A Technical overview of PerfXPy®

(Python high-performance IDE)

Document Version：V1.0

Release Date：2023/3/1

Content

Introduce 3

What is PerfXPy 4

Legal Information 5

PerfXPy Version and Download 6

PerfXPy Interface and Functionality 7

PerfXPy Performance Testing 9

PerfXPy Demonstration 10

PerfXPy Ecology 16

Introduce

Python was designed by Guido van Rossum of the Netherlands Research Society for Mathematics and Computer Science in the early 1990s as an alternative to a language called ABC. Python provides efficient high-level data structures and simple and effective object-oriented programming. Python syntax and dynamic types, as well as the nature of interpreted languages, make it a programming language for scripting and rapid application development on most platforms. With the continuous updating of versions and the addition of new language features, it is gradually used for independent project development.

Due to the simplicity, readability and scalability of the Python language, it is widely used in research institutions for **scientific computing**. Computer and non-computer majors in universities at home and abroad use Python to teach programming courses. With the advent of the era of artificial intelligence, Python is also increasingly widely used in the field of **artificial intelligence** due to its open source, rich tool library and the promotion of international large enterprises.

On the one hand, although the Python system has many excellent foreign IDE tools such as PyCharm, Anaconda, PyDev, Wing, VSCode, Jupyter, and Atom. However, the support for the new generation of computer architecture is slow (such as ARM ecology, RISC-V ecology), lack of support for heterogeneous computing systems such as CPU+GPU, CPU+NPU, especially the lack of support for Chinese computing chip platforms .

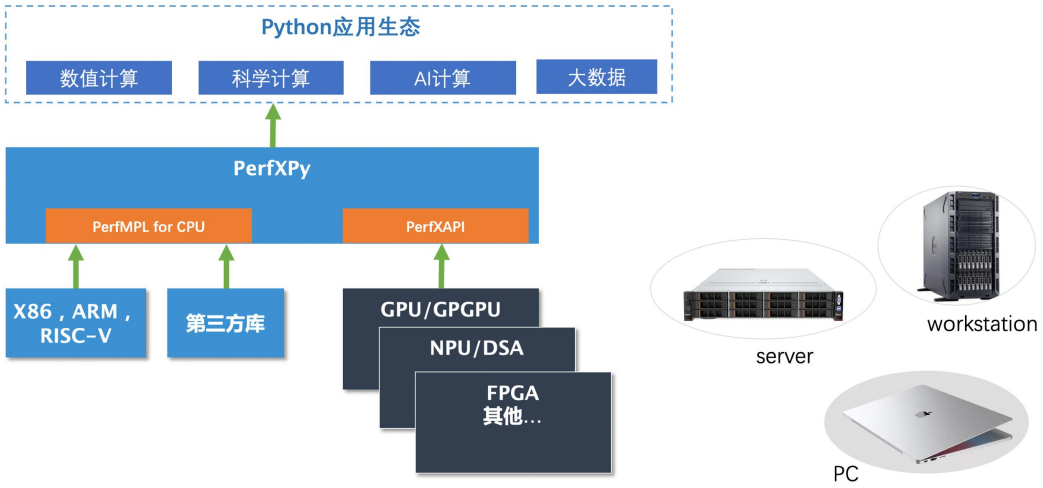
PerfXPy aims to provide a more complete Python integrated development environment for the new generation of computer architecture, focusing on the improvement of computing efficiency and computing performance. Promote the Python language to replace MATLAB in the field of scientific computing; promote the Python language to give full play to heterogeneous computing performance in the field of artificial intelligence; provide students with a free and excellent learning programming environment.

This document is intended as an introductory technical overview of PerfXPy® developed by PerfXLab.

What is PerfXPy

PerfXPy® is a Python language programming integrated development environment for the development of a new generation of computer architecture. There are three main features:

1. **The operating efficiency is close to C/C++ language.** The core mathematical computing library adopts PerfMPL independently developed by PerfXLab. PerfMPL provides high-performance and comprehensive mathematical computing capabilities for x86, ARM, and RISC-V instruction set CPUs (refer to intel MKL), and supports GPU, NPU, etc. acceleration Card operator library extension. For details, please refer to "PerfMPL® Technical White Paper"(download link: https://pan.baidu.com/s/12MCsQrjPgkXxDXE_xrWLyg, extraction code: jodl)
2. **Support high-performance heterogeneous computing.** PerfXPy integrates PerfXAPI independently developed by PerfXLab, which can quickly support various heterogeneous computing hardware, such as ARM CPU+xPU, RISC-V CPU+xPU (xPU refers to GPU, NPU and other DSAs).
3. **Perfect Python development environment.** PerfXPy has a complete integrated development environment (IDE, Integrated Development Environment), including code editors, compilers, debuggers and graphical user interface tools), which fully supports users from learning Python programming, algorithm design to large-scale project development.



Legal Information

Trademark Statement

PerfMPL® is a trademark of PerfXLab (Beijing) Technologies Co., Ltd.

All other trademarks or registered trademarks mentioned in this document are owned by their respective owners.

PerfXLab

Address: Room 304, Cuihu Science and Technology Innovation Platform, Building 9, Yard 55, Zique Road, Haidian District, Beijing

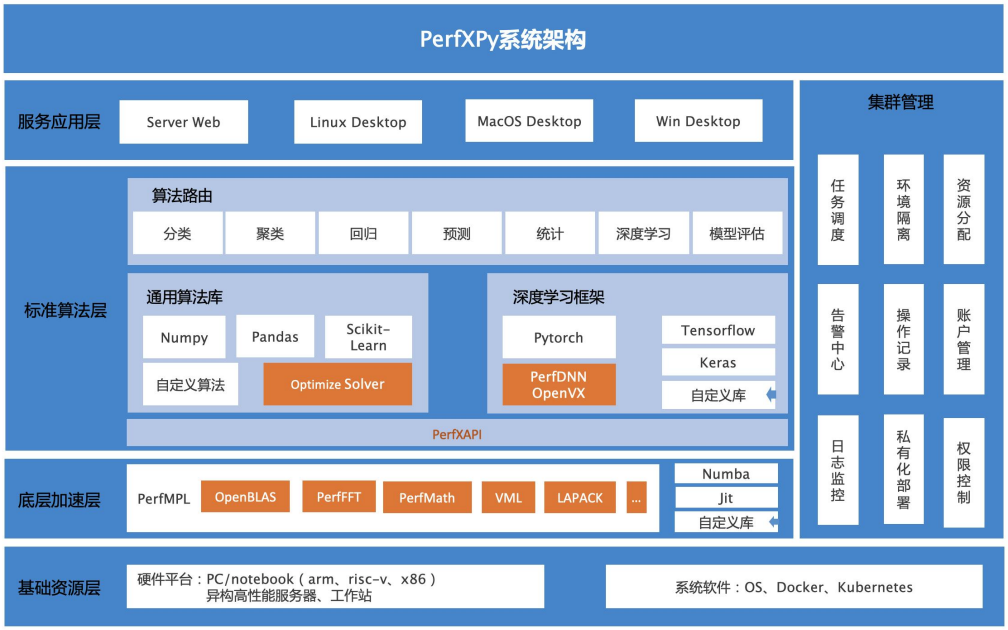
E-Mail: xianyi@perfxlab.com

Chinese Website: www.perfxlab.cn, www.perfxlab.com

PerfXPy Version and Download

There are three versions of PerfXPy:

1. **The stand-alone version is free for community fans.** It can run on personal computers, notebooks, and some development board hardware platforms, and supports Windows, Linux, OSX and other operating systems.
2. **The SaaS version is charged for business users.** Can run on computing server platform, support multi-user remote login.
3. **The cluster version is charged for commercial users.** Can run on high-performance computing clusters, support multi-user login, and rich management tools.

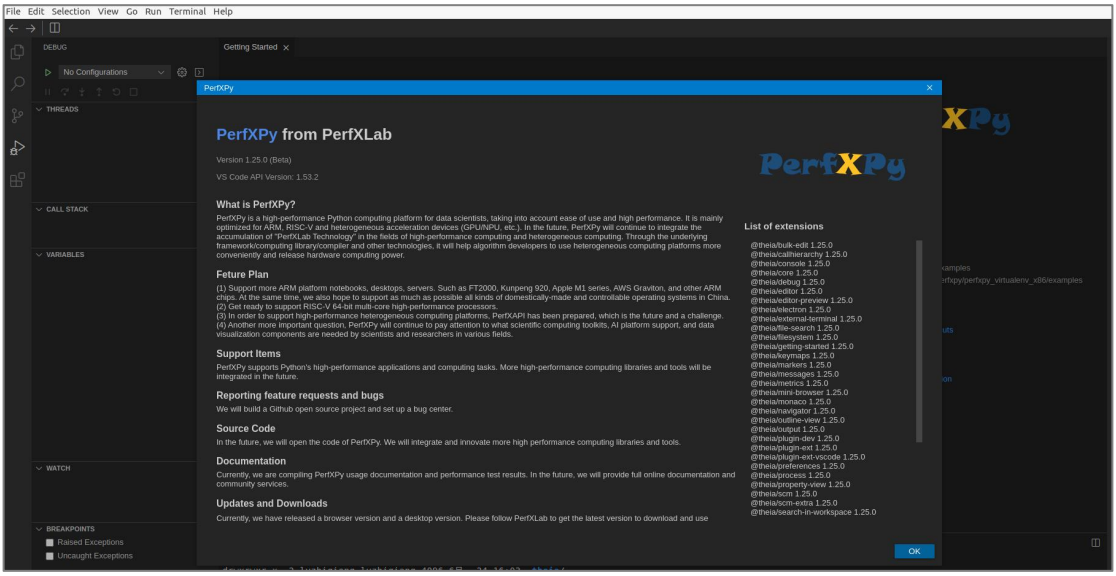


Installation package download: https://gitee.com/PerfXLab/perfxpy

Demo video: https://space.bilibili.com/1444176265

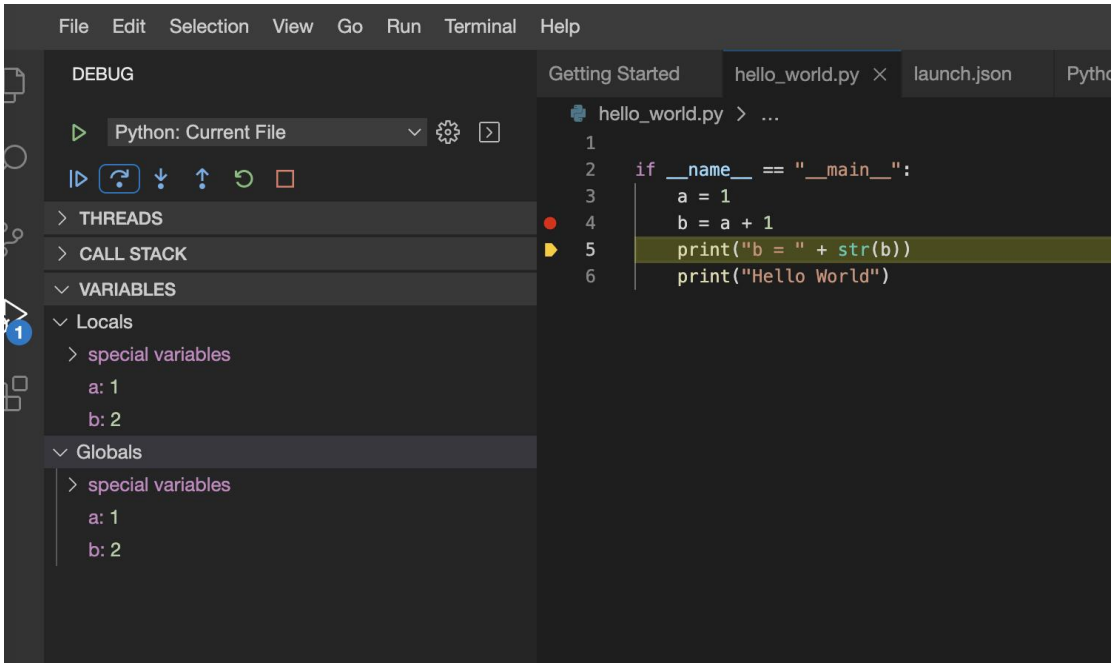
PerfXPy Interface and Functionality

1. Start Interface

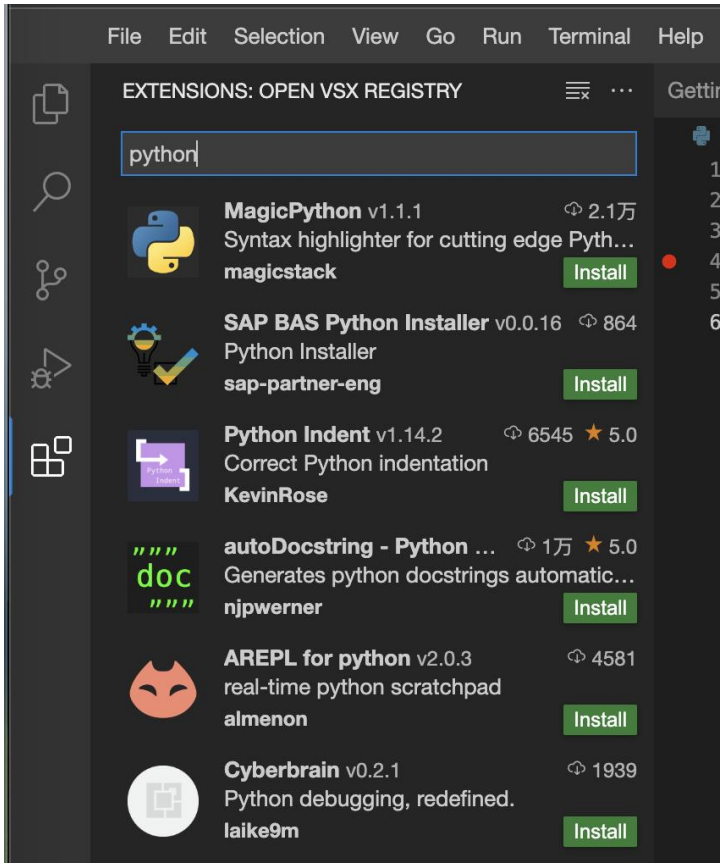


2. Run and Debug

The PerfXPy platform supports code debugging. Users can set breakpoints and select Start Debugging in the RUN function. In debug mode, users can perform regular operations such as Continue, Step Over, Step Into, and Stop, and can also view special variables through variable tracking or WATCH functions. At the same time, you can also choose a different environment to quickly test the code.



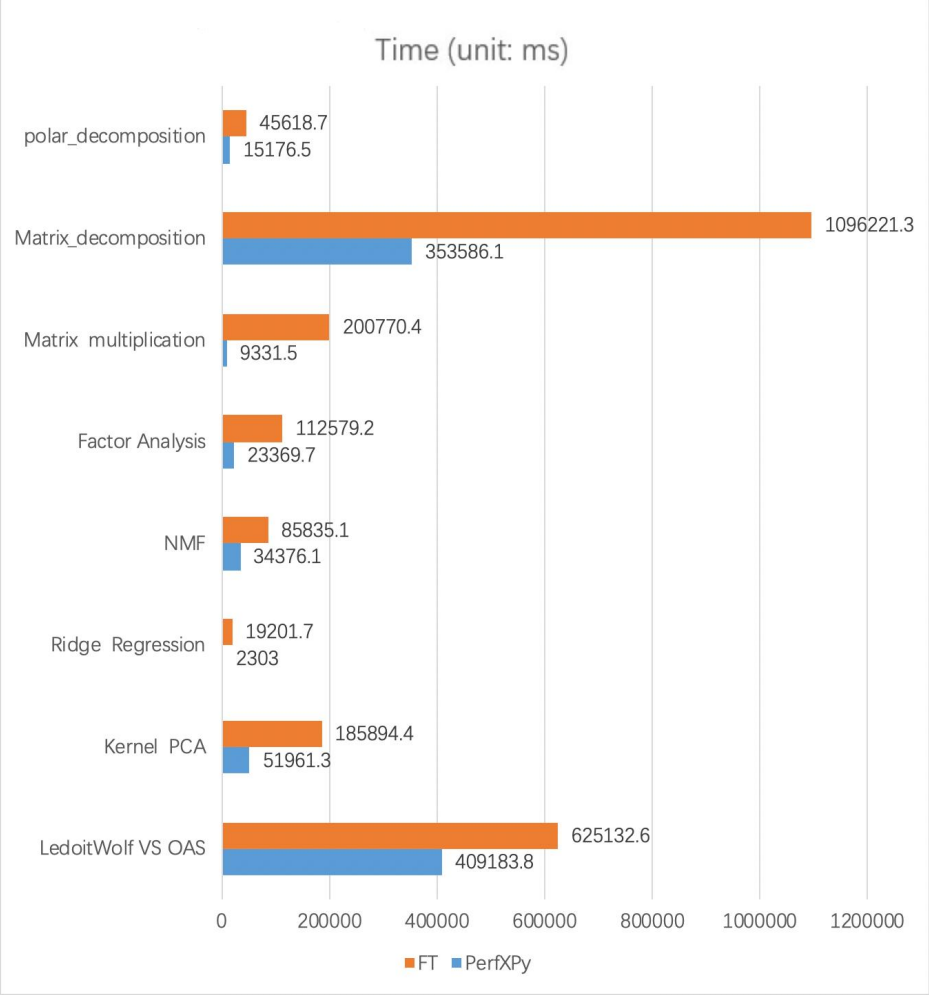
3. Plug-in installation and function extension



PerfXPy Performance Testing

Note: The following data only represent the performance comparison data during the test, and the test parameters are also related to the hardware platform and are for reference only.

- Basic test set 1:



polar_decompositon A pola decomposition is performed on the matrix

Matrix_decompostion The matrix is decomposed by LU and SVD

Matrix multiplication Optimizing for Higher-Order Matrix Multiplication

Factor Analysis Implement factor analysis on datasets

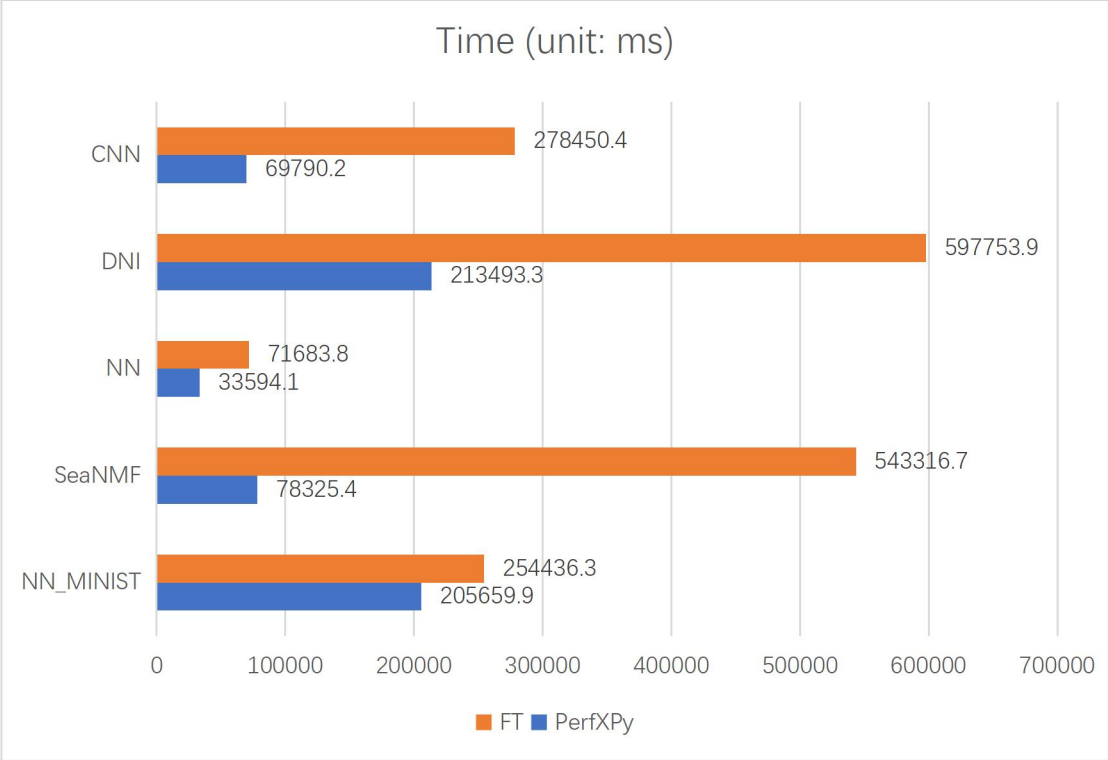
NMF Topic Modeling of Short Texts via Nonnegative Matrix Factorization

Ridge Regression Train Ridge Regression Model Inference

Kernel PCA Transform data using PCA algorithm and kernel-PCA algorithm

LedotiWolf VS OAS Comparing the MSE metrics of LW and OAS methods using Gaussian distributed data

- Basic test set 2



Note:

CNN CNN-based cifar10 image classification

DNI Classification and recognition of diabetic patients based on decoupled neural network

NN Classification and recognition of diabetic patients based on fully connected neural network

SeaNMF Semantic recognition of short texts based on neural network non-negative matrix factorization

NN_MNIST Classification of Handwritten Digits Based on Neural Network

PerfXPy Demonstration

1. Scientific Computing Examples

This is an example of a demonstration of the Mean Shift algorithm. In this example, we use the Mean Shift algorithm to perform cluster analysis on 10,000 sample points, and use matplotlib to draw the clustered image.

```
import numpy as np
from sklearn.cluster import MeanShift, estimate_bandwidth
from sklearn.datasets import make_blobs
```

Generate Sample Data

```
centers = [[1, 1], [-1, -1], [1, -1]]
X, _ = make_blobs(n_samples=10000, centers=centers, cluster_std=0.6)
```

Mean Shift Cluster Calculation

```
# The following bandwidth can be automatically detected using
bandwidth = estimate_bandwidth(X, quantile=0.2, n_samples=500)
ms = MeanShift(bandwidth=bandwidth, bin_seeding=True)
ms.fit(X)
labels = ms.labels_
cluster_centers = ms.cluster_centers_

labels_unique = np.unique(labels)
n_clusters_ = len(labels_unique)

print("number of estimated clusters : %d" % n_clusters_)
```

Clustering Result Calculation

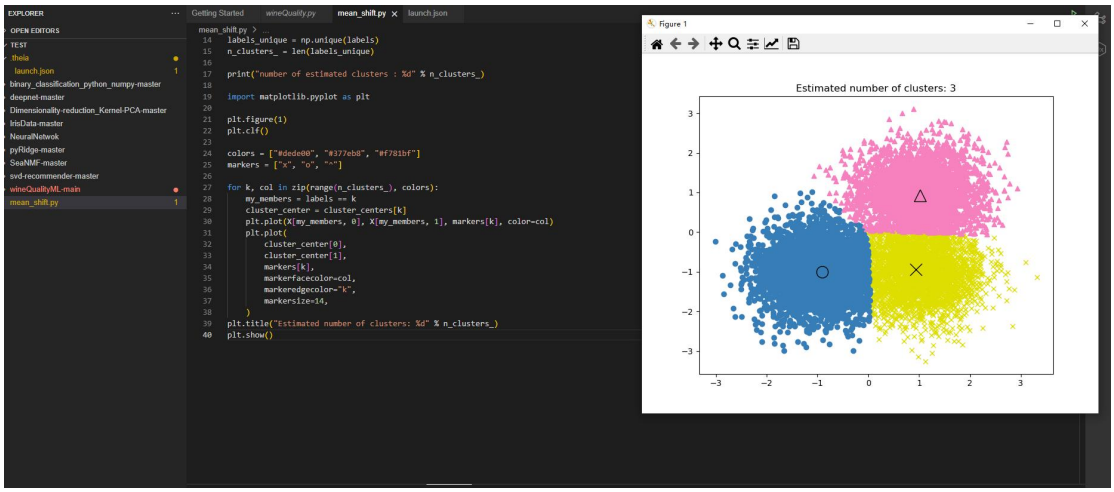
```
import matplotlib.pyplot as plt

plt.figure(1)
plt.clf()

colors = ["#dede00", "#377eb8", "#f781bf"]
markers = ["x", "o", "^"]

for k, col in zip(range(n_clusters_), colors):
```

```
my_members = labels == k
cluster_center = cluster_centers[k]
plt.plot(X[my_members, 0], X[my_members, 1], markers[k], color=col)
plt.plot(
cluster_center[0],
 cluster_center[1],
        markers[k],
        markerfacecolor=col,
        markeredgecolor="k",
        markersize=14,
)
plt.title("Estimated number of clusters: %d" % n_clusters_)
plt.show()
```



Here is an example of regression with a multi-output decision tree. In this example, a decision tree is used to simultaneously predict noisy x and y observations of a circle given a single underlying feature. Finally, use matplotlib (Python's 2D drawing library) to draw the regression image of the local linear approximation circle of the decision tree.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor
```

Dataset Generation

```
rng = np.random.RandomState(1)
X = np.sort(200 * rng.rand(100, 1) - 100, axis=0)
```

```
y = np.array([np.pi * np.sin(X).ravel(), np.pi * np.cos(X).ravel()]).T
y[::5, :] += 0.5 - rng.rand(20, 2)
```

Model Training

```
regr_1 = DecisionTreeRegressor(max_depth=2)
regr_2 = DecisionTreeRegressor(max_depth=5)
regr_3 = DecisionTreeRegressor(max_depth=8)
regr_1.fit(X, y)
regr_2.fit(X, y)
regr_3.fit(X, y)
```

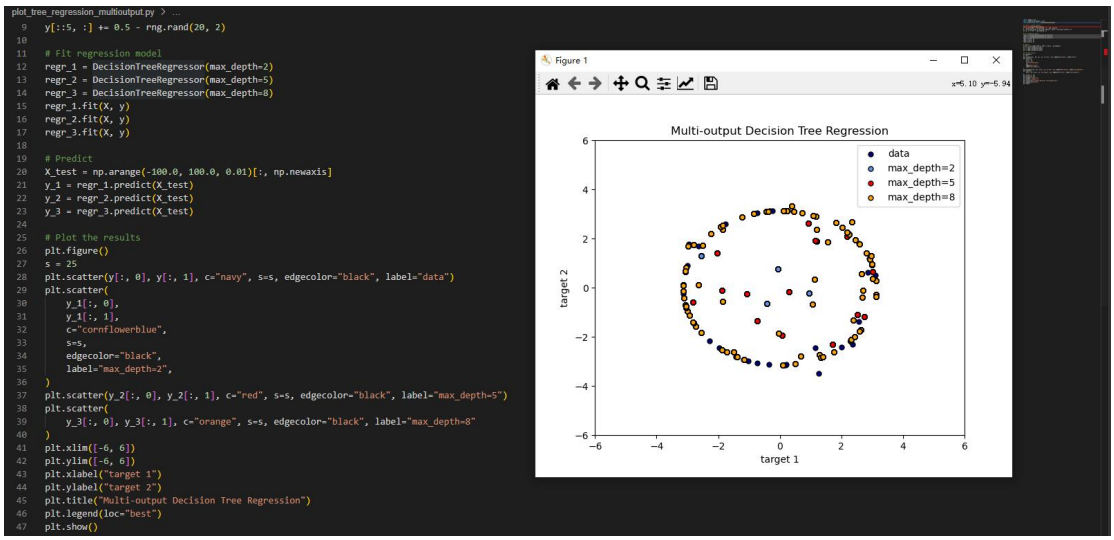
预测

```
X_test = np.arange(-100.0, 100.0, 0.01)[:, np.newaxis]
y_1 = regr_1.predict(X_test)
y_2 = regr_2.predict(X_test)
y_3 = regr_3.predict(X_test)
```

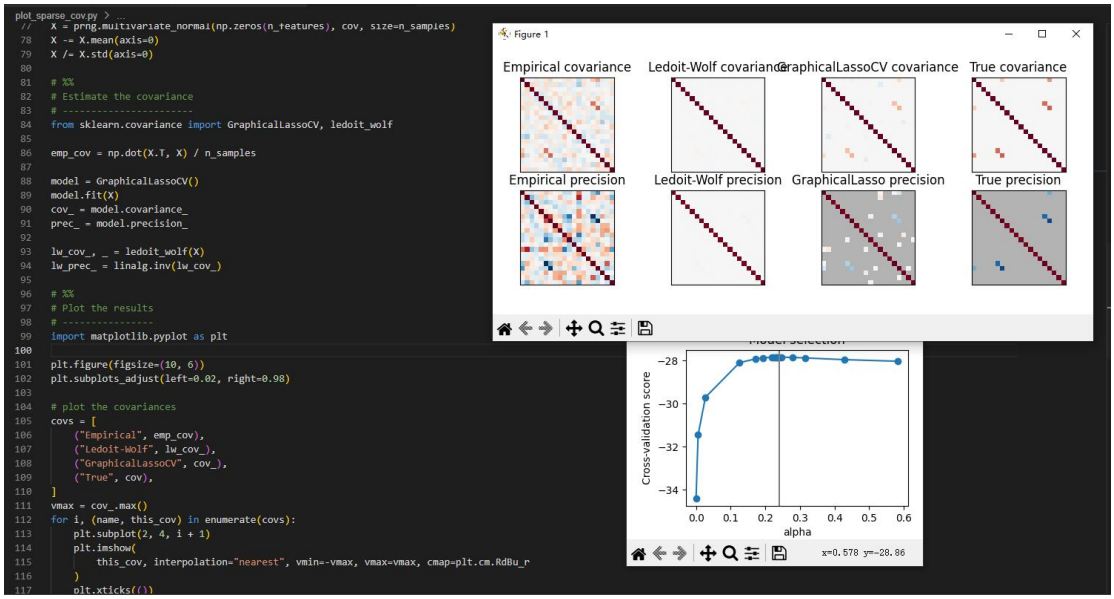
Plot The Resulting Image

```
plt.figure()
s = 25
plt.scatter(y[:, 0], y[:, 1], c="navy", s=s, edgecolor="black", label="data")
plt.scatter(
    y_1[:, 0],
    y_1[:, 1],
    c="cornflowerblue",
    s=s,
    edgecolor="black",
    label="max_depth=2",
)
plt.scatter(y_2[:, 0], y_2[:, 1], c="red", s=s, edgecolor="black",
label="max_depth=5")
plt.scatter(
    y_3[:, 0], y_3[:, 1], c="orange", s=s, edgecolor="black", label="max_depth=8"
```

```
)
plt.xlim([-6, 6])
plt.ylim([-6, 6])
plt.xlabel("target 1")
plt.ylabel("target 2")
plt.title("Multi-output Decision Tree Regression")
plt.legend(loc="best")
plt.show()
```

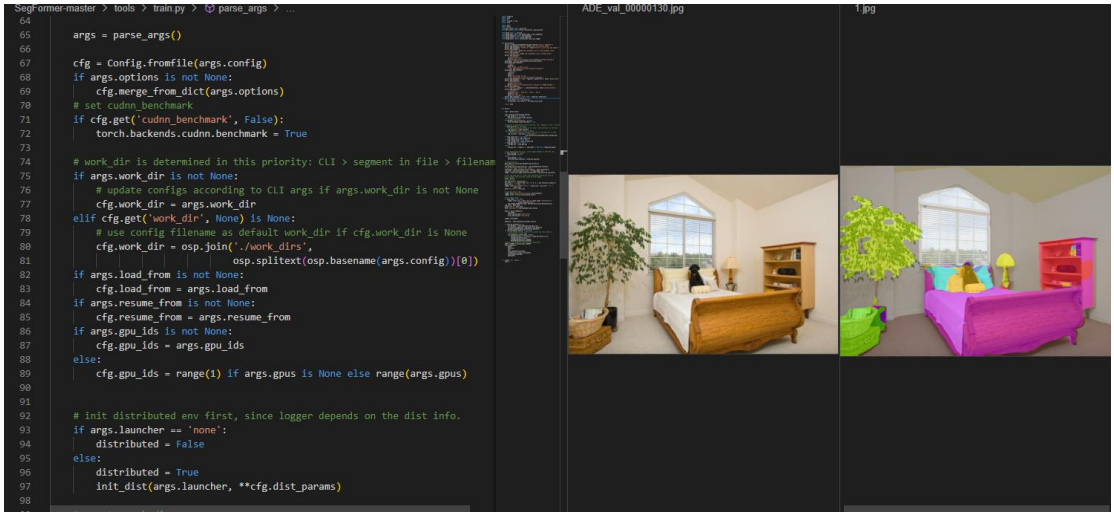


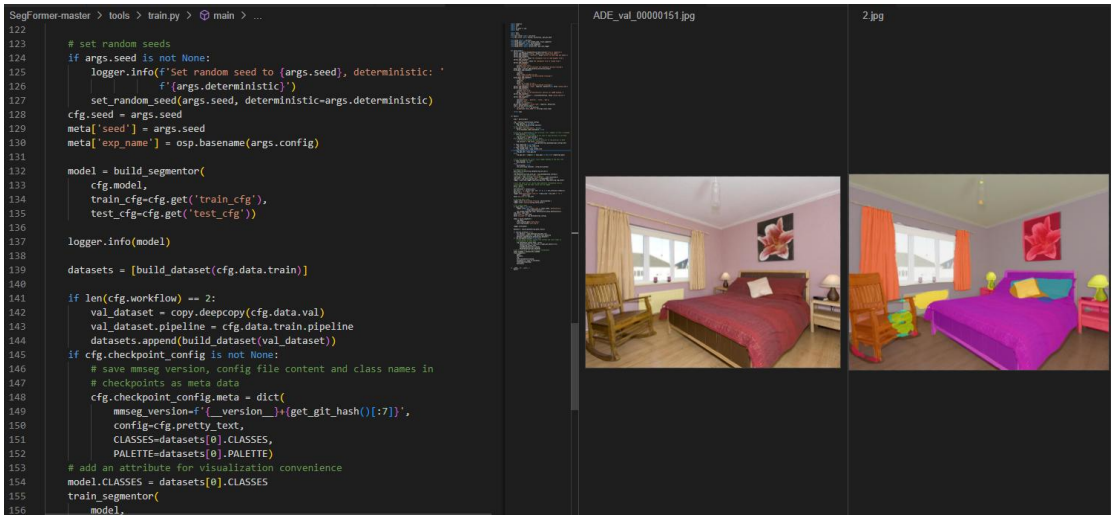
Here is an example of covariance estimation. In this example, the Ledoit-Wolf covariance estimation method and the GraphicalLassoCV estimation method are used to estimate the covariance of the data model, and the corresponding results are plotted.



2. AI Training - Inference Examples

This is an example of image segmentation. After completing the training of the neural network, we use it to graphically segment two images, the left side is the original image, and the right side is the result of image segmentation.





PerfXPy Ecology

PerfXPy is cooperating extensively with partners in computing chip ecology, operating system ecology, scientific research ecology, and education ecology, oriented to the new generation of computer architecture, and striving to build a world-class Python high-performance integrated development environment ecology originated from China's innovation.

1. PerfXPy has been running on the ARM processor platform and supports Apple M series, Phytium Feiteng processors, Huawei Kunpeng and other ARM processors.
2. PerfXPy has been running on the RISC-V processor platform, supports the RISC-V IP of Ali Pingtouge and Unisilicon Technology, and supports the RISC-V high-performance processor of Suaneng Technology (for example, SG2042 64 cores)
3. 3. PerfXPy already supports Linux operating systems such as ubuntu and Debian, and will be extended to domestic operating systems such as Kirin, Tongtongxin UOS, and openEuler.

---- End ----