上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

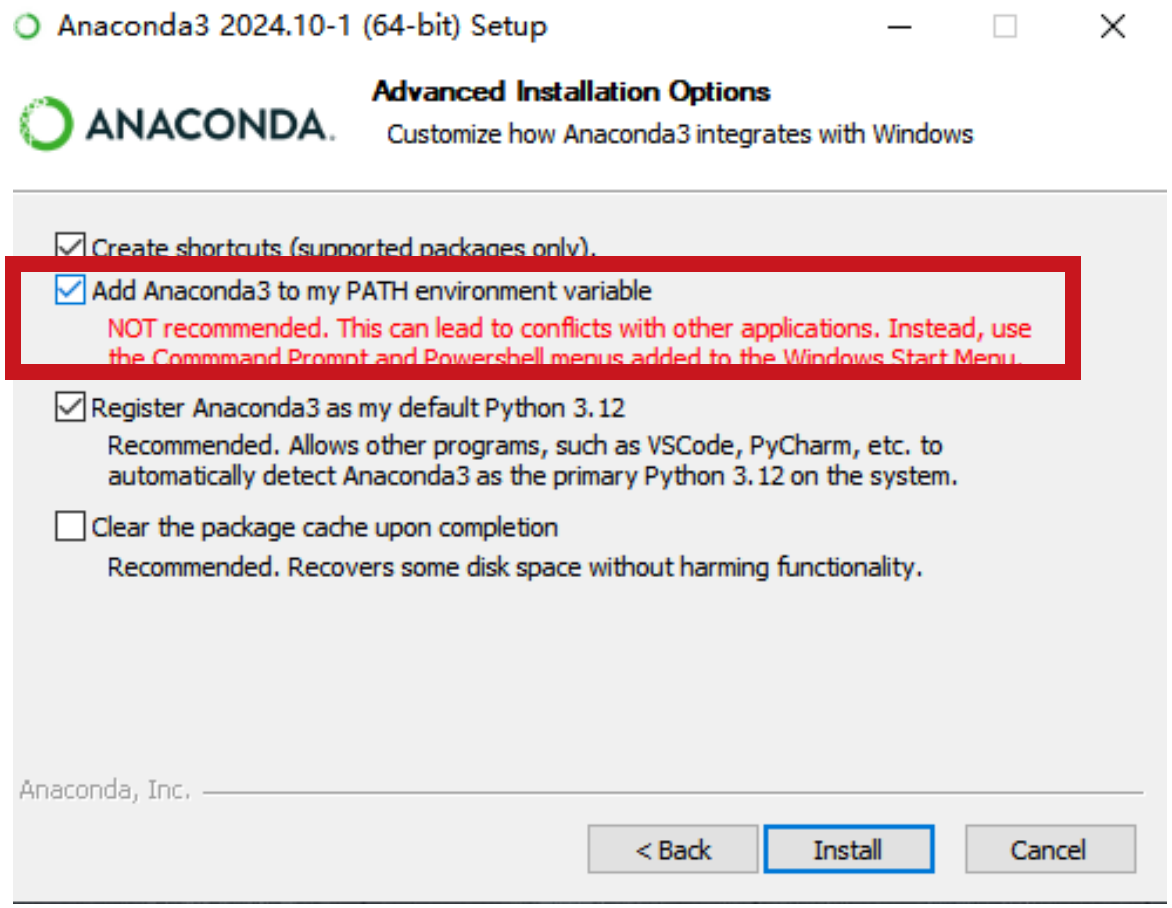
SJTU

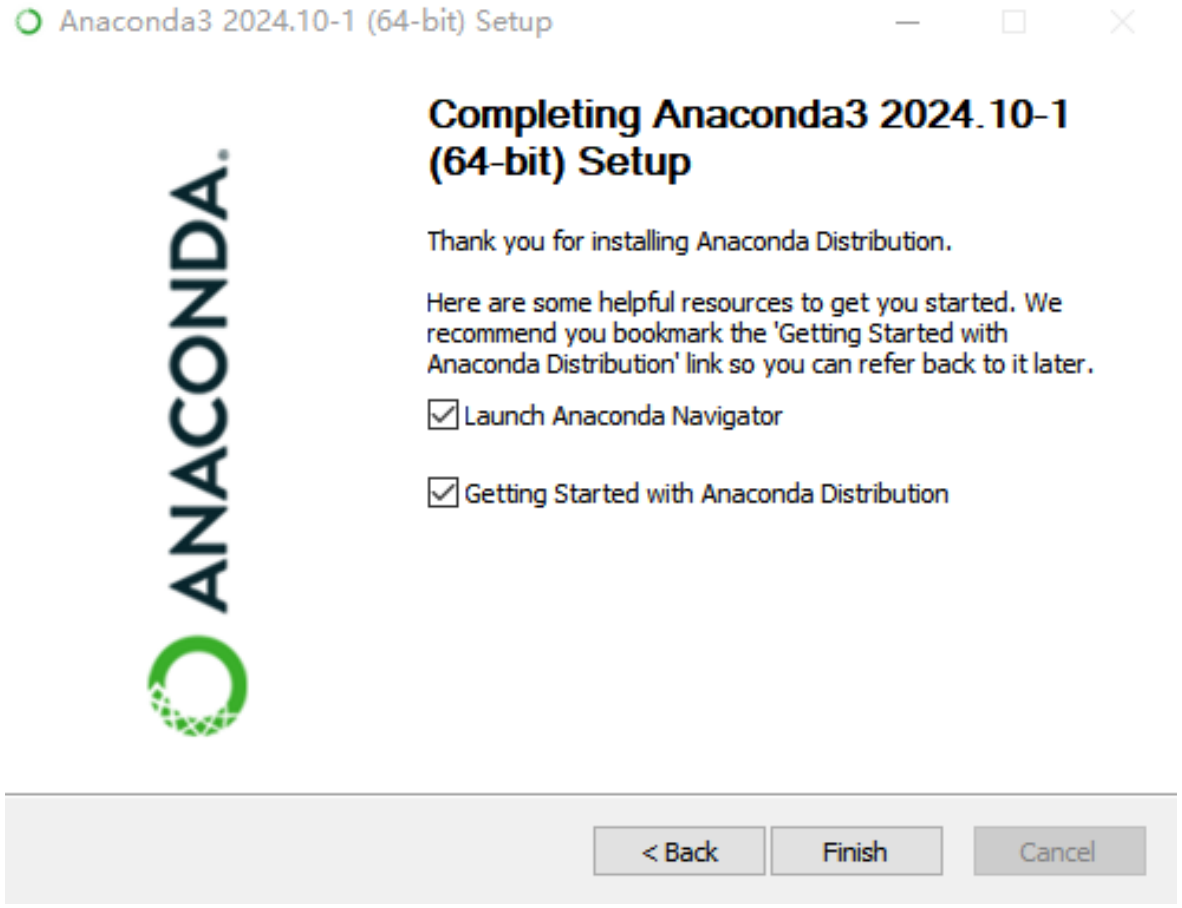
线性回归实验

刘满华

2025年3月29日

Anaconda3安装与环境配置





Anaconda3安装与环境配置

打开Anaconda PowerShell Prompt

conda init

conda create -n AP1226 python=3.8.13

输入y安装基础包

conda env list

conda activate AP1226

pip install -r 路径\\requirements.txt -i http
s://pypi.tuna.tsinghua.edu.cn/simple

pip install torch==1.13.1+cpu torchvision==0.
14.1+cpu torchaudio==0.13.1 --extra-index-u
rl https://download.pytorch.org/whl/cpu

输入python验证库是否安装正确

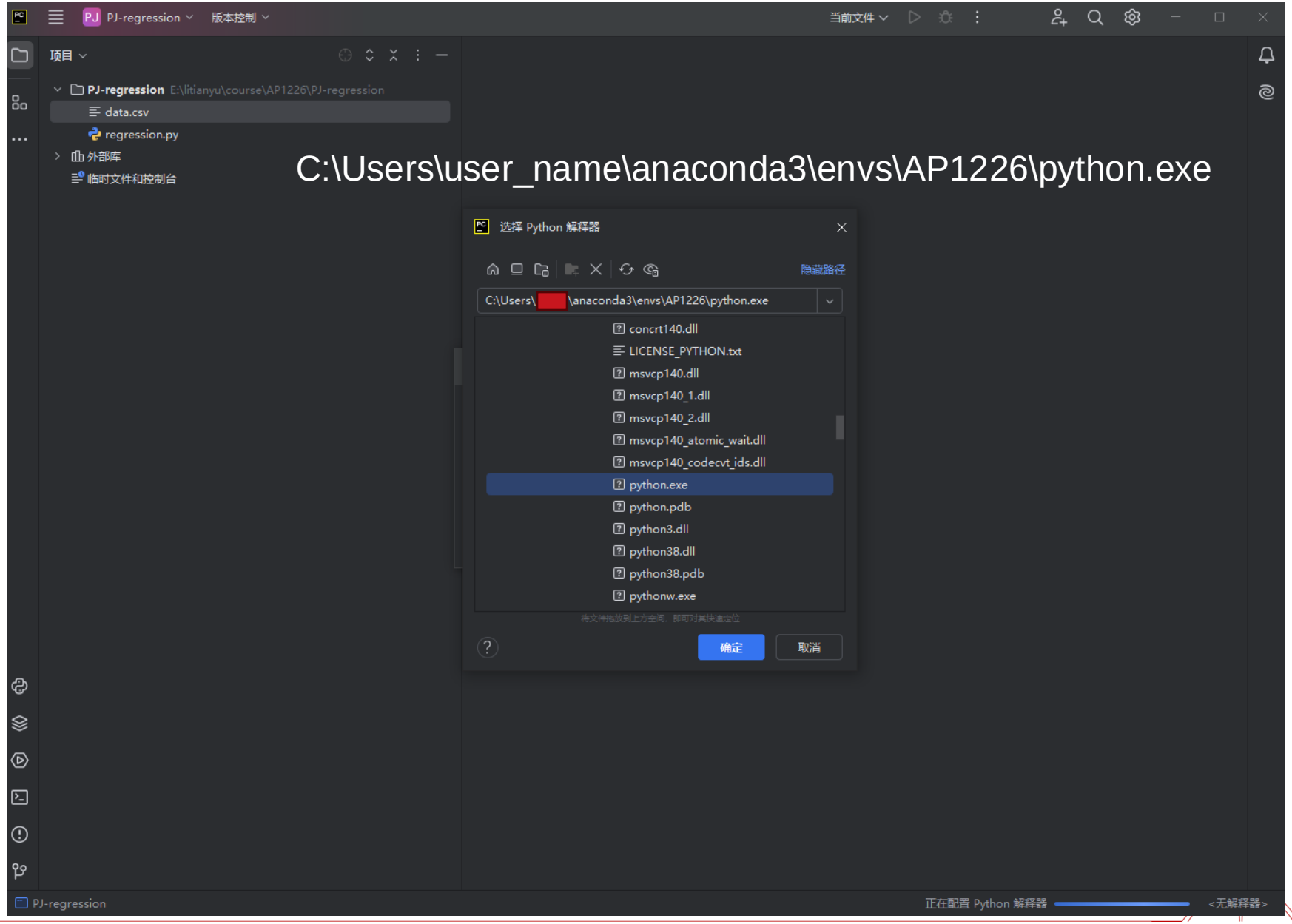
import torch

torch.__version__

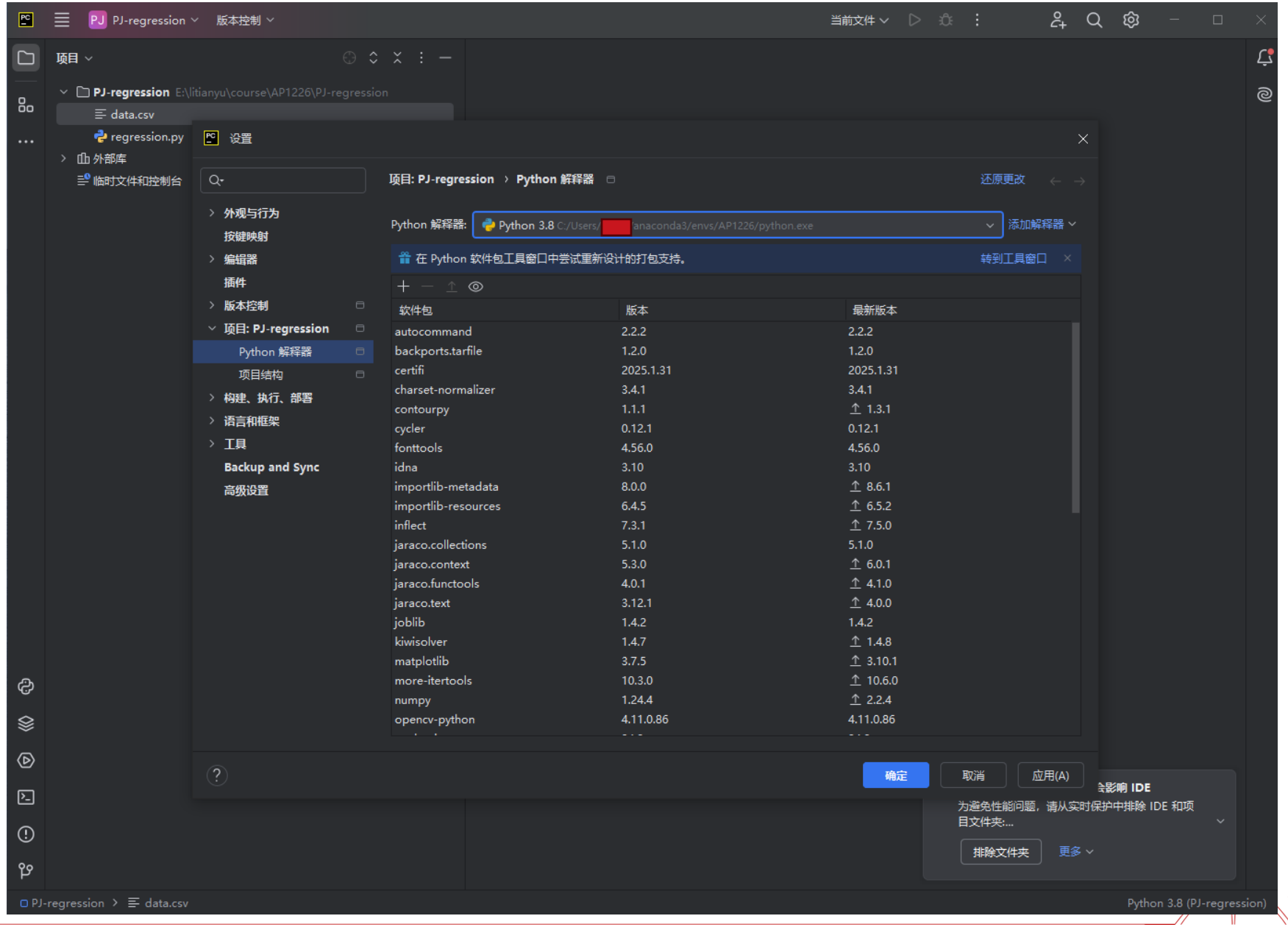
import sklearn

```
PS C:\WINDOWS\system32> conda -V
conda 24.11.3
PS C:\WINDOWS\system32> conda create -n AP1226 python=3.8.13
Channels:
 - defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\[redacted]\anaconda3\envs\AP1226

  added / updated specs:
    - python=3.8.13


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    openssl-1.1.1w             |       h2bbff1b_0         5.5 MB
    pip-24.2                   |  py38haa95532_0         2.4 MB
    python-3.8.13              |       h6244533_1        16.5 MB
    setuptools-75.1.0          |  py38haa95532_0         1.6 MB
    vc-14.42                   |       haa95532_4          11 KB
    vs2015_runtime-14.42.34433 |       he0abc0d_4         1.2 MB
    wheel-0.44.0               |  py38haa95532_0         137 KB
    ---------------------------|-----------------
                                           Total:        27.3 MB

The following NEW packages will be INSTALLED:

  ca-certificates    pkgs/main/win-64::ca-certificates-2025.2.25-haa95532_0
  openssl            pkgs/main/win-64::openssl-1.1.1w-h2bbff1b_0
  pip                pkgs/main/win-64::pip-24.2-py38haa95532_0
  python             pkgs/main/win-64::python-3.8.13-h6244533_1
  setuptools         pkgs/main/win-64::setuptools-75.1.0-py38haa95532_0
  sqlite             pkgs/main/win-64::sqlite-3.45.3-h2bbff1b_0
  vc                 pkgs/main/win-64::vc-14.42-haa95532_4
  vs2015_runtime     pkgs/main/win-64::vs2015_runtime-14.42.34433-he0abc0d_4
  wheel              pkgs/main/win-64::wheel-0.44.0-py38haa95532_0
```

Anaconda3安装与环境配置



Anaconda3安装与环境配置



1. 总体介绍

- 提供的数据包括：
  - csv数据：包含了Chins,Situps,Jumps,Weight,Class五个属性
  - py文件：基础的python代码
    - 可以使用ide（vscode/pycharm等）打开
- 要求完善基础的python代码，并完成线性回归实验

2. 实验介绍 — data visualization 代码解读

- 导入需要的库

```
# data visualization
import pandas as pd                          # 导入 pandas 库，用于读取和处理数据
import matplotlib.pyplot as plt     # 导入 matplotlib 库，用于数据可视化
```

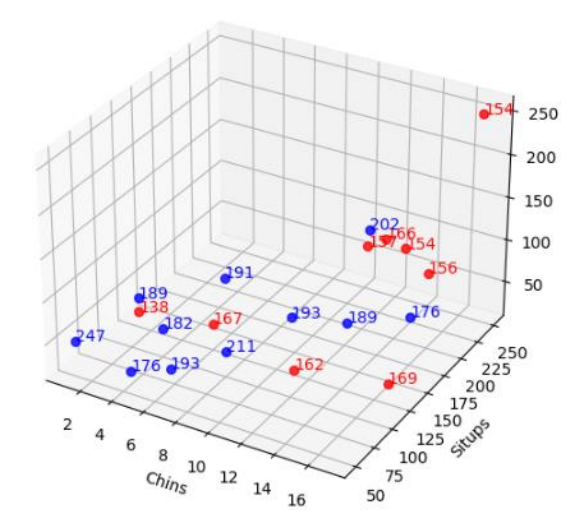
- 加载数据

```
# 加载数据
data = pd.read_csv('/kaggle/input/datacsv/data.csv')    # 使用 pandas 读取 CSV 格式的数据

feature_names = list(data.columns)[0:3]     # 提取数据中的属性特征
target_name = list(data.columns)[3:5]         # 提取数据中的目标特征
print("属性名称:", feature_names)      # 输出属性特征的名称
print("目标名称:", target_name)          # 输出目标特征的名称

X = data.iloc[:, 0:3]    # 提取属性特征列，所有行，0至2列
y = data.iloc[:, 3:5]      # 提取目标特征列，所有行，第3、4列
print("属性形状:", X.shape)  # 输出特征的矩阵维度
print("目标形状:", y.shape)    # 输出目标的矩阵维度
```

2. 实验介绍 — data visualization 代码解读

- 数据可视化

```
属性名称: ['Chins', 'Situps', 'Jumps']
目标名称: ['Weight', 'Class']
属性形状: (20, 3)
目标形状: (20, 2)
```




```python
# 可视化
fig = plt.figure(figsize=(8, 6))      # 创建一个大小为8*6英寸的图形窗口
ax = fig.add_subplot(111, projection='3d')      # 创建一个3D图形子图
ax.set_xlabel(feature_names[0], fontsize=10)    # 设置x轴标签
ax.set_ylabel(feature_names[1], fontsize=10)    # 设置y轴标签
ax.set_zlabel(feature_names[2], fontsize=10)    # 设置z轴标签
ax.set_title('Feature distribution')            # 设置图形标题
xs = X.iloc[:, 0]      # 提取属性特征的第0列（x）作为横坐标
ys = X.iloc[:, 1]      # 提取属性特征的第1列（y）作为纵坐标
zs = X.iloc[:, 2]      # 提取属性特征的第2列（z）作为高度坐标
nums = y.iloc[:, 0]    # 提取目标特征的第0列作为标签
classes = y.iloc[:, 1] # 提取目标特征的第1列作为类别
colors = ['red', 'blue']  # 颜色列表
for i in range(len(xs)):            # 为每个数据点添加标签
    ax.scatter(xs[i], ys[i], zs[i], c=colors[classes[i]-1], marker='o', s=30, alpha=0.8)  # 绘制散点图
    # xs[i]、ys[i]、zs[i] 表示数据点在三维空间中的坐标，c 参数指定点的颜色，marker 参数指定点的形状，s 参数指定点的大小，alpha 参数指定点的透明度。
    ax.text(xs[i], ys[i], zs[i], nums[i], color=colors[classes[i]-1])  # 添加数据点标签
    # nums[i] 表示数据点的标签，color 参数指定标签的颜色。colors 是对应颜色列表，classes[i]-1 的结果则是数据点所属类别对应的颜色在列表中的索引。
plt.show()          # 显示可视化结果
```

3. 线性回归实验 — 一元线性回归

- 导入需要的库

```
import pandas as pd     # 导入 pandas 库，用于读取和处理数据
from sklearn.linear_model import LinearRegression      # 导入线性回归模型
from sklearn.metrics import mean_squared_error      # 导入均方误差指标
import matplotlib.pyplot as plt     # 导入 matplotlib 库，用于数据可视化
```

- 加载数据

```
# 加载数据
df = pd.read_csv('/kaggle/input/datacsv/data.csv')     # 使用 pandas 读取 CSV 格式的数据
X = df.iloc[:, 0]     # 提取数据中的第0列作为属性特征，数据中的第0、1、2三列分别是Chins,Situps,Jumps三个特征的数据
y = df.iloc[:, 3]     # 提取数据中的第3列作为目标特征

# 定义测试样本的索引
test_indices = [1, 17]
# 拆分训练集和测试集
X_train = X.drop(test_indices, axis=0)
y_train = y.drop(test_indices, axis=0)
X_test = X.iloc[test_indices]
y_test = y.iloc[test_indices]
X_train = X_train.values.reshape(-1, 1)     # 将训练集属性特征的一维数组转换为二维数组
y_train = y_train.values.reshape(-1, 1)     # 将训练集目标特征的一维数组转换为二维数组
X_test = X_test.values.reshape(-1, 1)     # 将测试集属性特征的一维数组转换为二维数组
y_test = y_test.values.reshape(-1, 1)     # 将测试集目标特征的一维数组转换为二维数组
```

3. 线性回归实验 — 一元线性回归

- 构建线性回归模型并拟合

```
# 构建并拟合线性回归模型
model = LinearRegression()      # 创建线性回归模型对象
model.fit(X_train, y_train)     # 使用数据拟合线性回归模型
```

- 使用模型对数据进行预测

```
# 使用训练好的模型预测
y_pred = model.predict(X_test)      # 对特征进行预测
mse = mean_squared_error(y_test, y_pred)      # 计算均方误差（MSE）
```

3. 线性回归实验 — 一元线性回归

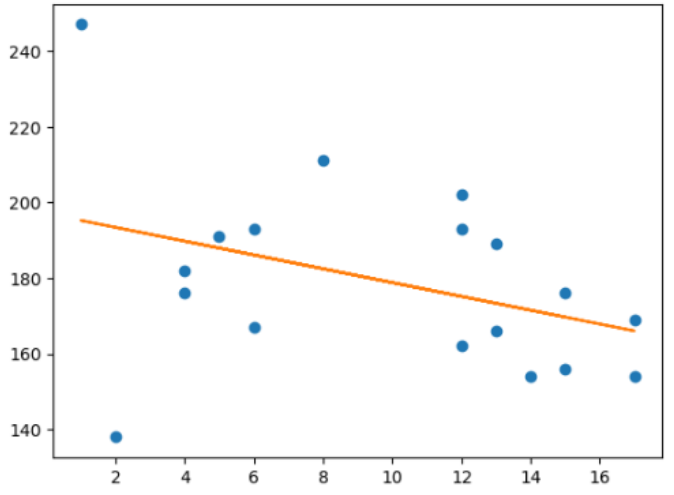
- 输出预测结果

```
# 输出模型权重、偏置、均方误差
print('权重:', model.coef_)
print('偏置:', model.intercept_)
print('均方误差MSE:', mse)
```

```
权重: [[-1.82358491]]
偏置: [197.05283019]
均方误差MSE: 209.57286801352836
```

- 可视化预测结果

```
# 绘制数据可视化图形
y_train_pred = model.predict(X_train)     # 对特征进行预测
fig, ax = plt.subplots()  # 创建一个子图
ax.plot(X_train, y_train, 'o')  # 绘制数据散点图
ax.plot(X_train, y_train_pred, '-')  # 绘制拟合直线
plt.show()  # 显示图形
```



3. 线性回归实验 — 一元线性回归

上述代码展示了 Chins特征与Weight特征的一元线性回归结果。

如何进行 Situps特征与Weight特征、 Jumps特征与Weight特征的一元线性回归?

请修改 one-variable regression 部分的代码，并填写下表的前三行。

	数据	权重	截距	均方误差MSE
任务1 回归（Regression）	X:**Chins(俯卧撑)** Y:Weight(体重)	-1.824	197.053	209.573
	X:**Situps(仰卧起坐)** Y:Weight(体重)			
	X:**Jumps(跳绳)** Y:Weight(体重)			
	X:[Chins, Situps, Jumps] Y:Weight			

3. 线性回归实验 — 多元线性回归

- 导入需要的库

```python
import pandas as pd    # 导入 pandas 库，用于读取和处理数据
from sklearn.linear_model import LinearRegression  # 使用以线性回归模型为基础的库
from sklearn.metrics import mean_squared_error  # 导入用于计算MSE的库
```

- 加载数据

```python
# 读取数据
df = pd.read_csv('/kaggle/input/datacsv/data.csv')
X = df.iloc[:, :3]  # 提取属性特征列 Chins,Situps,Jumps
y = df.iloc[:, 3]   # 提取目标特征列 Weights

# 定义测试样本的索引
test_indices = [1, 17]
# 拆分训练集和测试集
X_train = X.drop(test_indices, axis=0)
y_train = y.drop(test_indices, axis=0)
X_test = X.iloc[test_indices, :]
y_test = y.iloc[test_indices]
```

3. 线性回归实验 — 多元线性回归

- 构建线性回归模型并拟合

```
# 定义线性回归模型并用训练数据进行拟合
lin_reg = LinearRegression()
lin_reg.fit(    ,    )
```

- 使用模型对数据进行预测

```
# 输入测试数据并输出预测结果和评估指标
result(lin_reg,    ,    )
```

调用输出结果的函数

完善并运行代码，记录测试结果

```
def result(model, X_test, y_test, PrintSample=True):
    # 使用模型预测自变量的值
    y_pred = model.predict(X_test)
    if PrintSample:
        # 打印每个样本的预测值和真实值（可选）
        for i in range(len(y_pred)):
            print("预测值: {:.3f}, 真实值e: {}".format(y_pred[i], y_test.iloc[i]))
    # 计算误差指标：MSE
    mse = mean_squared_error(y_test, y_pred)
    # 打印模型参数和误差指标
    print('权重:', model.coef_)
    print('偏置:', model.intercept_)
    print('均方误差MSE:', mse)
```

请修改 multiple regression 部分的代码，并将结果填写在下表。

X:[Chins, Situps, Jumps] Y:Weight				

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

人工智能研究院
Artificial Intelligence Institute

谢 谢！

饮 水 思 源　　爱 国 荣 校