Best Python libraries for Machine Learning

Last Updated : 15 Nov, 2024

Machine learning has become an important component in various fields, enabling organizations to analyze data, make predictions, and automate processes. **Python** is known for its simplicity and versatility as it offers a wide range of libraries that facilitate machine learning tasks. These libraries allow developers and data scientists to quickly and effectively implement complex algorithms. By using Python's tools, users can efficiently tackle machine learning projects and achieve better results.

Best Python libraries for Machine Learning

In this article, we'll dive into the ***Best Python libraries for Machine Learning***, exploring how they facilitate various tasks like data preprocessing, model building, and evaluation. Whether you are a beginner just getting started or a professional looking to optimize workflows, these libraries will help you leverage the full potential of Machine Learning with Python.

Python libraries for Machine Learning

Here's a list of some of the **best Python libraries for Machine Learning** that streamline development:

1. Numpy

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow uses NumPy internally for manipulation of Tensors.

Example: Linear Algebra Operations

Python

```python
import numpy as np
# Create a feature matrix (X) and target vector (y)
X = np.array([[1, 2], [3, 4], [5, 6]])
y = np.array([1, 2, 3])

# Calculate the mean of each feature
mean = np.mean(X, axis=0)
print("Mean of features:", mean)
```

Output:

```
Mean of features: [3. 4.]
```

2. Pandas

Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training.

- In this case, Pandas comes handy as it was developed specifically for data extraction and preparation.
- It provides high-level data structures and wide variety tools for data analysis. It provides many inbuilt methods for grouping, combining and filtering data.

Example: Data Cleaning and Preparation

Python

```python
import pandas as pd

# Create a DataFrame with missing values
data = {
    'Country': ['Brazil', 'Russia', 'India', None],
    'Population': [200.4, 143.5, None, 52.98]
}
df = pd.DataFrame(data)

# Fill missing values
df['Population'].fillna(df['Population'].mean(), inplace=True)
print(df)
```

Output:

```
   Country  Population
0   Brazil      200.40
1   Russia       143.50
2   India      132.99
3    None       52.98
```

3. Matplotlib

Matplotlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots.

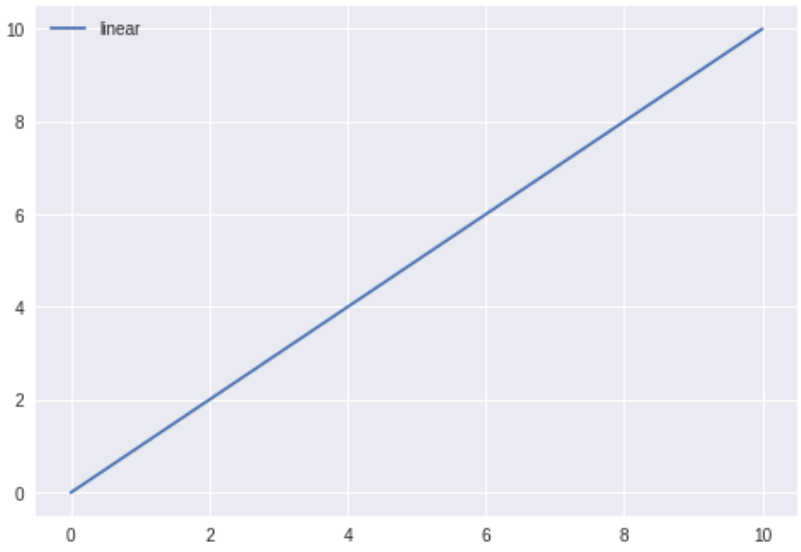
- A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc.
- It provides various kinds of graphs and plots for data visualization, viz., histogram, error charts, bar chats, etc,

Example: Creating a linear Plot

Python

```python
#  Python program using Matplotlib
# for forming a linear plot

# importing the necessary packages and modules
import matplotlib.pyplot as plt
import numpy as np

# Prepare the data
x = np.linspace(0, 10, 100)

# Plot the data
plt.plot(x, x, label ='linear')

# Add a legend
plt.legend()

# Show the plot
plt.show()
```

Output:



Output

4. SciPy

SciPy is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for image manipulation.

Example: Image Manipulation

Python

```python
# Python script using Scipy
# for image manipulation

from scipy.misc import imread, imsave, imresize

# Read a JPEG image into a numpy array
img = imread('D:/Programs / cat.jpg') # path of the image
print(img.dtype, img.shape)

# Tinting the image
img_tint = img * [1, 0.45, 0.3]

# Saving the tinted image
imsave('D:/Programs / cat_tinted.jpg', img_tint)

# Resizing the tinted image to be 300 x 300 pixels
img_tint_resize = imresize(img_tint, (300, 300))

# Saving the resized tinted image
imsave('D:/Programs / cat_tinted_resized.jpg',
img_tint_resize)
```

If scipy.misc import imread, imsave,imresize does not work on your operating system then try below code instead to proceed with above code

```
!pip install imageio
import imageio
from imageio import imread, imsave
```

Original image:

Tinted image:

Resized tinted image:

5. Scikit-Learn

Scikit-learn is one of the most popular ML libraries for classical ML algorithms. It is built on top of two basic Python libraries, viz., NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms. Scikit-learn can also be used for data-mining and data-analysis, which makes it a great tool who is starting out with ML.

Example: Decision Tree Classifier

Python

```python
# Import necessary libraries
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier

# Load the iris dataset
iris = datasets.load_iris()

# Split the dataset into features (X) and target labels (y)
X = iris.data   # Features (sepal length, sepal width, petal length, petal width)
y = iris.target # Target (species)

# Initialize the Decision Tree Classifier
clf = DecisionTreeClassifier()

# Train the model on the entire dataset
```

```
clf.fit(X, y)

# Make predictions on the same dataset
predictions = clf.predict(X)

# Print the first 10 predictions
print("Predicted labels for the first 10 samples:", predictions[:10])

# Print the actual labels for comparison
print("Actual labels for the first 10 samples:", y[:10])
```

Output:

```
Predicted labels for the first 10 samples: [0 0 0 0 0 0 0 0 0 0]
Actual labels for the first 10 samples: [0 0 0 0 0 0 0 0 0 0]
```

6. Theano

We all know that Machine Learning is basically mathematics and statistics. Theano is a popular python library that is used to define, evaluate and optimize mathematical expressions involving multi-dimensional arrays in an efficient manner.

- It is achieved by optimizing the utilization of CPU and GPU. It is extensively used for unit-testing and self-verification to detect and diagnose different types of errors.
- Theano is a very powerful library that has been used in large-scale computationally intensive scientific projects for a long time but is simple and approachable enough to be used by individuals for their own projects.

Example

Python

```
# Python program using Theano
# for computing a Logistic
# Function

```

```
import theano
import theano.tensor as T
x = T.dmatrix('x')
s = 1 / (1 + T.exp(-x))
logistic = theano.function([x], s)
logistic([[0, 1], [-1, -2]])
```

Output:

```
array([[0.5, 0.73105858],
       [0.26894142, 0.11920292]])
```

7. TensorFlow

TensorFlow is a very popular open-source library for high performance numerical computation developed by the Google Brain team in Google. As the name suggests, Tensorflow is a framework that involves defining and running computations involving tensors. It can train and run deep neural networks that can be used to develop several AI applications. TensorFlow is widely used in the field of deep learning research and application.

Example

Python

```
#  Python program using TensorFlow
#  for multiplying two arrays

# import `tensorflow`
import tensorflow as tf

# Initialize two constants
x1 = tf.constant([1, 2, 3, 4])
x2 = tf.constant([5, 6, 7, 8])

# Multiply
result = tf.multiply(x1, x2)

# Initialize the Session
```

```
sess = tf.Session()

# Print the result
print(sess.run(result))

# Close the session
sess.close()
```

Output:

```
[ 5 12 21 32]
```

8. Keras

Keras is a very popular ***Python Libaries for Machine Learning*** . It is a high-level neural networks API capable of running on top of TensorFlow, CNTK, or Theano. It can run seamlessly on both CPU and GPU. Keras makes it really for ML beginners to build and design a Neural Network. One of the best thing about Keras is that it allows for easy and fast prototyping.

Example

Python

```python
# Importing necessary libraries
from keras.models import Sequential
from keras.layers import Dense, Flatten
from keras.datasets import mnist
from keras.utils import to_categorical

# Loading the MNIST dataset
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# Normalizing the input data
X_train = X_train / 255.0
X_test = X_test / 255.0

# One-hot encoding the labels
y_train = to_categorical(y_train, 10)
```

```
y_test = to_categorical(y_test, 10)

# Building the model
model = Sequential()
model.add(Flatten(input_shape=(28, 28)))  # Flatten the 2D images into 1D vectors
model.add(Dense(128, activation='relu'))  # Hidden layer with ReLU activation
model.add(Dense(10, activation='softmax'))  # Output layer with Softmax for classification

# Compiling the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Training the model
model.fit(X_train, y_train, epochs=5, batch_size=32, validation_split=0.2)

# Evaluating the model
test_loss, test_accuracy = model.evaluate(X_test, y_test)
print(f"Test Accuracy: {test_accuracy:.4f}")
```

Output:

```
Epoch 1/5
1500/1500 [==============================] - 4s 2ms/step - loss: 0.2941
- accuracy: 0.9163 - val_loss: 0.1372 - val_accuracy: 0.9615
Epoch 2/5
1500/1500 [==============================] - 3s 2ms/step - loss: 0.1236
- accuracy: 0.9647 - val_loss: 0.1056 - val_accuracy: 0.9697
...
Test Accuracy: 0.9765
```

9. PyTorch

PyTorch is a popular open-source ***Python Library for Machine Learning*** based on Torch, which is an open-source Machine Learning library that is implemented in C with a wrapper in Lua. It has an extensive choice of tools and libraries that support Computer Vision, Natural Language Processing(NLP), and

many more ML programs. It allows developers to perform computations on Tensors with GPU acceleration and also helps in creating computational graphs.

Example

Python

```python
# Python program using PyTorch
# for defining tensors fit a
# two-layer network to random
# data and calculating the loss

import torch


dtype = torch.float
device = torch.device("cpu")
# device = torch.device("cuda:0") Uncomment this to run on GPU

# N is batch size; D_in is input dimension;
# H is hidden dimension; D_out is output dimension.
N, D_in, H, D_out = 64, 1000, 100, 10

# Create random input and output data
x = torch.random(N, D_in, device=device, dtype=dtype)
y = torch.random(N, D_out, device=device, dtype=dtype)

# Randomly initialize weights
w1 = torch.random(D_in, H, device=device, dtype=dtype)
w2 = torch.random(H, D_out, device=device, dtype=dtype)

learning_rate = 1e-6
for t in range(500):
    # Forward pass: compute predicted y
    h = x.mm(w1)
    h_relu = h.clamp(min=0)
    y_pred = h_relu.mm(w2)

    # Compute and print loss
    loss = (y_pred - y).pow(2).sum().item()
```

```
        print(t, loss)

        # Backprop to compute gradients of w1 and w2 with
    respect to loss
        grad_y_pred = 2.0 * (y_pred - y)
        grad_w2 = h_relu.t().mm(grad_y_pred)
        grad_h_relu = grad_y_pred.mm(w2.t())
        grad_h = grad_h_relu.clone()
        grad_h[h < 0] = 0
        grad_w1 = x.t().mm(grad_h)

        # Update weights using gradient descent
        w1 -= learning_rate * grad_w1
        w2 -= learning_rate * grad_w2
```

Output:

```
0 47168344.0
1 46385584.0
2 43153576.0
...
...
...
497 3.987660602433607e-05
498 3.945609932998195e-05
499 3.897604619851336e-05
```

Conclusion

In summary, Python's versatility, simplicity, and vast ecosystem make it a go-to choice for Machine Learning tasks. From Scikit-Learn for classical algorithms to TensorFlow and PyTorch for deep learning, Python libraries cater to every stage of the Machine Learning workflow. Libraries like Pandas and NumPy streamline data preprocessing, while Matplotlib and Seaborn aid in data visualization. Specialized tools such as NLTK, XGBoost, and LightGBM further enhance the ability to solve complex problems efficiently.

"This course is very well structured and easy to learn. Anyone with zero experience of data science, python or ML can learn from this. This course

R Rahul... 155

Previous Article

7 Best R Packages for Machine Learning

Next Article

ML | Introduction to Data in Machine Learning

Similar Reads

Getting started with Machine Learning || Machine Learning Roadmap

Machine Learning (ML) represents a branch of artificial intelligence (AI) focused on enabling systems to learn from data, uncover patterns, and autonomously...

11 min read

Top 10 Javascript Libraries for Machine Learning and Data Science

JavaScript is the programming language of the web which makes it pretty important! However, it has mostly been used as a scripting language in web...

6 min read

Top 5 Programming Languages and their Libraries for Machine Learning in...

If you are a newbie in machine learning you may have thought that what programming language should I learn? Nowadays different people are working...

6 min read

Java libraries for machine learning

Java, known for its robustness and performance, is a powerful choice for implementing machine learning (ML) solutions. Although Python is widely...

5 min read

C++ Libraries for Machine Learning

Machine learning (ML) has significantly transformed various industries by enabling systems to learn from data and make predictions. While Python is ofte...

5 min read

Top 7 Python Libraries For Reinforcement Learning

Reinforcement Learning (RL) has gained immense popularity due to its applications in game playing, robotics, and autonomous systems. Python, being...

5 min read

Support vector machine in Machine Learning

In this article, we are going to discuss the support vector machine in machine learning. We will also cover the advantages and disadvantages and application...

9 min read

Machine Learning Model with Teachable Machine

Teachable Machine is a web-based tool developed by Google that allows users to train their own machine learning models without any coding experience. It uses ...

7 min read

Artificial intelligence vs Machine Learning vs Deep Learning

Nowadays many misconceptions are there related to the words machine learning, deep learning, and artificial intelligence (AI), most people think all these things...

4 min read

Need of Data Structures and Algorithms for Deep Learning and Machine…

Deep Learning is a field that is heavily based on Mathematics and you need to have a good understanding of Data Structures and Algorithms to solve the...

6 min read

Article Tags : AI-ML-DS AI-ML-DS Blogs Technical Scripter Machine Learning Blogs

+1 More