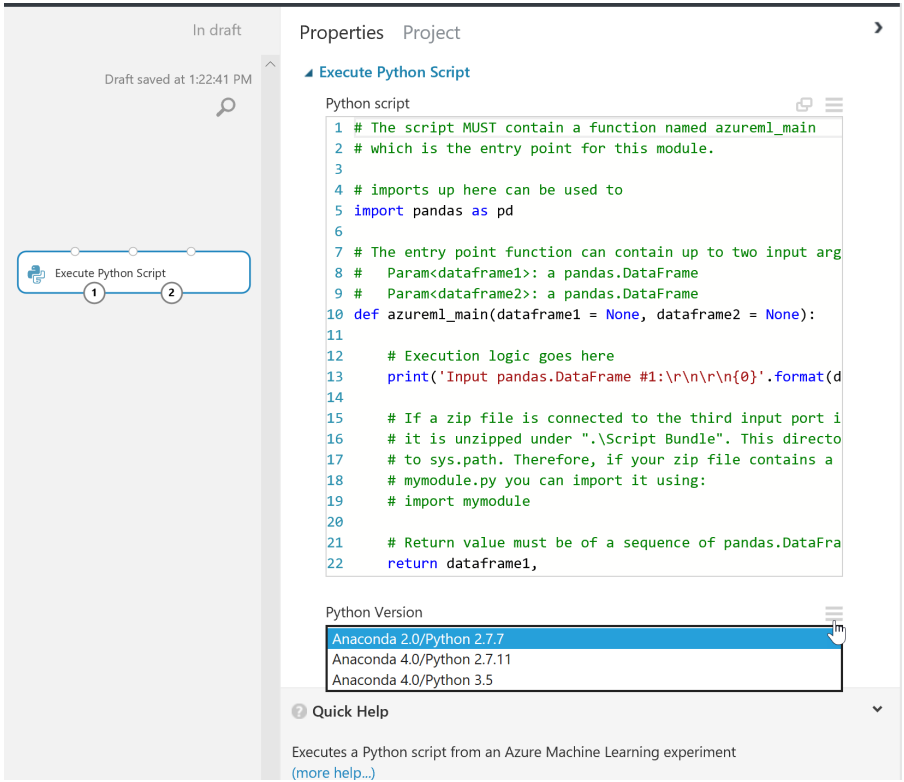



```
sp_execute_external_script
    @language = N'language' ,
    @script = N'script',

    @input_data_1 = ] 'input_data_1'
    [ , @input_data_1_name = ] N'input_data_1_name' ]
    [ , @output_data_1_name = 'output_data_1_name' ]
    [ , @parallel = 0 | 1 ]
    [ , @params = ] N'@parameter_name data_type [ OUT | OUTPUT ] [ ,...n ]'
    [ , @parameter1 = ] 'value1' [ OUT | OUTPUT ] [ ,...n ]
    [ WITH <execute_option> ]
[;]

<execute_option>::=
{
      { RESULT SETS UNDEFINED }
    | { RESULT SETS NONE }
    | { RESULT SETS ( <result_sets_definition> ) }
}

<result_sets_definition> ::=
{
    (
        { column_name
          data_type
        [ COLLATE collation_name ]
        [ NULL | NOT NULL ] }
        [,...n ]
    )
    | AS OBJECT
       [ db_name . [ schema_name ] . | schema_name . ]
       {table_name | view_name | table_valued_function_name }
    | AS TYPE [ schema_name.]table_type_name
}
```

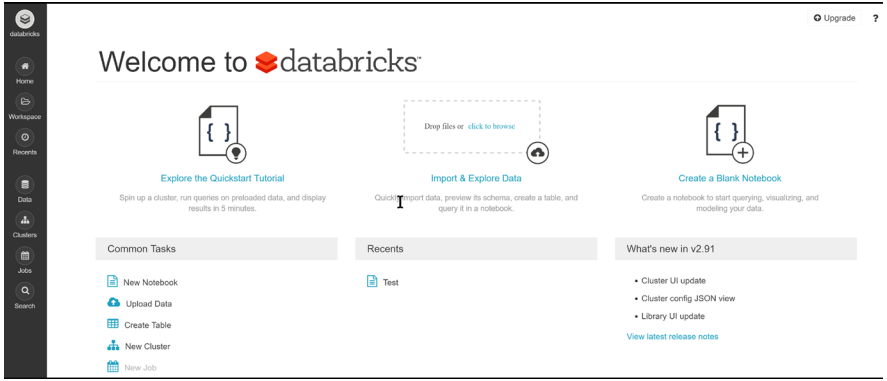
Python and Azure Machine Learning

Execute Python Script



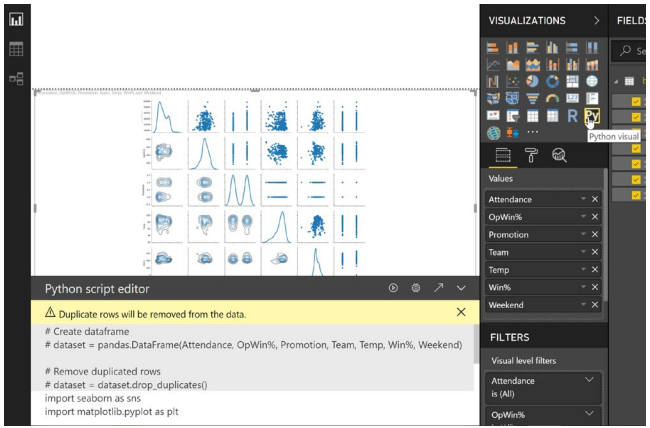


Python and Azure Databricks

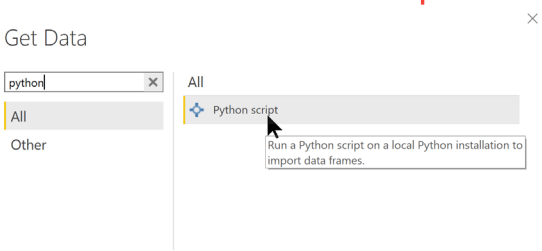


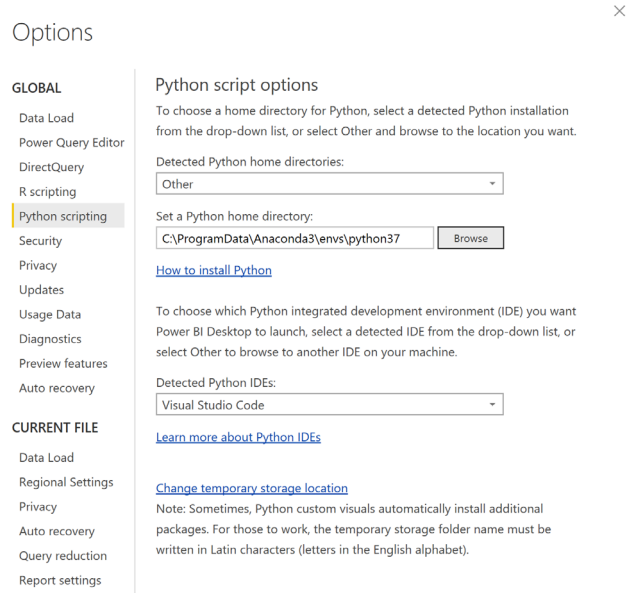
Workbooks

Python and Power BI



Visuals/Scripts





Set Env

PASS SQLSATURDAY PHILADELPHIA | MAY 04 2019

MS & Python

Demo

Data Science

Demo

References

Python Docs

https://docs.python.org/3/reference/introduction.html

Coursera

https://www.coursera.org/specializations/python

MS Academy

https://academy.microsoft.com/en-us/professional-program/tracks/data-science/

References

The Hitchhiker's Guide to Python!

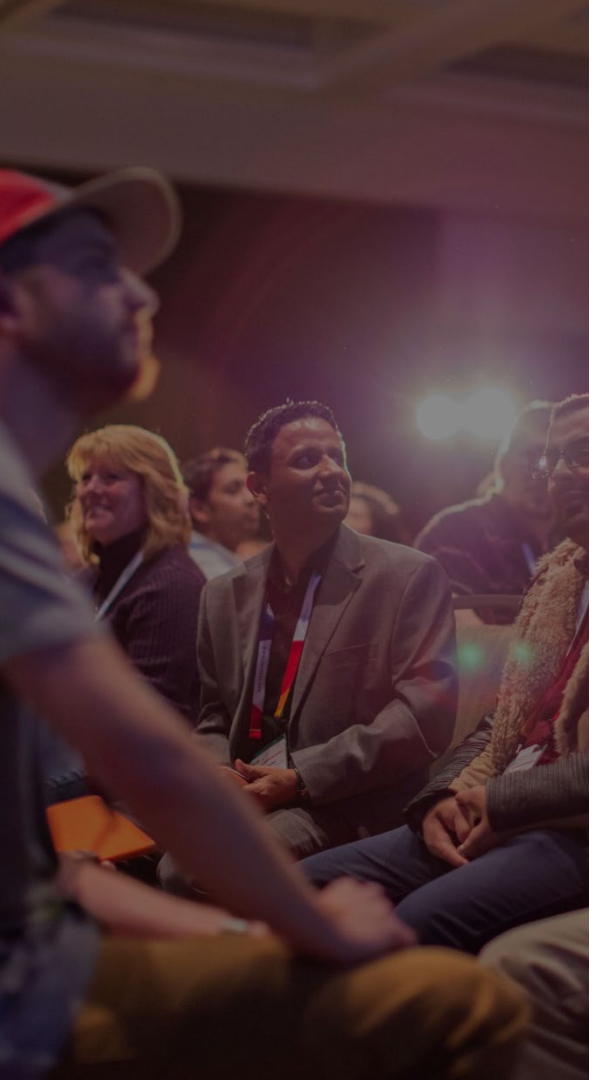
http://docs.python-guide.org/en/latest/

Code Academy

https://www.codecademy.com/en/tracks/python

Google

https://developers.google.com/edu/python/?hl=en



Thank You

Jamey Johnston

@STATCowboy jameyj@tamu.edu

Dr. Je'Anna Abbott

@STATWonderWoman jabbott@uh.edu

PASS SQLSATURDAY PHILADELPHIA | MAY 04 2019