

Deep Dive Into Important AI / ML Python Packages

Explore essential Python packages for AI and machine learning development. Learn about key libraries that power data analysis, model building, and deployment.

Contents

01	Introduction	3
02	Frameworks and Libraries	3
03	Hugging Face	12
04	Datasets	14
05	Plotting and Computer Vision Tools	15
06	Usability Tools	18
07	Development Environments	20
08	Conclusion	21

Introduction

Artificial Intelligence (AI) and Machine Learning (ML) are important cloud computing fields that involve the implementation of multiple software packages. These packages offer developers a suite of ready-to-deploy programs that handle specific tasks. This way, you can focus on building solutions instead of writing underlying code and implementing complex algorithms from scratch.

This article explains important packages commonly used to build production-ready Machine Learning and Artificial Intelligence applications.

Frameworks and Libraries

TensorFlow

TensorFlow is an open-source library of tools designed to accelerate Machine Learning tasks. It is used in both machine learning and deep learning models with support for CPUs, GPUs, and TPUs. By application, TensorFlow is commonly used for tasks such as:

- Traditional ML problems such as classification and regression
- Generative AI applications like Stable Diffusion
- Computer Vision (CV) models such as object recognition
- Deployment of production models using TensorFlow Serving

To install TensorFlow in a Python environment, use a package manager such as

```
pip
```

CONSOLE

```
$ pip install tensorflow
```

When installed, import TensorFlow to your applications to start using the package

PYTHON

```
>>> import tensorflow as tf
```

To use TensorFlow on your server, visit the following resources:

- [How to Install TensorFlow on Ubuntu 22.04](#)
- [Implement Stable Diffusion with TensorFlow and Keras](#)
- [Build a Discord Chatbot with TensorFlow](#)
- [Build a Question Answering Engine Using TensorFlow](#)
- [Implement an Object Detection Model using Keras and TensorFlow](#)
- [Deploy a Machine Learning Model in Production with TensorFlow Serving](#)

PyTorch

PyTorch is an open-source framework based on the Torch machine learning library that's commonly used in Computer Vision (CV) and Natural Language Processing (NLP) tasks. PyTorch packages work on both GPU and CPU systems with the ability to:

- Work with large datasets
- Build Multimodal models such as handwriting recognition
- Build deep learning models
- Deploy deep learning models in production using TorchServe

To install PyTorch in a Python environment, use a package manager such as `pip` and specify the system type, whether GPU or CPU. For example, install PyTorch with GPU support

CONSOLE

```
$ pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118
```

Install PyTorch with CPU support

CONSOLE

```
$ pip install torch torchvision torchaudio
```

To use PyTorch on a Vultr Cloud Server, visit the following resources:

- [How to Install PyTorch on Ubuntu 22.04](#)
- [Deploy a PyTorch Workspace](#)
- [Use WebDataset with PyTorch](#)
- [Perform Neural Style Transfer with PyTorch](#)

PyTorch Lightning

PyTorch Lightning is a community-driven version of PyTorch designed to flexibly perform deep learning tasks without affecting system performance. It supports multi-GPUs, TPUs, 16-bit and 32-bit precision floats to perform tasks such as:

- Training ML models on CPU or GPU clusters
- Evaluating Deep Learning model performance
- Improving model performance by discovering bottlenecks
- Reducing model sizes using quantization

To Install PyTorch Lightning in a Python environment, use a package manager such as `pip` or `conda`:

- Python Pip

CONSOLE

```
$ pip install lightning
```

- Conda

CONSOLE

```
$ conda install lightning -c conda-forge
```

For more information on how to use PyTorch Lightning, visit the [official documentation](#).

Keras

Keras is a high-level open-source deep-learning library for implementing neural networks. It's based on TensorFlow and packages many lower-level functions into single API endpoints. Keras includes modules to implement different types of neural network layers and combine layers to create models. This allows developers to choose from different optimizers, activation functions, and evaluation metrics. Keras is used in different applications such as:

- Natural Language Processing (NLP) projects like sentiment analysis
- Deep learning projects such as generative AI
- Building deep learning tools for mobile devices
- Image classification

To use Keras, import it from the TensorFlow package in your project

```
PYTHON
```

```
>>> from tensorflow import keras
```

Visit the following resources to implement Keras on your Vultr Cloud Server:

- [Keras Code Examples](#)
- [Implement Stable Diffusion using Keras and TensorFlow](#)

KerasCV

KerasCV is a horizontal extension of the Keras core library that includes modular Computer Vision (CV) components. Keras CV models are used to access many pretrained models such as Stable Diffusion and Vision Transformer to perform machine learning tasks such as:

- Computer Vision applications such as object detection and image segmentation
- Generative AI and image-processing applications

To install the latest KerasCV version in your Python environment, use the `pip` package manager and upgrade Tensorflow

CONSOLE

```
$ pip install keras-cv tensorflow --upgrade
```

When installed, import the package together with Keras core in your application

PYTHON

```
>>> import keras_cv
>>> import keras_core as keras
```

For more information, visit the [KerasCV documentation](#)

NumPy

Numerical Python (NumPy) is an open-source numerical scientific computing library used in Python applications. It supports linear algebra operations on high-dimensional vectors and tensors, performs mathematical operations like arithmetic, trigonometry, and complex numbers which makes it a necessary package for handling computation tasks. NumPy works directly on CPU systems with no direct support for GPUs, TPUs, or multi-GPU cluster systems to perform tasks such as:

- Machine Learning
- Generative AI
- Recommendation systems
- Imaging applications such as image classification and segmentation

To install NumPy, use a package manager such as Pip or Conda as described below:

- PIP:

CONSOLE

```
$ pip install numpy
```

- Conda:

```
CONSOLE
```

```
$ conda install numpy
```

Visit the following resources to implement NumPy:

- [NumPy user guide](#)
- [How to Use Stable Diffusion to Manipulate Images](#)

SciPy

Scientific Python (SciPy) is an open-source collection of libraries designed to perform mathematics, science, and engineering tasks built to extend NumPy functionalities. It uses sub-packages to perform low-level scientific computation tasks like numerical integration and differentiation, differential equations and eigenvalue problems, signal processing, and Fourier Transforms in applications such as:

- Statistical analysis
- Industrial optimization problems
- Audio processing
- Scientific computing applications, like modeling and simulations

You can install SciPy as a Python module, or use a package manager as described below:

- Python Module:

Debian/Ubuntu:

```
CONSOLE
```

```
$ sudo apt install python3-scipy
```

CentOS/ RHEL:

```
CONSOLE
```

```
$ sudo dnf install python3-scipy
```

- PIP:

```
CONSOLE
```

```
$ pip install scipy
```

- Conda:

```
CONSOLE
```

```
$ conda install scipy
```

For more information on how to use SciPy, visit the [official user guide](#).

JAX

JAX is a high-performance library for numerical computations and allows you to run NumPy functions on GPU and TPU systems. It shares the same syntax as NumPy, and supports complex low-level functions such as automatic and higher-order differentiation. However, JAX follows a pure functional programming paradigm and it's implemented in projects such as:

- Research in machine learning and deep learning
- Complex deep-learning applications

To install JAX, use the Python `pip` package manager and use the matching system CPU or GPU package

```
CONSOLE
```

```
$ pip install --upgrade "jax[cpu]"
```

On a GPU-based system, replace the `[cpu]` value with your GPU driver. For example, CUDA 12 and reference the releases file to install extra dependencies

```
CONSOLE
```

```
$ pip install --upgrade "jax[cuda11_pip]" -f https://storage.googleapis.com/jax-releases/jax_cuda_releases.html
```

For more information on how to use JAX, visit the [official developer documentation](#).

Flax

Flax is an open-source neural network library based on the JAX framework. It supports Neural network research and model development with the ability to use its API to:

- Construct neural network layers
- Train neural networks
- Check-point models
- Access pretrained and fine-tuned models

To install Flax, use the pip package manager

CONSOLE

```
$ pip install flax
```

For more information on how to use the package, visit the [official Flax documentation](#).

Scikit-learn

`scikit-learn` is a Python module designed for machine learning and data analysis tasks. It supports a variety of machine learning techniques such as classification and clustering, random forests, Support Vector Machines (SVMs), among others. Scikit-learn is built on top of NumPy, SciPy, and matplotlib with common usage in applications such as:

- Machine learning
- Statistical modeling
- Image recognition

Use a package manager such as Pip or Conda to install scikit-learn for use in your applications

- Pip:

CONSOLE

```
$ pip install -U scikit-learn
```

- Conda:

CONSOLE

```
$ conda install -c conda-forge scikit-learn
```

To build applications using scikit-learn, visit the following resources:

- [scikit-learn Developers Guide](#)
- [Build a Machine Learning Classifier in Python](#)

Pandas

`pandas` is a Python package that provides fast, extensive, and flexible data structures to create data analysis applications. It supports working with relational and labeled data with common usage in the following operations:

- Real-time applications that use dictionary-based data structures
- Pre-processing data for ML applications
- Viewing the data in multiple ways such as transposes and pivot tables
- Importing and exporting different file formats such as CSV, txt, JSON, among others

To install pandas, use a package manager such as Conda or Pip

- Conda:

CONSOLE

```
$ conda install -c conda-forge pandas
```

- Pip

```
CONSOLE
```

```
$ pip install pandas
```

To use pandas, import it to your application environment

```
PYTHON
```

```
>>> import pandas as pd
```

To use Panda on a Vultr Cloud Server, visit the following resources:

- [Pandas Documentation](#)
- [Deploy a Deep Learning Model with Streamlit](#)

Hugging Face

Hugging Face is an open-source community-driven platform that hosts ready-to-use models, datasets, and applications. Hugging Face Hub is the main platform repository that includes a large collection of machine learning and deep learning models. Using the platform, you can upload models, checkpoints, and datasets, or build projects using popular open-source models such as Transformers, Diffusers, Datasets, among others described in the following sections.

Transformers

Hugging Face Transformers is a machine learning package designed to work with PyTorch, TensorFlow, and Jax. It provides tools and APIs with access to over 25,000 pretrained models you can download and use in your project environment. You can use available models in projects such as:

- Natural Language Processing applications such as language modeling, text classification, and question answering
- Large Language Models (LLMs) applications such as Falcon and Llama

- Computer Vision applications such as object detection, segmentation, and classification
- Multimodal solutions such as character recognition, video classification, and visual question-answering
- Tools like tokenizers, pipelines, trainers, training configuration tools, data preprocessing tools, among others

Visit the following resources to implement Transformers on a Vultr Cloud GPU Server:

- [Transformers Documentation](#)
- [How to use Hugging Face Transformer Models on a Vultr Cloud GPU](#)
- [Implement the Falcon LLM](#)
- [Implement the Llama 2 LLM](#)
- [Build an Inference API Using Hugging Face Transformers and FastAPI](#)

Diffusers

Diffusers is a Hugging Face library that consists of pretrained diffusion models designed to perform image generation, audio, and 3D structure tasks. It consists of diffusion pipelines, noise schedulers, and pretrained models. Commonly, you can use the Diffusers library for tasks such as:

- Accessing pretrained image generation models like Stable Diffusion
- Accessing pretrained audio-diffusion models for audio generation
- Fine-tuning generative AI models
- Building generative AI models to serve using an API

To implement Diffusers in your project environment, visit the following resources:

- [Diffusers documentation](#)
- [How to Use Hugging Face Diffusion Models on Vultr Cloud GPU](#)
- [Text Guided Image to Image Generation with Stable Diffusion](#)
- [Deploy Dreambooth and Stable Diffusion](#)
- [Build an Inference API Using Hugging Face Diffusers and FastAPI](#)

Datasets

Hugging Face Datasets

Hugging Face Datasets is a library that allows you to access and share datasets for Audio, Computer Vision, and Natural Language Processing (NLP) tasks. The library streamlines the process of exporting and sharing datasets to use in your projects. Commonly, you can use the library for tasks such as:

- Importing datasets to fine-tune models
- Importing datasets to test model performance

To implement Hugging Face Datasets in your project, visit the following resources:

- [Hugging Face Datasets Documentation](#)
- [Fine-tune a Hugging Face Transformer Model](#)
- [Fine-tune a Hugging Face Diffuser Model](#)

TensorFlow Datasets

TensorFlow Datasets is a library that consists of read-to-use datasets for TensorFlow, Jax, NumPy and supported machine learning frameworks. It downloads and prepares data by constructing `tf.data.Dataset` or NumPy array `np.array`. You can apply TensorFlow datasets to:

- Access multiple publicly available datasets
- Pre-process training data
- Build and share new datasets to the TensorFlow Hub

To install the TensorFlow Datasets library, use the Python Pip package manager:

CONSOLE

```
$ pip install tensorflow-datasets
```

To use the datasets, import the package to your application environment:

PYTHON

```
>>> import tensorflow_datasets as tfds
```

For more information, visit the [TensorFlow Datasets documentation](#).

Plotting and Computer Vision Tools

Matplotlib

Matplotlib is a visualization Python library for creating static, animated, and interactive visualizations. You can use the package to:

- Create graphs, charts, and animations
- Create interactive visualizations
- Export the created illustrations in different formats

To install Matplotlib, either use the Python Pip package manager or Conda:

- Pip

CONSOLE

```
$ pip install -U matplotlib
```

- Conda

CONSOLE

```
$ conda install matplotlib
```

For more information and how to use Matplotlib, visit the [official documentation](#).

OpenCV

Open Source Computer Vision (OpenCV) is an open-source library that includes hundreds of computer vision algorithms optimized for real-time applications. It supports GPU-based operations and offers a Python API interface with a modular structure. You can use OpenCV in application tasks such as:

- Face detection and recognition
- Object tracking
- Gesture recognition
- Augmented Reality (AR)

The `cv2` package exposes the OpenCV Python API for use in applications. To implement the library, visit the following resources:

- [OpenCV documentation](#)
- <https://www.vultr.com/docs/how-to-install-opencv-on-centos-7/>
- [Human Face Detection with OpenCV](#)
- [How to Perform Facial Recognition in Python on a Vultr Cloud GPU](#)

Pillow

Pillow is a fork of the Python Imaging Library (PIL) imaging library that enhances the Python shell interpreter with basic image processing capabilities. It supports multiple file formats and internal data representation. You can apply Pillow in your environment to perform tasks such as:

- Opening images in different formats
- Pre-processing images for deep learning models
- Switching between image formats and NumPy arrays
- Fetching images from remote URLs

To install Pillow, use the Python Pip package manager

CONSOLE

```
$ pip install Pillow
```

To use Pillow in your application environment, visit the following resources:

- [Pillow Documentation](#)
- [Build a Super Resolution Image Server](#)
- [Text Guided Image to Image Generation](#)
- [AI Image Manipulation Using InstructPix2Pix](#)

PyTorch Image Models (TIMM)

PyTorch Image Models (TIMM) is a collection of image models, utilities, optimizers, and PyTorch image models designed for Computer Vision (CV) tasks. You can use TIMM to access:

- Neural network layers preconfigured for CV
- Optimizers and schedulers for training CV models
- Image-data preprocessing functions
- AI models for image processing tasks such as feature extraction and object detection

To install TIMM, verify that PyTorch is available on your system and use the Python Pip Package manager to add TIMM to your environment

CONSOLE

```
$ pip install timm
```

For more information on how to use TIMM, visit the [official documentation](#).

Albumentations

Albumentations is a flexible image augmentation machine learning library written in Python. Through the image augmentation process, it supports the modifications of existing images for use in tasks such as Computer vision models that require multiple images in a training set. Based on OpenCV, Albumentations is commonly used in tasks such as:

- Image augmentation using different techniques to modify the original images

- Image segmentation and classification
- Object detection

To install Albumentations in your environment, use the Pip package manager

CONSOLE

```
$ pip install -U albumentations
```

For more information and usage examples, visit the [Albumentations documentation](#).

Usability Tools

Accelerate

Accelerate is a library that enables the re-use of existing PyTorch code across any distributed configuration. This is important when running multiple CPUs, GPUs, or TPUs, the same PyTorch training code is re-used and can run on any system. Hence, the primary Accelerate use case is to make PyTorch code highly reusable.

To install Accelerate, use the Conda or the Python Pip package manager

- Conda:

CONSOLE

```
$ conda install -c conda-forge accelerate
```

- Pip

CONSOLE

```
$ pip install accelerate
```

To access Accelerate code samples, visit the [official documentation](#).

xFormers

xFormers is an open-source library that extends the Hugging Face Transformers library functionalities. It provides customizable building blocks to construct transformer models, and can be used to:

- Speed up the training of transformer and diffuser models
- Speed up inference processes, such as text and image generation
- Reduce memory consumption in transformer-based models
- Enhance the performance of diffuser models by optimizing attention blocks

To install xFormers in your environment, either use Conda or Pip.

- Conda:

CONSOLE

```
$ conda install xformers -c xformers
```

- Pip

CONSOLE

```
$ pip3 install -U xformers --index-url https://download.pytorch.org/whl/cu121
```

For more information about xFormers, visit the [project repository](#).

Einops

Einops is a powerful library designed to handle the notation of tensor operators. Tensors are the building blocks of deep learning models while Numerical operations involve complex mathematical rules. Tensor variables are written with superscripts, subscripts, and other notational artifacts. These notations are

not always consistent across different publishers, packages, and programmers, but can be normalized with Einops to perform tasks such as:

- Use consistent notational conventions for tensor representation and operations across NumPy, PyTorch, TensorFlow, JAX, and other major packages
- Ensure that code involving tensor operations is readable and error-free

To install the Einops library, use the Python Pip package manager

CONSOLE

```
$ pip install einops
```

For more information on how to use Einops, visit the [official documentation](#).

Evaluate

Evaluate is a machine learning and datasets evaluation library that enables the scrutinization of Natural Language Processing (NLP), Computer Vision, and Reinforcement learning tasks. The package enables the use of a single line of code to evaluate models consistently.

To install Evaluate, use the Python Pip package manager

CONSOLE

```
$ pip install evaluate
```

For more information and code samples, visit the [Evaluate documentation](#).

Development Environments

Jupyter Notebook

Jupyter Notebook is a self-hosted interactive web application for creating and sharing computational documents that may contain code, descriptions,

visualization data, illustrations, or control elements. It supports common programming languages such as Python, Julia, and R. By usage, Jupyter Notebook allows you to:

- Create, edit, and save computational documents
- Execute individual commands and programs
- Share generated notebooks for collaboration

To install Jupyter Notebook on your system, visit the following resources:

- [Jupyter Notebook Documentation](#)
- [How to Install Jupyter Notebook on a Vultr Ubuntu 16.04 Server](#)

JupyterLab

JupyterLab is a self-hosted web-based Interactive Development Environment (IDE) that supports computational documents and development. The IDE supports file formats such as `CSV`, `JSON`, `markdown`, among others, and offers ease-of-use features such as:

- Visual debugging and code inspection
- Opening multiple tabs in a single session
- Split views
- Theming and customization options

To install and use JupyterLab, visit the following resources:

- [Jupyter Lab Documentation](#)
- [How to Set Up a Jupyter Lab Environment on Ubuntu 22.04](#)

Conclusion

You have explored common machine learning and artificial intelligence packages. Depending on your project needs, there are many libraries you can deploy to extend your application functionalities. For the best performance, it's recommended to run packages on a GPU-based instance such as a [Vultr NVIDIA A100 Cloud server](#).



VULTR

