

# 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： 姚桂涛

学 院： 信息与工程学院

专 业： 信息工程

学 号： 3190105597

指导老师： 胡浩基、魏准

2022 年 3 月 7 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  $\theta$ ）；（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

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 x = 2 * np.random.rand(100, 1)
5 y = 4 + 3 * x + np.random.randn(100, 1)
6 x1 = np.c_[np.ones(100, float), x]
7
8 theta = np.dot(np.dot(np.linalg.inv(np.dot(np.transpose(x1), x1)), np.transpose(x1)), y)
9
10 print('%c = [%f, %f]' % (chr(952), theta[0], theta[1]))
11 y1 = theta[0] + theta[1] * 0
12 print('x = 0, y = %f' % (y1))
13 y2 = theta[0] + theta[1] * 2
14 print('x = 2, y = %f' % (y2))
15 y3 = theta[0] + theta[1] * x
16
17 plt.scatter(x, y, label='original')
18 plt.plot(x, y3, color='red', label='Predict')
19 plt.legend()
20 plt.xlabel('x')
21 plt.ylabel('y')
22 plt.savefig('figure1.png', dpi = 300)
23 plt.show
```

## Polynomial Regression

```

1 import pandas as pd
2 import kNN
3 from sklearn.model_selection import train_test_split
4
5 df = pd.read_table('datingTestSet2.txt', sep='\s+', names = ['A', 'B', 'C', 'Y'])
6 # 对特征进行归一化处理
7 df2 = df.iloc[:, :3]
8 df2 = (df2 - df2.mean()) / df2.std()
9 lable = df.iloc[:, 3:4]
10 df2.loc[:, 'Y'] = lable
11 # 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
12 X_train, X_test, Y_train, Y_test = train_test_split(df2.iloc[:, :3], df2.Y, train_size
    = .90)
13 # 将DataFrame格式转化为numpy格式处理
14 group = X_train.values
15 label = Y_train.values
16 length = len(X_test)
17 X_test.iloc[0:1, :]
18 # res以储存测试结果
19 res = []
20 # 设置错误正确数count以计算正确率
21 Tnum = 0
22 Fnum = 0
23 for i in range(length):
24     inX = X_test.iloc[i:i+1, :].values
25     res.append(kNN.classify0(inX, group, label, k = 3))
26     if(kNN.classify0(inX, group, label, k = 3) == Y_test.values[i]):
27         Tnum += 1
28     else:
29         Fnum += 1
30 res1 = pd.DataFrame(data = res, columns=['TestResult'])
31 Y_test.reset_index(inplace=True, drop=True)
32 res1.loc[:, 'OriginTest'] = Y_test
33
34 print('前20个数据测试结果和原数据比较')
35 print('-----')
36 print(res1.head(20))
37 print('-----')
38 print('正确率%.2f%%' % (100 * Tnum / (Tnum + Fnum)))

```

## 2. 多项式回归

## Logistic Regression

```

1
2 import matplotlib.pyplot as plt
3 from sklearn.linear_model import LogisticRegression
4 import numpy as np

```

```
5 from sklearn import datasets
6
7 iris = datasets.load_iris()
8 list(iris.keys())
9 X = iris["data"][:, 3:]
10 y = (iris["target"] == 2).astype(int)
11
12 model = LogisticRegression(random_state=1, solver='liblinear')
13 model.fit(X, y)
14
15 x_test_range = np.linspace(0, 3, 1000).reshape(-1, 1)
16 y_proba = model.predict_proba(x_test_range)
17
18 plt.plot(x_test_range, y_proba[:, 1], color = 'red', linewidth=1, label="Iris—Virginica")
19 plt.plot(x_test_range, y_proba[:, 0], color = 'blue', linewidth=1, label="Not Iris—Virginica")
20 plt.legend()
21 plt.savefig('figure3.png', dpi = 300)
22 plt.show
```

### 3. 多项式回归

## 三、 实验结果