How to Install and use Python in KNIME

1) Install Python3 on your Local Machine.

2) Install Anaconda Environment

3) Once you've got Anaconda installed, open a Shell (Linux), Terminal (Mac), or Command Prompt (Windows) and create a new Python environment for use inside of KNIME.

Using the following command:

```
conda create -y -n py35_knime python=3.5 pandas jedi
```

4) If there are additional packages you would like to install, go ahead and add them to the end of the command line. If you would like to install Python 2.7 instead of 3.5, just change the version number in the command.

5) In order to use this new Python Environment from inside KNIME, you could create a Script (Shell Script in Linux and Mac or a .bat file on Windows) to launch it.

6) If you are using Linux or Mac, here is an example of the Shell Script for creating the Python Environment defined below:

```
#! /bin/bash
  # start by making sure that the anaconda directory is on the
PATH
  # so that the source activate command works.
  # This isn't necessary if you already know that
  # the anaconda bin dir is on the PATH
  export PATH="PATH_WHERE_YOU_INSTALLED_ANACONDA/bin:$PATH"

  source activate py35_knime
  python "$@" 1>&1 2>&2
```

You will have to edit `PATH_WHERE_YOU_INSTALLED_ANACONDA` with the directory there you have installed Anaconda in the Shell Script above.

Here we have named the Script as py35.sh, and made this executable using:

("`chmod gou+x py35.sh`") and put it on my home directory.

7) For Windows:

We need to create a .bat file:

```
@REM Adapt the directory in the PATH to your system
  @SET PATH=C:\tools\Anaconda3\Scripts;%PATH%
  @CALL activate py35_knime || ECHO Activating py35_knime failed
  @python %*
```

You will need to replace `C:\tools\Anaconda3` with the directory of where you have installed Anaconda.

Here we have named the file py35.bat and put it in my home directory.

Congratulations!! Now you have everything required to use Python in KNIME.

Configuring KNIME

The Python nodes can be added in KNIME under "File → Install KNIME Extension" by selecting "KNIME Python Integration".

WINDOWS:

1) Install Anaconda**(for Python 2 not Python 3)**

2) Start installing Anaconda (make sure that "Add Anaconda to my PATH Environment Variable" is checked.

3) Open Command Prompt.

4) Execute 'pip install protobuf'

5) In KNIME go to "File → Preferences → KNIME → Python" and select the python.exe in Anaconda Directory.

6) Restart KNIME

MAC OS:

There are 2 options:

1) Using the Homebrew package manager

- Open the Terminal
- Execute "brew install python "
- Execute "pip install protobuf jedi"

- In KNIME under " File →Preferences → KNIME →Python" enter /usr/local/bin/python (Homebrew default location)
- Restart KNIME

2) Plain Installation:

- Open the Terminal
- Execute "sudo easy_install pip"
- Execute "sudo pip install pandas protobuf jedi"
- Restart KNIME

Note: The first option is preferred as it uses a Python installation separate from which is available in your system. This could avoid problems on your local system (Python)

LINUX:

Ubuntu:

- Open the Terminal
- Execute 'sudo apt-get install python-pandas python-protobuf python-jedi'
- Restart Knime.
- Configure Python using the Preferences page KNIME → Python:
- On this page you need to provide the path to the script/bat file you created to start Python. If you like, you can have configurations for both Python 2 and Python 3 (as I do above). Just select the one that you would like to have as the default.
- If you've completed the steps above and after you click "Apply" KNIME shows the correct version number for Python in the dialog, you're ready to go.

Note:

Same way you can Install Python Plugin called "KNIME Python Scripting extension" which is a Python integration provided by the community.

HOW TO INSTALL PYTHON LIBRARIES IN KNIME:

The following script should be copied in Python Scripting Node:

```
from pandas import DataFrame
import subprocess
import sys

def install(package):
    subprocess.call([sys.executable, "-m", "pip", "install", "--user",
package])

install("Name of your package")
install("Name of your package")
flow_variables['installed'] = "true"
```

You will just need to edit “Name of your package” to the library/package you want to use example. Seaborn / matplotlib/ pandas/ numpy etc.