浙江大学

本科实验报告

回归模型

课程名称：　人工智能实验

姓　　名：

学　　院：　信息与电子工程学院

专　　业：　信息工程

学　　号：

指导老师：　胡浩基、魏准

2023 年 6 月 4 日

一、 实验题目

1. 线性回归

编写程序，通过产生的附加噪声的随机数据，（1）做线性回归（即求出 ）；（2）利用取得的回归模型，对 x=0 和 x=2 两个点做预测；（3）对上述随机数据和预测结果利用 plot 可视化。

2. 多项式回归

通过网上调研了解 Scikit -Learn 中的 LinearRegression 模块，编写程序，利用 LinearRegression 模块：（1）对上述产生的二级多项式数据（X_poly, y）做线性回归。（2）可视化回归模型和数据。

3. 逻辑回归

（1）查阅网上资料，通过 sklearn.linear_model 中的 LogisticRegression，利用花瓣宽度，实现一个分类器检测维吉亚鸢尾花。（2）可视化花瓣宽度 0-3 cm 之间的鸢尾花模型估算出的概率

二、 实验代码

1. 线性回归

Linear Regression

```
import matplotlib.pyplot as plt
import numpy as np

x = 2 * np.random.rand(100, 1)
y = 4 + 3 * x + np.random.randn(100, 1)
x1 = np.c_[np.ones(100 , float) , x]

theta= np.dot(np.dot(np.linalg.inv(np.dot(np.transpose(x1), x1)), np.transpose(x1)), y)

print('%c = [%f, %f]' %(chr(952), theta[0], theta[1]))
y1 = theta[0] + theta[1] * 0
print('x = 0 , y = %f' %(y1))
y2 = theta[0] + theta[1] * 2
print('x = 2 , y = %f' %(y2))
y3 = theta[0] + theta[1] * x

plt.scatter(x, y, label='original')
plt.plot(x, y3, color='red', label='Predict')
plt.legend()
plt.xlabel('x')
plt.ylabel('y')
plt.savefig('figure1.png', dpi = 300)
plt.show
```

Polynomial Regression

```
import pandas as pd
import kNN
from sklearn.model_selection import train_test_split

df = pd.read_table('datingTestSet2.txt',sep='\s+', names = ['A', 'B', 'C', 'Y'])
# 对特征进行归一化处理
df2 = df.iloc[:, :3]
df2 = (df2-df2.mean())/df2.std()
lable =df.iloc[:,3:4]
df2.loc[:, 'Y'] = lable
# 对数据集进行测试集和训练集划分，90%作为训练集，10%作为测试集
X_train, X_test, Y_train, Y_test = train_test_split(df2.iloc[:, :3], df2.Y, train_size
    =.90)
# 将DataFrame格式转化为numpy格式处理
group = X_train.values
label = Y_train.values
length = len(X_test)
X_test.iloc[0:1,:]
# res以储存测试结果
res = []
# 设置错误正确数count以计算正确率
Tnum = 0
Fnum = 0
for i in range(length):
    inX = X_test.iloc[i:i+1 , :].values
    res.append(kNN.classify0(inX, group, label, k = 3))
    if(kNN.classify0(inX, group, label, k = 3) == Y_test.values[i]):
        Tnum += 1
    else:
        Fnum += 1
res1 = pd.DataFrame(data = res, columns=['TestResult'])
Y_test.reset_index(inplace=True,drop=True)
res1.loc[:, 'OriginTest'] = Y_test

print('前20个数据测试结果和原数据比较')
print('————————————————————————')
print(res1.head(20))
print('————————————————————————')
print('正确率%.2f%%' %(100*Tnum/(Tnum+Fnum)))
```

2. 多项式回归

Logistic Regression

```

import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
import numpy as np
```

```
from sklearn import datasets

iris = datasets.load_iris()
list(iris.keys())
X = iris["data"][:, 3:]
y = (iris["target"] == 2).astype(int)

model = LogisticRegression(random_state=1, solver='liblinear')
model.fit(X, y)

x_test_range = np.linspace(0, 3, 1000).reshape(-1, 1)
y_proba = model.predict_proba(x_test_range)

plt.plot(x_test_range, y_proba[:, 1], color = 'red', linewidth=1, label="Iris—Virginica")
plt.plot(x_test_range, y_proba[:, 0], color = 'blue', linewidth=1, label="Not Iris—Virginica")
plt.legend()
plt.savefig('figure3.png', dpi = 300)
plt.show
```

3. 多项式回归

三、 实验结果

1. 线性回归

结果如下：

$= [4.122495, 2.863779]$

模型为：

$$y = 4.122495 + 2.863779x$$

预测情况：

x = 0 , y = 4.122495

x = 2 , y = 9.850053

结果可视化如下图：

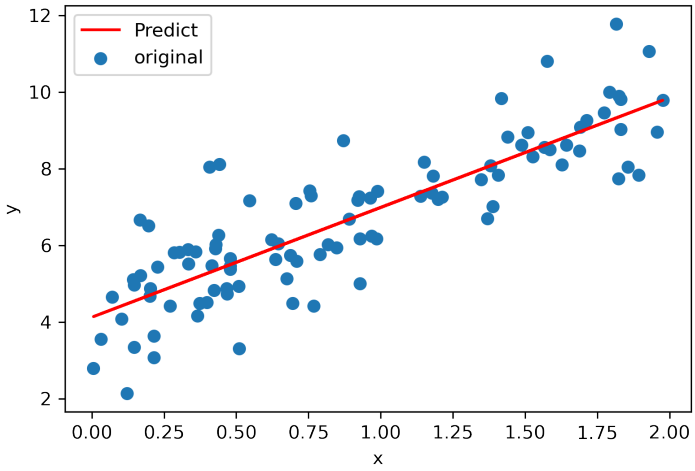

图 1: 线性回归结果可视化

2. 多项式回归

结果如下：
系数 = [0.94986314, 0.47823706]
截距 = [2.03896946]
模型为：

$$y = 0.47823706x^2 + 0.94986314x + 2.03896946$$

结果可视化如下图：

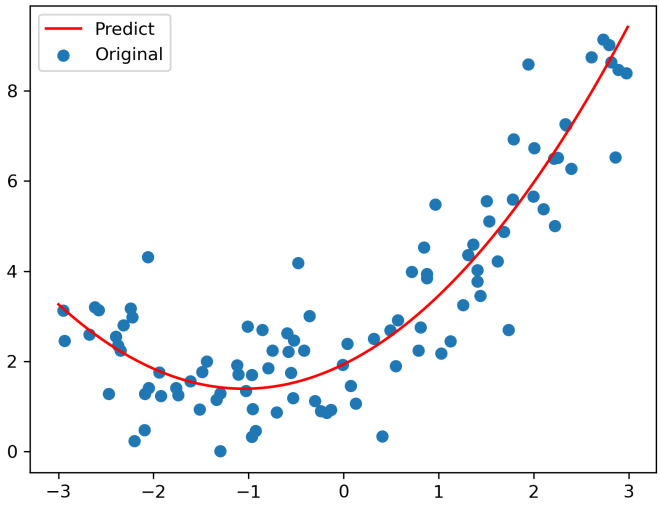

图 2: 多项式回归结果可视化

3. 逻辑回归

结果可视化如下图：

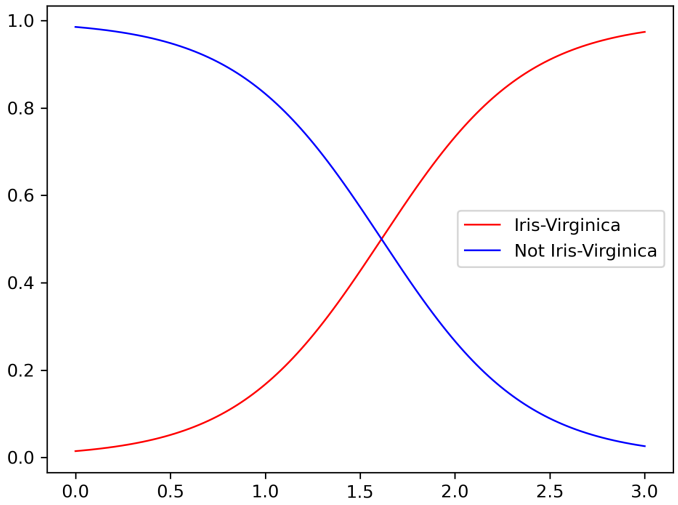

图 3: 逻辑回归结果可视化