1、读取data\_multivar.txt数据

（1）划分训练集和测试集

（2）使用线性模型、岭回归和随机梯度下降模型训练数据，分别输出训练集在三种模型下的mse 和mae，三种模型对测试集的测试结果mse，mae,r2

   (3) 选择三种模型中的一种使用十折交叉验证输出交叉验证后的mse，mae,r2

import numpy as np  
import pandas as pd  
from sklearn.model\_selection import train\_test\_split, cross\_val\_score  
from sklearn.linear\_model import LinearRegression, Ridge  
from sklearn.linear\_model import SGDRegressor  
from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score  
  
# 读取数据  
data = np.loadtxt('../3/data\_multivar.txt',delimiter=',')  
# print(data)  
X=data[:,:-1]  
y=data[:,-1]  
# print(X)  
# print(y)  
  
  
# 划分训练集和测试集  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
# 定义模型  
models = {  
 "Linear Regression": LinearRegression(),  
 "Ridge Regression": Ridge(),  
 "SGD Regression": SGDRegressor()  
}  
  
# 训练模型并输出结果  
results = {}  
for name, model in models.items():  
 model.fit(X\_train, y\_train)  
 train\_pred = model.predict(X\_train)  
 test\_pred = model.predict(X\_test)  
 train\_mse = mean\_squared\_error(y\_train, train\_pred)  
 train\_mae = mean\_absolute\_error(y\_train, train\_pred)  
 test\_mse = mean\_squared\_error(y\_test, test\_pred)  
 test\_mae = mean\_absolute\_error(y\_test, test\_pred)  
 test\_r2 = r2\_score(y\_test, test\_pred)  
 results[name] = {  
 "Train MSE": train\_mse,  
 "Train MAE": train\_mae,  
 "Test MSE": test\_mse,  
 "Test MAE": test\_mae,  
 "Test R2": test\_r2  
 }  
  
# 输出结果  
for name, metrics in results.items():  
 print(name)  
 for metric, value in metrics.items():  
 print(f"{metric}: {value}")  
 print()  
  
# 选择一种模型进行十折交叉验证  
selected\_model = SGDRegressor()  
cv\_scores\_mse = cross\_val\_score(selected\_model, X, y, cv=10, scoring='neg\_mean\_squared\_error')  
cv\_scores\_mae = cross\_val\_score(selected\_model, X, y, cv=10, scoring='neg\_mean\_absolute\_error')  
cv\_scores\_r2 = cross\_val\_score(selected\_model, X, y, cv=10, scoring='r2')  
  
print("Cross-validation results:")  
print(f"CV MSE: {-cv\_scores\_mse.mean()}")  
print(f"CV MAE: {-cv\_scores\_mae.mean()}")  
print(f"CV R2: {cv\_scores\_r2.mean()}")



2、获取加州(california)住房数据集

from sklearn.datasets import fetch\_california\_housing

housing = fetch\_california\_housing()

艾姆斯住房数据集

# from sklearn.datasets import fetch\_openml

# housing = fetch\_openml(name="house\_prices", as\_frame=True)

（1）划分训练集和测试集

（2）岭回归（测试不同的alpha参数）和随机梯度下降模型（测试不同的参数）训练数据，分别输出训练集在两种模型下的mse 和mae，两种模型对测试集的测试结果mse，mae,r2

   (3) 选择一种调试最优的模型使用十折交叉验证输出交叉验证后的mse，mae,r2

(4)保存模型

from sklearn.datasets import fetch\_california\_housing  
from sklearn.model\_selection import train\_test\_split, cross\_val\_score  
from sklearn.linear\_model import LinearRegression, Ridge  
from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score  
import joblib  
import numpy as np  
  
# 获取加州住房数据集  
housing = fetch\_california\_housing()  
  
# 划分训练集和测试集  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(housing.data, housing.target, test\_size=0.2, random\_state=42)  
  
# 线性模型  
linear\_model = LinearRegression()  
linear\_model.fit(X\_train, y\_train)  
train\_pred\_linear = linear\_model.predict(X\_train)  
test\_pred\_linear = linear\_model.predict(X\_test)  
train\_mse\_linear = mean\_squared\_error(y\_train, train\_pred\_linear)  
train\_mae\_linear = mean\_absolute\_error(y\_train, train\_pred\_linear)  
test\_mse\_linear = mean\_squared\_error(y\_test, test\_pred\_linear)  
test\_mae\_linear = mean\_absolute\_error(y\_test, test\_pred\_linear)  
test\_r2\_linear = r2\_score(y\_test, test\_pred\_linear)  
  
print("Linear Model Results:")  
print(f"Train MSE: {train\_mse\_linear}")  
print(f"Train MAE: {train\_mae\_linear}")  
print(f"Test MSE: {test\_mse\_linear}")  
print(f"Test MAE: {test\_mae\_linear}")  
print(f"Test R2: {test\_r2\_linear}")  
  
# 岭回归  
alpha\_values = [0.1, 1, 10] # 测试不同的alpha参数  
ridge\_results = {}  
for alpha in alpha\_values:  
 ridge = Ridge(alpha=alpha)  
 ridge.fit(X\_train, y\_train)  
 train\_pred = ridge.predict(X\_train)  
 test\_pred = ridge.predict(X\_test)  
 train\_mse = mean\_squared\_error(y\_train, train\_pred)  
 train\_mae = mean\_absolute\_error(y\_train, train\_pred)  
 test\_mse = mean\_squared\_error(y\_test, test\_pred)  
 test\_mae = mean\_absolute\_error(y\_test, test\_pred)  
 test\_r2 = r2\_score(y\_test, test\_pred)  
 ridge\_results[alpha] = {  
 "Train MSE": train\_mse,  
 "Train MAE": train\_mae,  
 "Test MSE": test\_mse,  
 "Test MAE": test\_mae,  
 "Test R2": test\_r2  
 }  
  
print("\nRidge Regression Results:")  
for alpha, metrics in ridge\_results.items():  
 print(f"Alpha: {alpha}")  
 for metric, value in metrics.items():  
 print(f"{metric}: {value}")  
 print()  
  
# 选择模型  
selected\_model = linear\_model  
  
# 十折交叉验证  
cv\_scores\_mse = cross\_val\_score(selected\_model, housing.data, housing.target, cv=10, scoring='neg\_mean\_squared\_error')  
cv\_scores\_mae = cross\_val\_score(selected\_model, housing.data, housing.target, cv=10, scoring='neg\_mean\_absolute\_error')  
cv\_scores\_r2 = cross\_val\_score(selected\_model, housing.data, housing.target, cv=10, scoring='r2')  
  
print("\nCross-validation results:")  
print(f"CV MSE: {-cv\_scores\_mse.mean()}")  
print(f"CV MAE: {-cv\_scores\_mae.mean()}")  
print(f"CV R2: {cv\_scores\_r2.mean()}")  
print()  
# 保存模型  
joblib.dump(selected\_model, 'california\_housing\_model.pkl')  
  
# 从文件中加载模型  
model = joblib.load('california\_housing\_model.pkl')  
  
sample = X\_train[1]  
  
# 使用加载的模型进行预测  
prediction = model.predict(sample.reshape(1, -1))  
# print(y\_train[1])  
# 打印预测结果  
print("Predicted housing price:", prediction)

