# 线性回归

## 导入实验所需包

import numpy as np  
import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression #线性回归  
from sklearn.linear\_model import LogisticRegression #逻辑回归  
from sklearn.preprocessing import StandardScaler  
from sklearn.metrics import mean\_squared\_error

## 定义模型

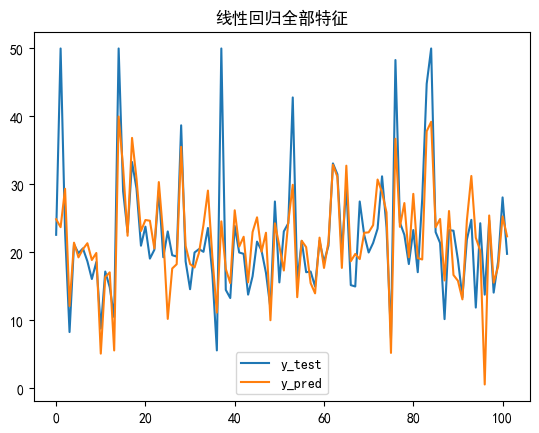
# 定义模型  
def linear\_model(X\_train, X\_test, y\_train):  
 # 线性回归  
 lr = LinearRegression()  
 lr.fit(X\_train, y\_train)  
 y\_pred = lr.predict(X\_test)  
 return y\_pred

## 数据处理

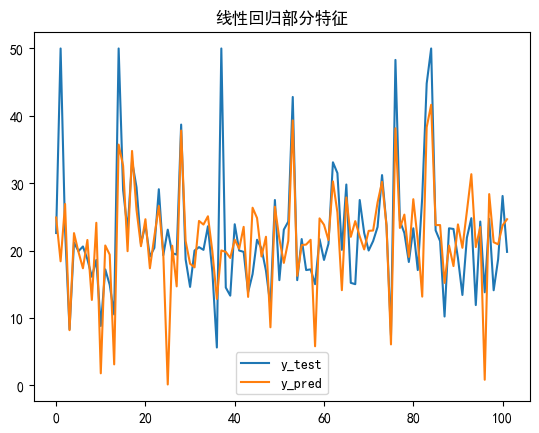
# 全部特征  
def dataProcess\_full():  
 # 加载 dataset/housing-data.csv  
 df = pd.read\_csv('dataset/housing-data.csv')  
 # 数据处理  
 # 平均值填充  
 df = df.fillna(df.mean())  
 X = df.iloc[:, :-1]  
 y = df.iloc[:, -1]  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)  
 # 标准化  
 sc = StandardScaler()  
 X\_train = sc.fit\_transform(X\_train)  
 X\_test = sc.transform(X\_test)  
 return X\_train, X\_test, y\_train, y\_test  
# 部分特征  
def dataProcess\_part():  
 # 加载 dataset/housing-data.csv  
 df = pd.read\_csv('dataset/housing-data.csv')  
 # 数据处理  
 # 平均值填充  
 df = df.fillna(df.mean())  
 X = df.iloc[:, [0, 1, 2, 3, 4, 5]]  
 y = df.iloc[:, -1]  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)  
 # 标准化  
 sc = StandardScaler()  
 X\_train = sc.fit\_transform(X\_train)  
 X\_test = sc.transform(X\_test)  
 return X\_train, X\_test, y\_train, y\_test

## 主函数

# 设置显示中文字体  
from pylab import mpl  
mpl.rcParams["font.sans-serif"] = ["SimHei"]  
  
# 全部特征  
X\_train, X\_test, y\_train, y\_test = dataProcess\_full()  
# 将y\_test转换为一维数组  
y\_test = np.array(y\_test).reshape(-1, 1)  
y\_pred = linear\_model(X\_train, X\_test, y\_train)  
# 可视化  
from matplotlib import pyplot as plt  
plt.title('线性回归全部特征')  
plt.plot(y\_test, label='y\_test')  
plt.plot(y\_pred, label='y\_pred')  
plt.legend()  
plt.show()  
print('全部特征MSE:', mean\_squared\_error(y\_test, y\_pred))  
# 部分特征  
X\_train, X\_test, y\_train, y\_test = dataProcess\_part()  
y\_test = np.array(y\_test).reshape(-1, 1)  
y\_pred = linear\_model(X\_train, X\_test, y\_train)  
# 可视化  
plt.title('线性回归部分特征')  
plt.plot(y\_test, label='y\_test')  
plt.plot(y\_pred, label='y\_pred')  
plt.legend()  
plt.show()  
print('部分特征MSE:', mean\_squared\_error(y\_test, y\_pred))



全部特征MSE: 33.448979997676496



部分特征MSE: 47.24194082350583

# 逻辑回归

## 导入实验所需包

import numpy as np  
import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression #线性回归  
from sklearn.linear\_model import LogisticRegression #逻辑回归  
from sklearn.preprocessing import StandardScaler  
from sklearn.metrics import mean\_squared\_error

## 定义模型

def logistic\_model(X\_train, X\_test, y\_train):  
 # 逻辑回归  
 lr = LogisticRegression()  
 lr.fit(X\_train, y\_train)  
 y\_pred = lr.predict(X\_test)  
 return y\_pred

## 数据处理

# 数据处理  
# 全部特征  
def dataProcess\_full():  
 path = 'dataset/breast-cancer-wisconsin.data'  
 df = pd.read\_csv(path, header=None)  
 # 0填充'?'  
 df = df.replace('?', 0)  
 # 将所有数据转换为int类型  
 df = df.astype(int)  
 # 特征值  
 X = df.iloc[:, 1:-1]  
 # 目标值  
 y = df.iloc[:, -1]  
 # 划分数据集  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)  
 # 标准化  
 sc = StandardScaler()  
 X\_train = sc.fit\_transform(X\_train)  
 X\_test = sc.transform(X\_test)  
 return X\_train, X\_test, y\_train, y\_test  
  
# 部分特征  
def dataProcess\_part():  
 path = 'dataset/breast-cancer-wisconsin.data'  
 df = pd.read\_csv(path, header=None)  
 # 0填充'?'  
 df = df.replace('?', 0)  
 # 将所有数据转换为int类型  
 df = df.astype(int)  
 # 特征值  
 X = df.iloc[:, [1, 2, 3, 4, 5]]  
 # 目标值  
 y = df.iloc[:, -1]  
 # 划分数据集  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)  
 # 标准化  
 sc = StandardScaler()  
 X\_train = sc.fit\_transform(X\_train)  
 X\_test = sc.transform(X\_test)  
 return X\_train, X\_test, y\_train, y\_test

## 主函数

# 设置显示中文字体  
from pylab import mpl  
mpl.rcParams["font.sans-serif"] = ["SimHei"]  
from matplotlib import pyplot as plt  
  
# 全部特征  
X\_train, X\_test, y\_train, y\_test = dataProcess\_full()  
y\_test = np.array(y\_test).reshape(-1, 1)  
y\_pred = logistic\_model(X\_train, X\_test, y\_train)  
# 混淆矩阵  
from seaborn import heatmap  
from sklearn.metrics import confusion\_matrix  
cm = confusion\_matrix(y\_test, y\_pred)  
heatmap(cm, annot=True, fmt='d')  
plt.title('逻辑回归全部特征')  
plt.show()  
  
# 部分特征  
X\_train, X\_test, y\_train, y\_test = dataProcess\_part()  
y\_test = np.array(y\_test).reshape(-1, 1)  
y\_pred = logistic\_model(X\_train, X\_test, y\_train)  
# 混淆矩阵  
cm = confusion\_matrix(y\_test, y\_pred)  
heatmap(cm, annot=True, fmt='d')  
plt.title('逻辑回归部分特征')  
plt.show()

