## PYTHON, PACKAGES, TENSORFLOW, and KERAS

Software libraries get revised frequently. Therefore, the commands, and software versions mentioned below may undergo changes. Refer to the home page of the respective software projects for current usage. Alternatively, one may refer to https://github.com/srikrishnan-punalur/MVA-ML.

### Package Managers – conda, and pip

Conda and pip are both used for installing Python packages. We can install anaconda and use the conda command from the anaconda prompt to install other packages of the Python ecosystem. Alternatively, the pip command from the command prompt or the !pip from a Python IDE can be used to install packages. The earlier method (conda) is recommended for beginners, though the latter method may be used by professionals for installing specific packages and setting up work environments. The objective of this section is to develop a primary understanding of software installation for building a working Python development environment for machine learning.

As mentioned above, we can use the !pip for package installation in Google Colab, and other IDEs. The command the !pip is also used to check and resolve path dependencies of the installed packages. This ensures that the packages are not broken and are accessible for use.

The Python Package Index (PyPI) is a repository for Python Software packages. PyPI. Python software package developers use PyPI to distribute their software. PyPI has a massive 348,665 projects. **Pip** is the Python Packaging Authority’s recommended package manager for installing python packages from the Python Package Index. Pip installs Python software packaged as wheels or source distributions. The latter may require that the system have compatible compilers, and possibly libraries, installed before invoking pip to succeed.

Conda is a cross-platform package and environment manager that installs and manages software packages from the Anaconda repository and the Anaconda Cloud. The repository has about 1500 software packages. They have Python packages and other software such as C or C++ libraries, and R packages. Conda packages are binaries. Therefore, there is no need for compilation on the host system.

Python must be installed, before using pip manager. We may download Python from python.org and install it. However, Conda can install Python, and Python packages directly using conda commands. Pip depends on other tools like virtualenv or venv to create isolated environments. However, Conda can create isolated environments that consist of different versions of Python/packages.

While installing packages, pip installs dependencies in a recursive, serial loop. This may lead to environments that are broken if the compatibility of packages installed earlier conflicts with the packages installed later. However, conda checks the package metadata and resolves dependencies, thereby creating a robust work environment. The steps involved in software installation are shown below.

### Install Anaconda, and Python

Install Anaconda

It is recommended that we download the latest version of Anaconda and install it before installing Python or Python packages

* If we accept the default option to install Anaconda, it is installed in your user home directory:

C:\Users\<your-username>\Anaconda3\

* It is recommended that we do not install anaconda in paths that contain spaces such as C:\Program Files. Instead, create your directory in an appropriate folder with an appropriate name. For example,

C:\anaconda

* For doing the above (a) you must select the ‘multiuser’ option and (b) you must have admin authority.
* While installing Anaconda, it is recommended that you may not add the anaconda path (C:\anaconda) to the Windows PATH.

Install Python

After installing anaconda, you may install Python, using Conda prompt. For this, on Windows, open the Start menu and open Anaconda Command Prompt. Execute the following command

conda create -n py39 python=3.9

conda activate py39

(assuming that python 3.9 is the latest stable version of python)

If an older version of Python is already installed, you may upgrade it from the ‘anaconda prompt’ using the above command.

### Install Other Packages

We can use the anaconda package manager to install any package. See Table 1-1.

Table ‑ Using the ‘conda’ command for Package Installation

|  |  |
| --- | --- |
| Run ‘conda’ Package Manager Command | Package Functionality  (See Table1-2 for references |
| conda install -c conda-forge python-graphviz  conda install -c conda-forge pingouin  conda install -c desilinguist factor\_analyzer  conda install -c sebp scikit-survival  conda install -c conda-forge eli5  {Use *pip install apyori* command; See Table 1-2} | To Draw Decision Tree  Partial Correlation coef.  Factor Analysis  Survival Analysis  Survival -Feature Ranking  Market Basket Analysis |

* if a package is already installed, it may be upgraded using the same command used above. For example,

conda install pingouin

### Install TensorFlow and Keras

On Windows, open the Start menu, then open an Anaconda Command Prompt.

* Choose a name for your TensorFlow environment, such as ‘TensorFlow’. Beginners are recommended to install the current release of CPU-only TensorFlow.

conda create -n tf tensorflow

conda activate tf

* After installing anaconda, python, and tensor flow, install Keras

conda install -c conda-forge keras

(or Run conda install -c conda-forge/label/broken keras)

### Google Colab, Spyder, and other IDEs

We saw how to use conda or pip for package installation. There is yet another method. We can use the !pip command from within an IDE (e.g., google colab or sypder) for package installation. See Table 1-2. So, how do you install python packages, in Google Colab? You must use the above method (!pip command) from within the Colab.

Table ‑ !pip - Google Colab and other IDEs, Package Installation / Path Resolution

|  |  |
| --- | --- |
| Installation of Python Packages in Google Colab or other IDEs (the same can be used to resolve code break issues, due to incorrect installation) | |
| !pip3 install --upgrade pip | <https://pypi.org/project/pip/> |
| !pip install graphviz | <https://graphviz.org/> |
| !pip install pingouin | https://pingouin-stats.org/ |
| !pip install factor-analyzer | <https://factor-analyzer.readthedocs.io/en/latest/factor_analyzer.html> |
| !pip install apyori | <https://pypi.org/project/apyori/> |
| !pip scikit-survival | <https://scikit-survival.readthedocs.io/en/stable/install.html> |
| !pip eli5 | (Same as above) |
| !pip3 install --upgrade tf | <https://www.tensorflow.org/install> |

### Check / Resolve conflicts in package installation

When we try to access an installed package from within an IDE, we may get the error - ‘ModuleNotFoundError’. One of the reasons may that the system is not able to resolve the access path. !pip command can be used to check and resolve path dependencies of the installed packages (See Table 1-2). This ensures that the packages are not broken and are accessible for usage. You may do the following inspections to ensure smooth library access.

* Check TensorFlow installation

!python -m pip show tf

>> WARNING: Package(s) not found: tf

* TensorFlow – Resolve the error reported above

!pip3 install --upgrade tf

!python -m pip show tf

>> Name: tf

>> Version: 1.0.0

>> Location: c:\users\sri-s\anaconda3\lib\site-packages

* Check Keras installation

!python -c "import keras; print(keras.\_\_version\_\_)"

2.9.0

* Check Access to any Package, from within an IDE

Import pandas as pd

from sksurv.datasets import load\_gbsg2

### Installation from the Command Prompt - for experienced programmers

It is recommended that we use the conda package manager for installing the Python work environment. However, we may need to use the pip package manager from the command prompt for some packages. See Table 1-3. The only difference with Table 1-2, is the absence of the exclamation mark (pip, instead of !pip)

Table ‑ System Command Prompt pip Command and package references

|  |  |
| --- | --- |
| Installing Python Packages from system command prompt | Download the package to an appropriate directory. From there issue pip command |
| pip3 install --upgrade pip | Package Manager pip |
| pip install graphviz | <https://graphviz.org/> |
| pip install pingouin | https://pingouin-stats.org/ |
| pip install factor-analyzer | [https://factor-analyzer.readthedocs.io/en/latest/factor\_analyzer.html](https://factor-analyzer.readthedocs.io/en/latest/factor_analyzer.html%20) |
| pip install apyori | https://pypi.org/project/apyori/ |
| pip scikit-survival  pip eli5 | https://scikit-survival.readthedocs.io/en/stable/install.html |

We may download a package from the web, install it, and add the path to the ‘PATH’ variable. For example, download the Graphviz installer to an appropriate folder, e.g., C:\Analytics\graphviz. After downloading the installer, we can use ‘pip’ from the Microsoft Windows command prompt 'cmd'.

* Download-> graphviz-2.50.0 (64-bit) EXE installer [sha256]
* CD C:\Analytics\graphviz
* pip install graphviz

* Check the python version in the IDE

import os

os.environ["PATH"] #check the current access paths

* If graphvz is not present in the path, add it to the path, as follows:-

os.environ["PATH"] += os.pathsep + 'C:\\Analytics\\graphviz\\bin'

* In the same way, as described above, other packages can be installed
* if a package is already installed, it may be upgraded. For example,

pip install --upgrade pingouin

pip install pingouin