溶解氧预测实验

目录

[1 概述 0](#_Toc199937464)

[1.1 案例介绍 0](#_Toc199937465)

[1.2 案例时间 0](#_Toc199937466)

[1.3 案例流程 0](#_Toc199937467)

[1.4 资源总览 0](#_Toc199937468)

[2 章节名称 1](#_Toc199937469)

[2.1 启用华为开发者空间 1](#_Toc199937470)

[2.2 下载数据 2](#_Toc199937471)

[2.3 编写代码 3](#_Toc199937472)

[2.4 运行代码 6](#_Toc199937473)

[2.5 获得结果 6](#_Toc199937474)

[3 释放资源 8](#_Toc199937475)

[3.1 关闭云服务器 8](#_Toc199937476)

# 概述

## 案例介绍

溶解氧（dissolved oxygen, DO）在水体中的含量能够反映出水体的污染程度、生物的生长状况，它是衡量水质优劣的重要指标之一。国内外相关文献表明溶解氧的含量受到多种因素的影响，如水温、pH值、生物种类等，同时直接或者间接影响着养殖生物的生长。因此，在水产养殖过程中，监测养殖水体溶解氧浓度，并预测其变化趋势具有重要意义。近年来，溶解氧预测方法的研究主要集中在时间序列预测、支持向量机、组合预测、人工神经网络等方面。时间序列预测方法基于溶解氧含量与其历史变化之间的关联进行预测，然而，由于溶解氧的变化受多种复杂因素影响，该方法仅考虑了预测变量与其历史变化之间的关联，未能充分纳入相关影响因子，因此其预测准确性相对较低。支持向量机作为一种针对有限样本设计的学习方法，在处理大规模数据集时，面临着算法效率低下、存在大量冗余支撑向量以及缺乏有效的参数选择策略等问题。相比之下，人工神经网络因其具备自学习、自组织、并行处理信息以及处理非线性信息的能力，能够深入挖掘数据背后难以用数学公式描述的非线性特征，从而有效弥补了传统时间序列模型的不足。正因如此，人工神经网络在溶解氧预测领域得到了广泛应用。

本案例使用课题组2016年4月5日～25日的水质监测数据来构建BP神经网络模型进

行预测，该数据由AP2000型多参数水质仪探测到的水温(TEMP)、酸碱度(pH)、氧化还原电位(ORP)、溶解氧(DO)、盐度(SLA)、浊度(TDS)和海水比重(SSG)共７项参数。水质检测仪每３分钟获取一组数据，共9600组。根据长时间监测数据表明，每小时内各项参数指标浮动范围很小，因此可以小时为单位，计算每小时各项参数的平均值，得到共计480组数据。

本案例借助开发者空间云主机提供的免费CodeArts IDE编辑器进行本地编辑函数、开发预测，直观地展示云上云下的开发与调试能力和实际应用开发中为开发者带来的便利。

## 案例时间

本案例总时长预计15分钟。

## 案例流程

## 资源总览

本案例预计花费总计**0**元。体验完成后请及时释放资源，避免产生多余的费用。

|  |  |  |  |
| --- | --- | --- | --- |
| **资源名称** | **规格** | **单价（元）** | **时长（分钟）** |
| [华为云主机](https://www.huaweicloud.com/product/ecs.html) | X86计算 | 0 | 15 |

# 章节名称

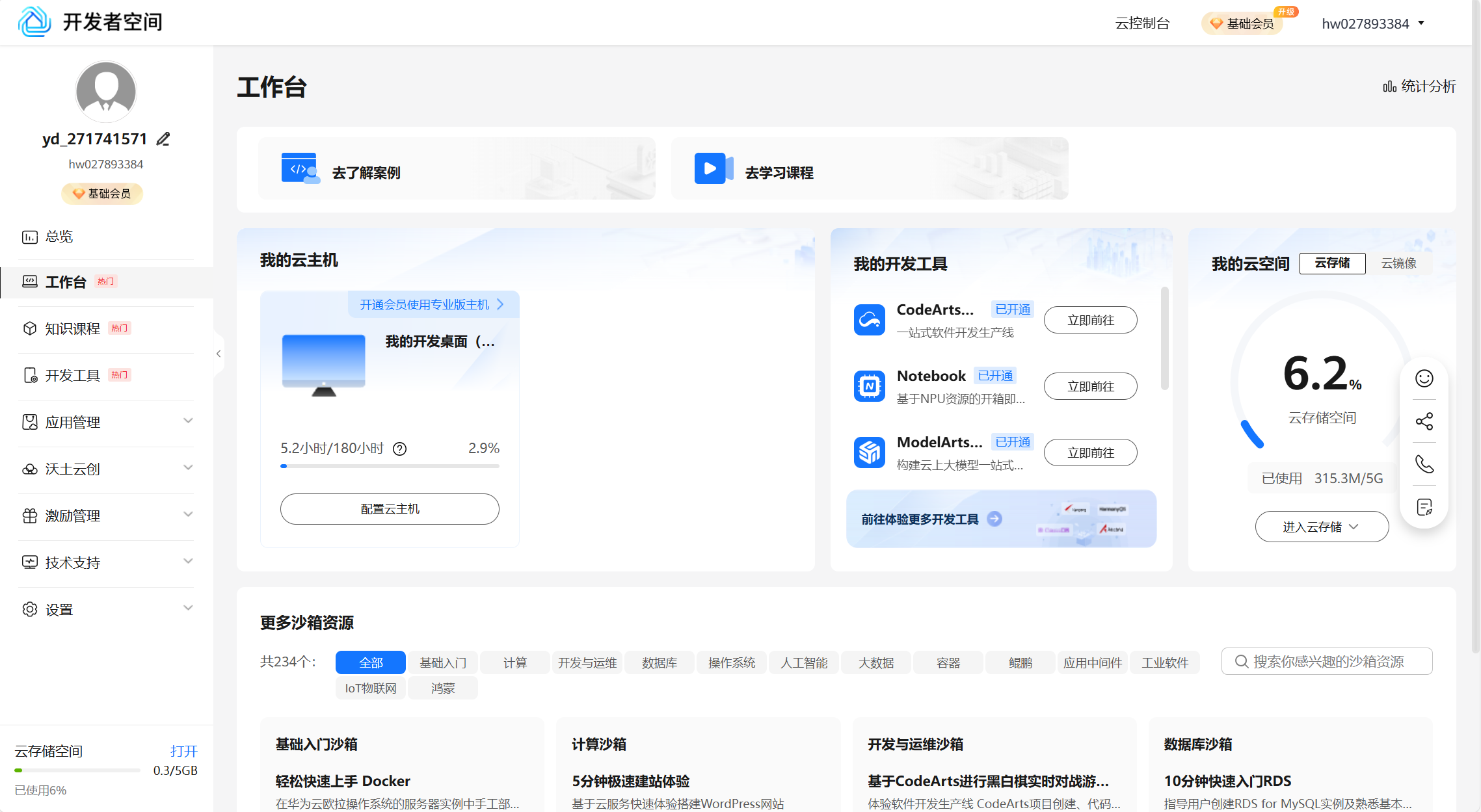
## 启用华为开发者空间

本案例中，使用华为开发者空间作为云上开发平台，完成代码托管、代码检查、编译构建和部署的功能。

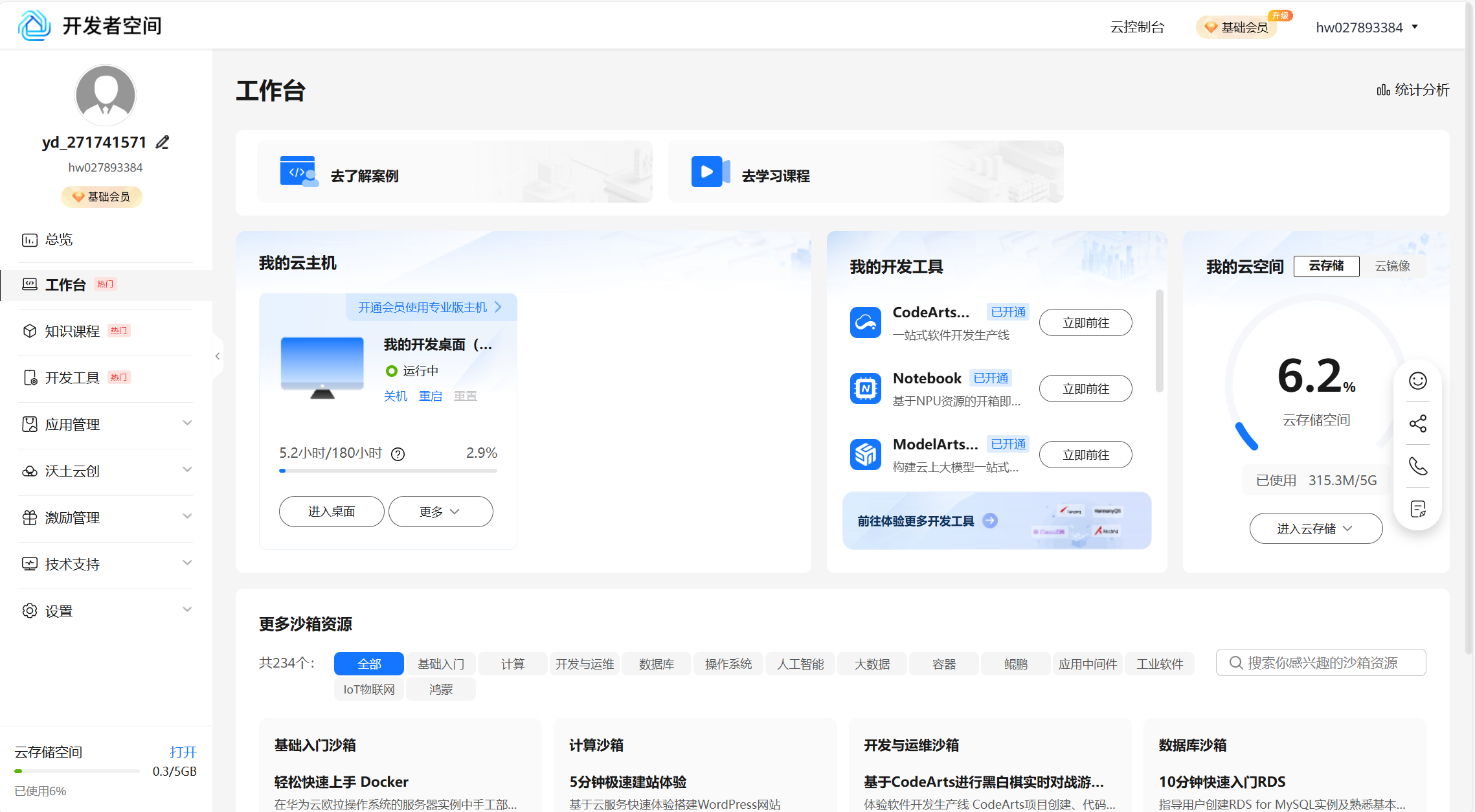
1. 在浏览器中输入华为开发者空间网址：<https://www.huaweicloud.com/>进入华为云首页。



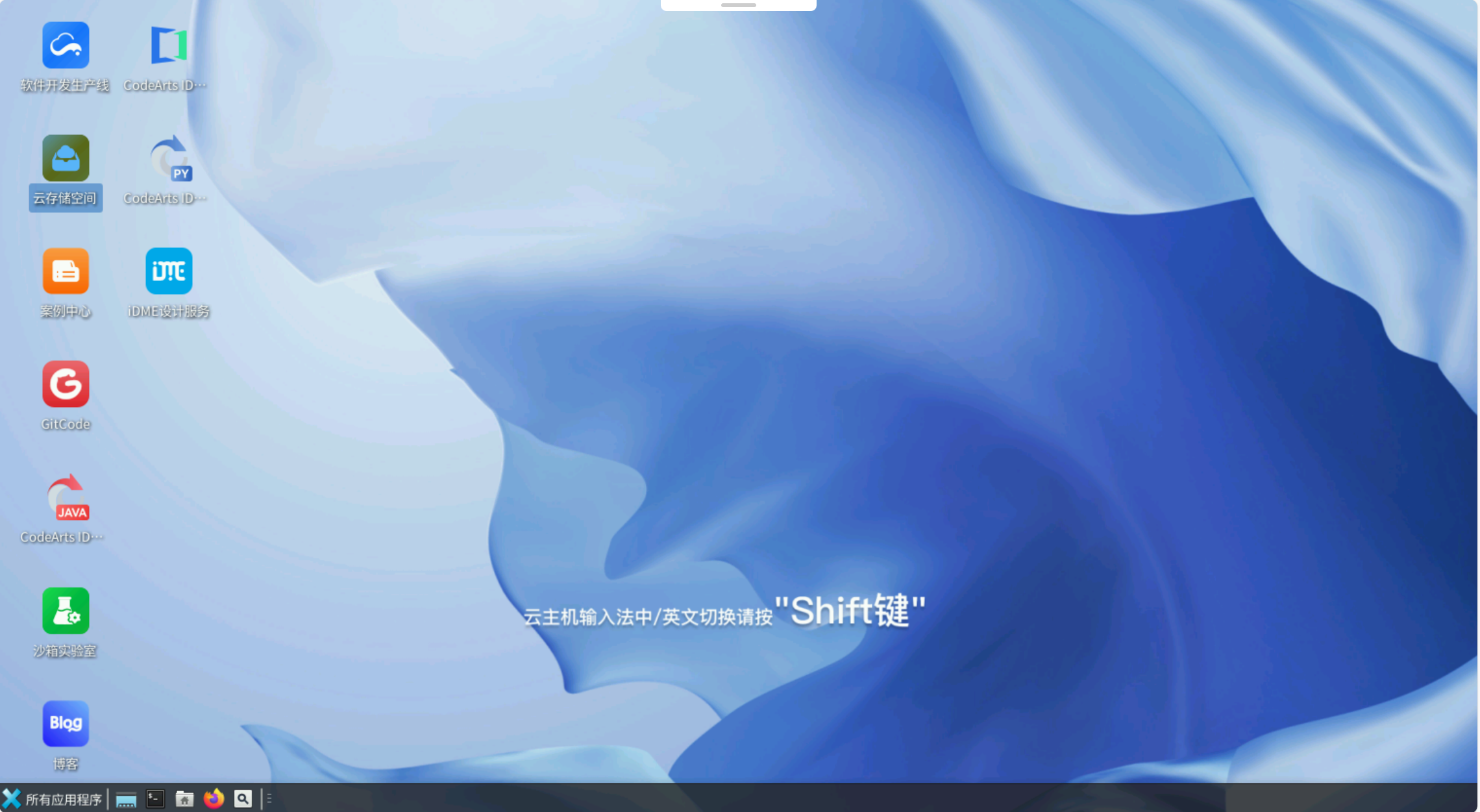
1. 在华为云的首页右上方，点击“登录”，在弹出的对话框中输入自己的用户名和密码。
2. 在华为云首页，选择用户>开发者空间，进入开发者空间页面



1. 在工作台，点击配置云主机，选择如下配置并安装



1. 点击进入桌面

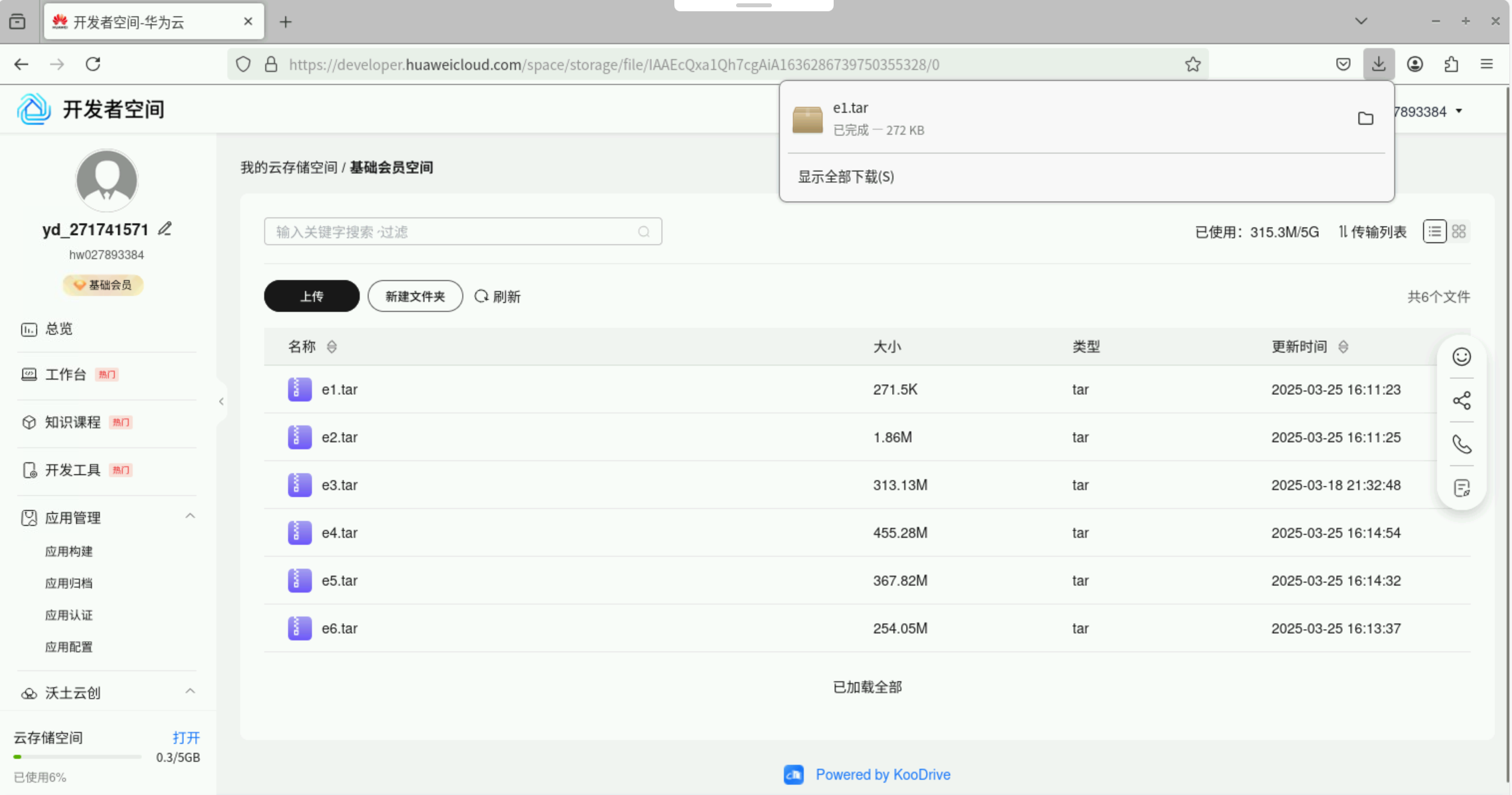


## 下载数据

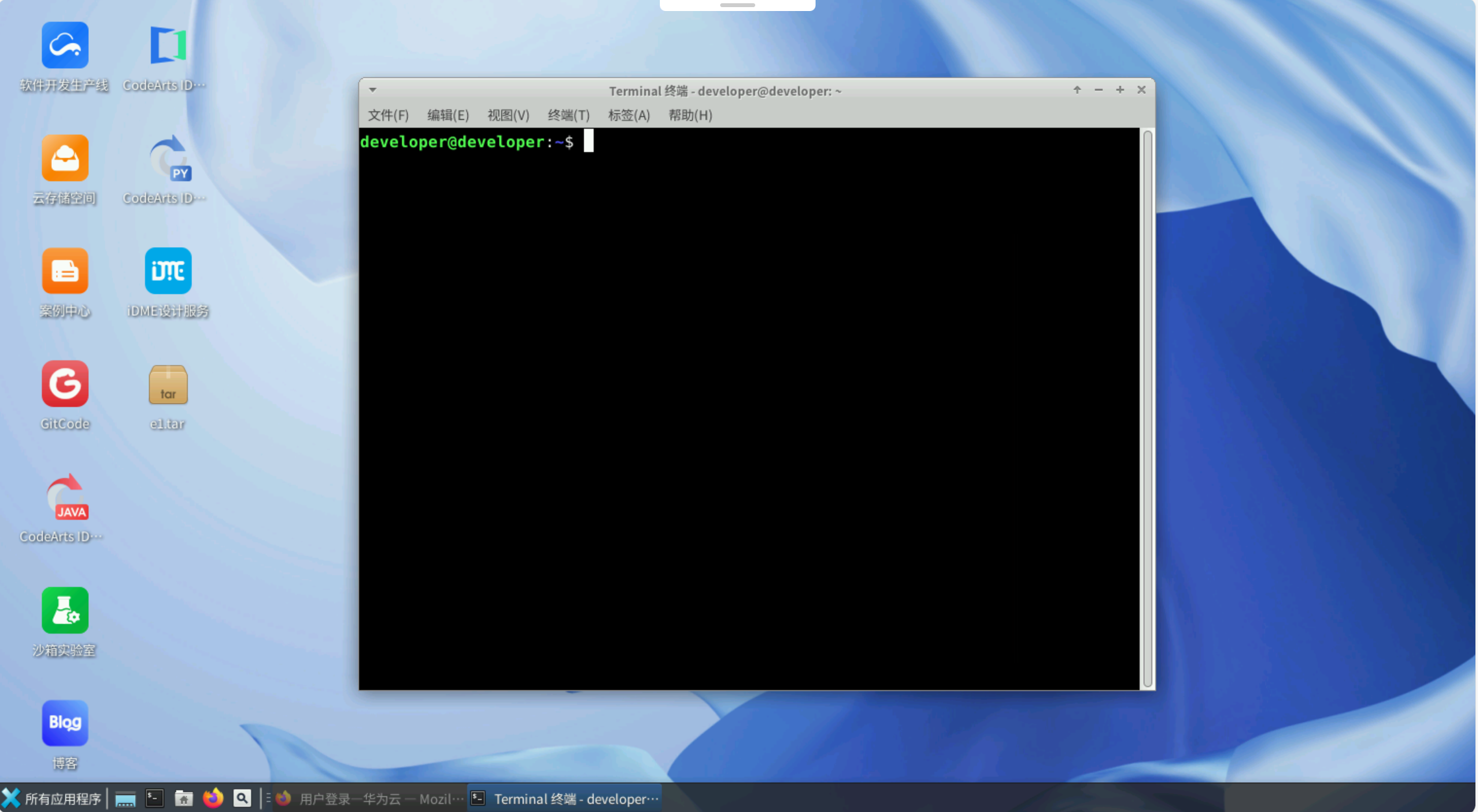
1. 从百度网盘下载实验用数据

https://pan.baidu.com/s/1wi4S19NJSqhSoQkYByaSDQ?pwd=xitm 提取码: xitm

1. 在本机将数据上传至华为云存储空间
2. 在弹出的云主机桌面双击云存储空间，登录并进入华为云存储



1. 下载实验包，点击右上角下载列表，点击实验包右侧文件夹打开，并将实验包拖到桌面
2. 点击云主机桌面左下角终端按钮打开终端，输入以下命令



* cd /home/developer/Desktop/
* tar -xf el.tar

转到桌面并解压缩源码

## 编写代码

打开CodeArts IDE for Python，点击文件>打开项目，新建.py文件

电子设备的屏幕

AI 生成的内容可能不正确。

溶解氧预测实验使用多层感知机模型，利用海洋环境变量预测溶解氧浓度。其包含以下步骤：

1. 引入必要库

from pandas import concat

from numpy import concatenate

from pandas import read\_csv

import numpy as np

from sklearn.preprocessing import MinMaxScaler

import matplotlib.pyplot as pyplot

from keras.models import Sequential

from keras.layers import Dense

from keras.layers import LSTM

from keras.layers import Dense

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from keras.optimizers import Adam

import matplotlib.pyplot as plt

其中Dense为感知机模型层；Adam为优化器，用于反向传播时动态调整参数；pd用于加载和处理一般数据；sklearn库中方法用于数据的预处理，matplotlib用于结果的可视化

1. 加载数据

data = pd.read\_excel('数据.xlsx')

y = data['溶解氧'].values

x = data.drop(['溶解氧'], axis=1)

使用pd.read\_excel方法从数据文件中加载数据，并分成样本与标签，其中每个样本都包含温度、ph值、温度等变量，标签为样本对应溶解氧的值

1. 预处理数据

scaler = StandardScaler().fit(x)

x = pd.DataFrame(scaler.transform(x)).values

# 设定训练集和测试集

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2)

由于不同变量的量纲和分布不同，因此需要对原始数据作标准化，先初始化缩放器StanderdScaler，再使用transform方法将原本数据标准化为均值为0方差为1的分布。最后使用train\_test\_split将所有数据按80%/20%比例划分为训练集和测试集。

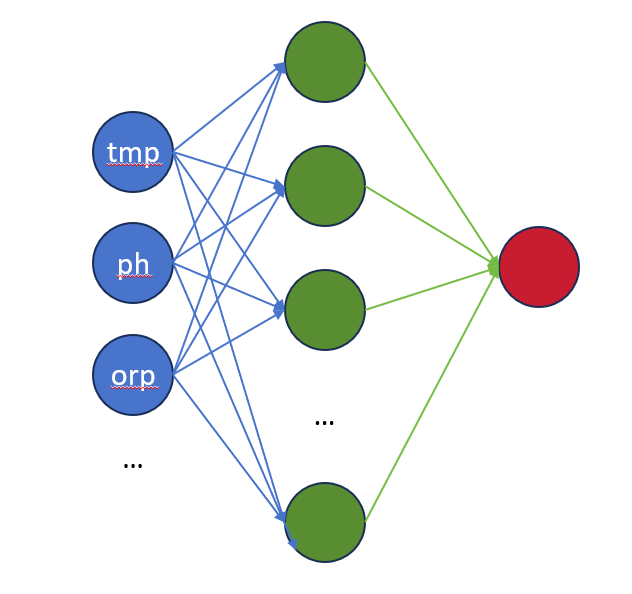
1. 构建预测模型

model = Sequential()

model.add(Dense(units=14, input\_dim=x\_train.shape[1], activation='sigmoid'))

model.add(Dense(units=1, input\_dim=14, activation='sigmoid'))

通过keras和tensorflow库构建多层感知机，第一层感知机有14个神经单元，输入维度与训练数据每个样本的属性数量相同，使用sigmoid作为激活函数；第二层感知机有1个神经单元用于计算输出，输入维度与上一层输出维度相同，使用sigmoid作为激活函数。



蓝色为输入，绿色为14个神经元的中间层，红色为输出

图表, 折线图

AI 生成的内容可能不正确。

sigmoid损失函数

1. 添加损失函数

model.compile(loss='mse', optimizer=Adam(lr=0.01), metrics=['mape'])

使用均方误差（MSE）作为损失函数。损失函数通常用于计算预测结果与标签之间的插值，并通过后续反向传播调整中间参数以便使结果逐渐接近标签。

1. 训练

history = model.fit(x\_train, y\_train, epochs=200, batch\_size=8)

使用model.fit将训练样本与标签送入模型进行训练，训练周期为200，训练批次大小为8，即每次会有8个样本同时进入模型计算。

1. 训练损失可视化

loss = history.history['loss']

epochs = range(len(loss))

plt.plot(epochs, loss, '-b', label='Train\_loss')

plt.legend()

plt.savefig('loss.png')

通过训练过程历史记录获取损失，使用plt可视化图形库绘制训练损失下降过程并保存到本地。

1. 测试

result = model.predict(x\_test, batch\_size=1)

print(y\_test)

print('测试集的预测结果为：', result)

模型在训练集中训练后还需要使用测试集测试，用于验证模型的学习结果能够有足够的泛化性在未知来源或新数据中保持较高的预测准确度。

1. 测试集结果可视化

plt.figure()

plt.plot(y\_test, 'b-o', label='true data')

plt.plot(result, 'r:v',label='predict')

plt.legend()

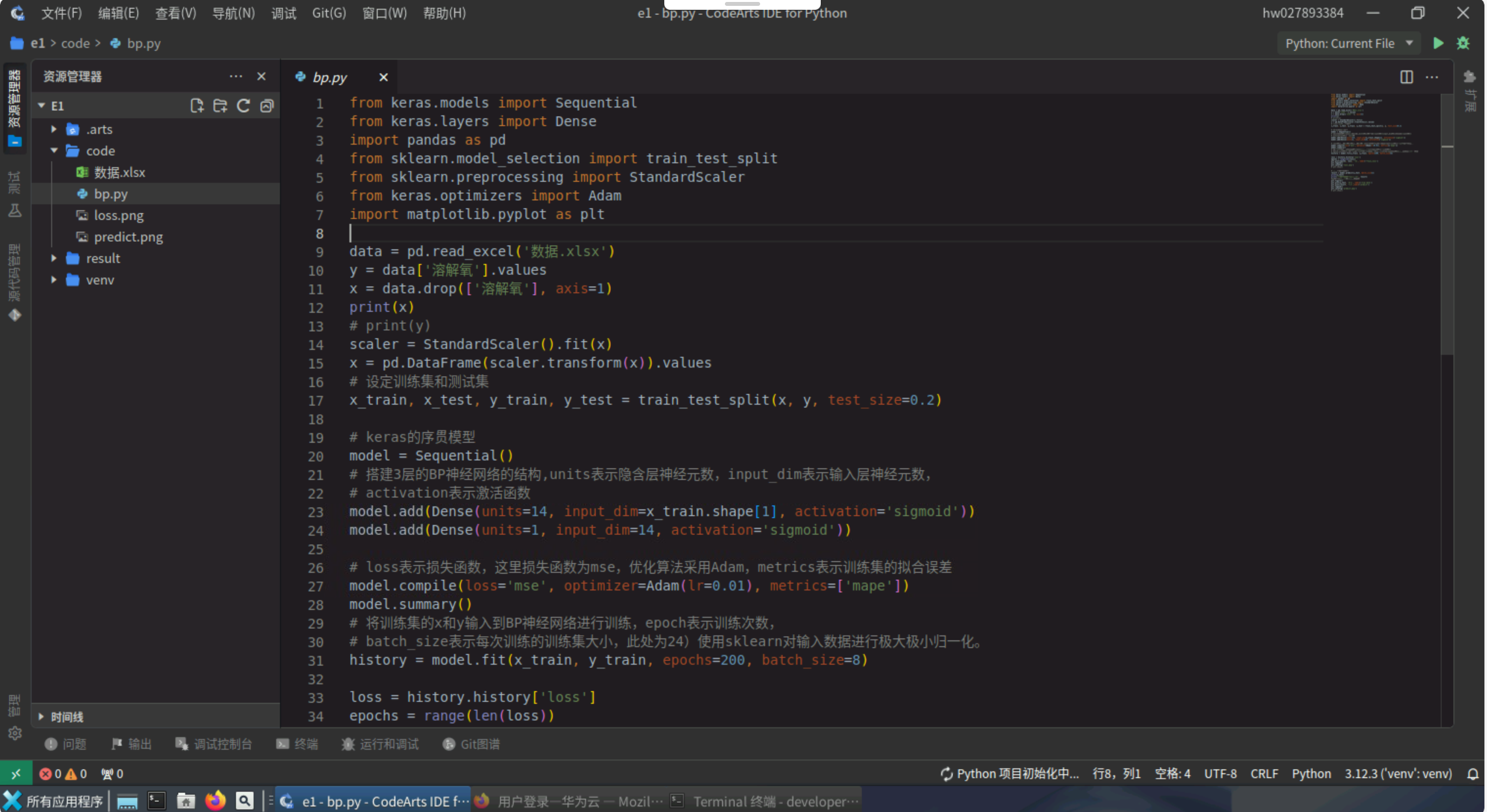
plt.savefig('predict.png')

使用plt可视化图形库绘制测试集预测结果并保存。

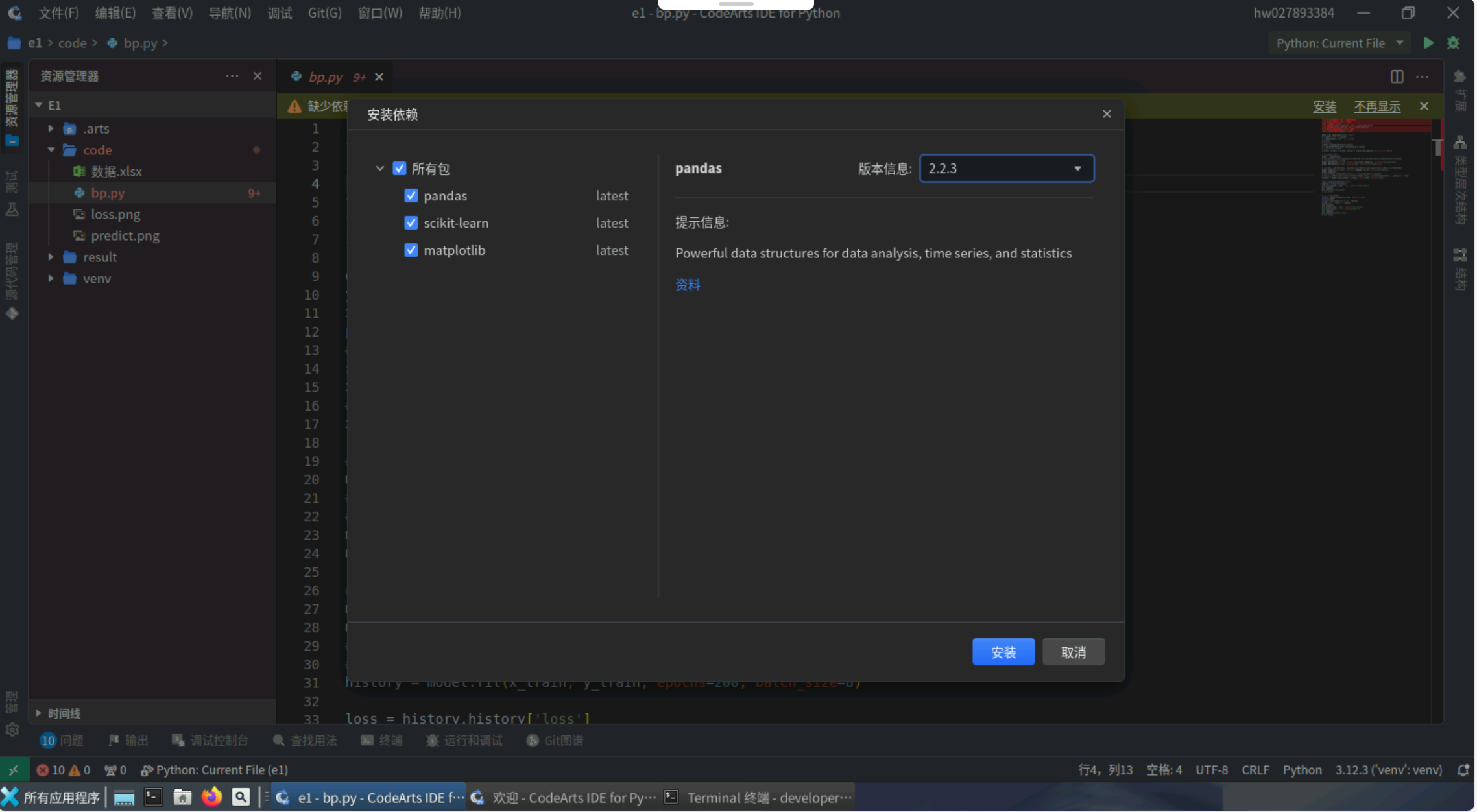
## 构建环境与运行代码

运行代码需要python环境以及相关依赖，根据上述代码，我们需要安装的依赖有tensorflow、openpyxl、keras、scikit-learn、pandas

1. 已编写好的代码



1. 点击底部栏终端，弹出的输入框中输入以下命令安装依赖

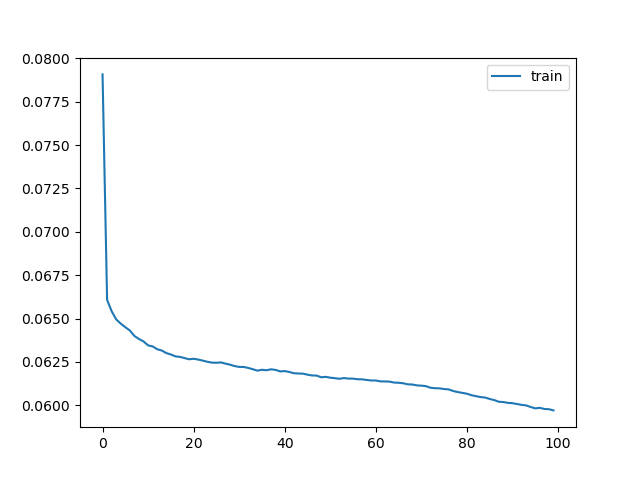


* source F:/PyCharm/Ework/e2/venv/bin/activate
* pip install tensorflow openpyxl keras scikit-learn pandas

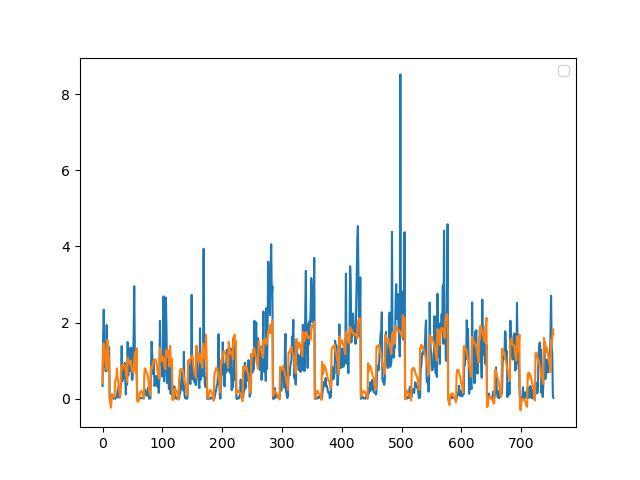
1. 点击右上角运行，执行代码

## 获得结果

在F:/PyCharm/Ework/目录下获得预测结果（示例）



loss.png



predict.png

至此本次实验全部内容完成。

释放资源

## 关闭云服务器

在个人中心处关闭云服务器