西北大学

智能信息系统综合实践

实验报告

题　　目：回归分析

年　　级：2019

专　　业：软件工程

姓　　名：郭锐哲

一、题目（原题目）

1、在摩擦实验中，当电压为-285 V 时电机开启，之后的时间与速度关系如表所示。

时间/ms	10	30	50	70	90	110	130	150	170
速度/(m/s)	0	0.066	0.088	0.1539	0.1979	0.2419	0.2639	0.3079	0.3299
时间/ms	190	210	230	250	270	290	310	330	350
速度/(m/s)	0.3738	0.3958	0.4398	0.4398	0.4618	0.4838	0.5498	0.5498	0.5718
时间/ms	370	390	410	430	450	470	490		
速度/(m/s)	0.5938	0.6158	0.6158	0.6377	0.6377	0.6597	0.6597		

（1）编程实现拟合模型，对下列各函数模型进行曲线拟合，画出拟合图，求出函数表达式以及误差平方和，线性拟合，多项式二次拟合，多项式三次拟合，对数函数拟合，幂函数拟合。（最小二乘法）

（2）分析各种曲线的拟合结果与误差，选择最佳模型，画出速度与时间的拟合曲线及 200 MS 时的加速度。

2、使用例题 1 中的数据生成方法，生成一组数据，并实现其多项式回归。假设多项式为：

$$h(x) = a1 + a2 * x + a3 * x^2$$

要求画出散点图和拟合后的曲线。（梯度下降法）

3、使用鸢尾花数据中前两类别的样本，任取其中两维特征构建逻辑回归分类器，并对原始数据和分类边界进行可视化。（梯度下降法）

二、解题步骤（思路 + 代码）

1. 本题是考察多个模型的拟合，有多项式拟合，对数函数拟合，幂函数拟合，解决思路针对不同拟合模型构造函数原型，并定义一个普适性的误差平方和函数，紧接着使用**最小二乘法**拟合函数对不同原型函数进行参数估计，对于估计出来的参数，回代原始数据，并计算拟合**误差平方和**，从而对不同的拟合结果进行**评估**，以下是解题详细步骤：

Step1：载入数据

```
# ============= 第一小问求解 =============

# 原始数据样本, 给时间除以1000单位变为s
time = np.array([10, 30, 50, 70, 90, 110, 130, 150, 170, 190, 210, 230, 250,
                 270, 290, 310, 330, 350, 370, 390, 410, 430, 450, 470, 490]) / 1000

speed = np.array([0, 0.066, 0.088, 0.1539, 0.1979, 0.2419, 0.2639, 0.3079,
                  0.3299, 0.3738, 0.3958, 0.4398, 0.4398, 0.4618, 0.4838,
                  0.5498, 0.5498, 0.5718, 0.5938, 0.6158, 0.6158, 0.6377,
                  0.6377, 0.6597, 0.6597])
```

图 1 问题原始数据录入

Step2：定义三种不同拟合模型的原型函数

```python
# 多项式拟合函数
def polynomial(p, x):
    f = np.poly1d(p)
    return f(x)

# 对数拟合函数
def logarithm(p, x):
    a1, a0 = p
    return a0 + a1 * np.log(x)

# 幂函数拟合函数
def powerFun(p, x):
    b, a = p
    return a * x ** b
```

图 2 三种拟合模型的原型定义

在这一步定义的三个函数中，第一个函数适用于题设所提到的线性拟合、多项式二次拟合、多项式三次拟合，**参数 p** 的长度即对应了不同拟合类型；后面两个函数分别对应对数函数拟合，幂函数拟合。

Step3:定义**训练函数**（包含可视化拟合结果）

```python
def showResult(X, Y, n, fun, title='拟合方式'):
    # 定义误差函数
    def error(p0, x, y, function):
        return function(p0, x) - y

    # 利用scipy库的最小二乘拟合函数对数据进行拟合
    p = np.linspace(1, 1, n)
    para = leastsq(error, p, args=(X, Y, fun))
    y0 = fun(para[0], X)

    # 对拟合结果进行可视化并进行误差分析
    plt.rcParams['font.sans-serif'] = ['SimHei']
    plt.rcParams['axes.unicode_minus'] = False
    plt.scatter(X, Y, color="blue", label='original data')
    plt.plot(X, y0, color="red", label='fitted curve')
    plt.legend(loc="lower right")
    title = title + "\n(" + "参数列表:" + str(para[0]) + ")"
    plt.title(title)
    # 计算拟合平方误差和
    loss = np.sum((Y - y0) ** 2)
    plt.text(x=0.3, y=0.1,
             s=str("误差平方和: %.4f" % loss),
             fontdict=dict(fontsize=15, color='r', family='SimHei')
             )
    plt.xlabel("时间(s)")
    plt.ylabel("速度(m/s)")
    plt.show()
    return para
```

图 3 训练函数以及可视化函数

这一步的核心是引用了 scipy 中的 **leastsq** 函数进行最小二乘拟合，在此之前定义了误差函数，即预测值-真实值，之后即是定义初始参数并代入函数进行计算便可得出模型拟合的结果，之后根据训练得到的参数进行误差平方和的计算以及可视化即可。

Step4：实施训练（包含了可视化）：

```
para1 = showResult(time, speed, 2, polynomial, "线性拟合")
para2 = showResult(time, speed, 3, polynomial, "多项式二次拟合")
para3 = showResult(time, speed, 4, polynomial, "多项式三次拟合")
para4 = showResult(time, speed, 2, logarithm, "对数拟合")
para5 = showResult(time, speed, 2, powerFun, "幂函数拟合")
```

图 4 训练五个不同的模型

Step5：结果展示及分析：

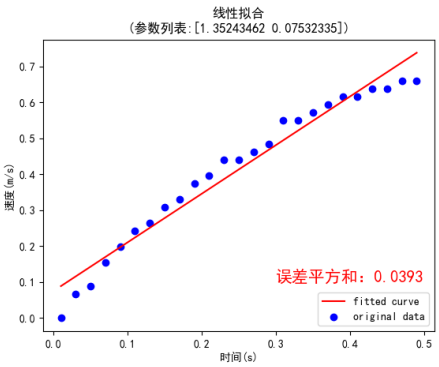
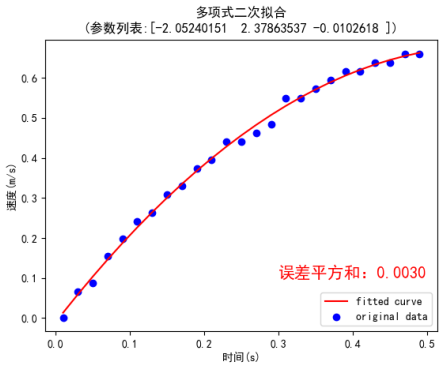


图 5 线性拟合结果



6 多项式二次拟合结果

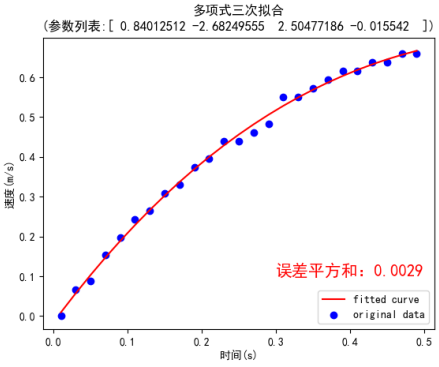


图 7 多项式三次拟合结果

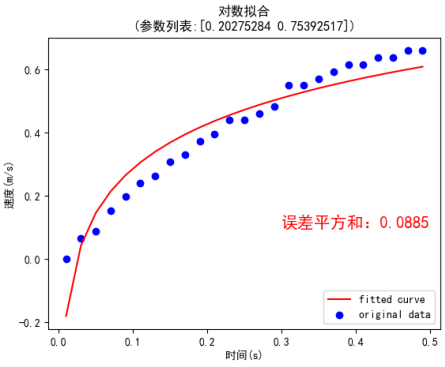


图 8 对数拟合结果

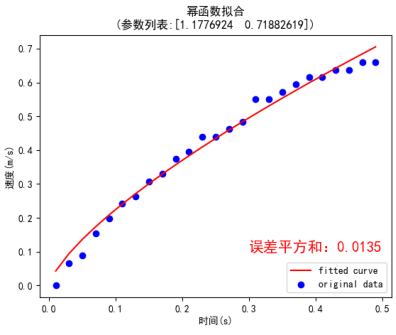


图 9 幂函数拟合结果

表 1 拟合结果对比

拟合函数模型	拟合函数表达式	拟合误差平方和
线性拟合	$y=1.3524x+0.0753$	0.0393
多项式二次拟合	$y=-2.0524x^2+2.3786x-0.0103$	0.0030
多项式三次拟合	$y=0.8401x^3-2.6825x^2+2.5048x-0.0155$	0.0029
对数拟合	$y=0.2028\log x+0.7539$	0.0885
幂函数拟合	$y=1.1777x^{0.7188}$	0.0135

结合上表、本题实际背景，在误差很小的情况下，更倾向于选择多项式二次函数进行拟合，对拟合函数**求导**得电机**加速度方程**为 $y'=-4.1048x+2.3786$，可以得到 200Ms 处的加速度为 1.5576m/s^2。

```
72      # ============== 第二小问求解 ==============
73      def accSpeed(p, x):
74          return 2 * p[0] * x + p[1]
75
76      res = accSpeed(para2[0], 200 / 1000)
77      print("200ms处的加速度为:", res, "m/s^2")
78
        pra2()
class2 ×
D:\software\python\Conda3\envs\machineLearning\python.exe
200ms处的加速度为: 1.557674765490842 m/s^2
```

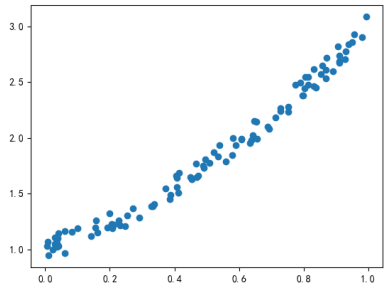
图 9 第二小问求解

2. 读题意可初步得到本题题解分为三步走：生成数据-梯度下降求解-拟合曲线可视化。得到解题路线后，按部就班实现即可：

Step1：仿照例题 1 生成数据（含可视化）

```
# 生成待拟合的数据并可视化
n = 100
x = np.zeros(n)
y = np.zeros(n)
for i in range(n):
    x[i] = random()
    y[i] = random() * 0.2 - 0.1 + x[i] * x[i] + x[i] + 1
plt.scatter(x, y)
plt.show()
```

图 10 随机生成数据图



11 初始数据散点图

Step2：**梯度下降法**求解参数

```
# 系数随机赋初值
a2, a1, a0 = random(), random(), random()
h = np.zeros(n)
lr = 0.3 # 学习率设置为0.3
# 梯度下降法求参数
for i in range(1500):
    sum1, sum2, sum3 = 0, 0, 0
    for j in range(n):
        h[j] = a2 * x[j] ** 2 + a1 * x[j] + a0
        sum1 += h[j] - y[j]
        sum2 += (h[j] - y[j]) * x[j]
        sum3 += (h[j] - y[j]) * x[j] ** 2
    a2 -= lr * sum3 / n
    a1 -= lr * sum2 / n
    a0 -= lr * sum1 / n
```

图 12 梯度下降法训练模型

在这一步，先定义**初始参数**以及**学习率**，**迭代次数**设置为 1500 次，然后循环迭代更新参数即可。

Step4：拟合曲线可视化

```
# 对拟合结果进行可视化并进行结果分析
x0 = np.linspace(0, 1, n)
for i in range(n):
    h[i] = a2 * x0[i] ** 2 + a1 * x0[i] + a0
plt.scatter(x, y, label="original data")
plt.plot(x0, h, color="red", label="fitted curve")
plt.legend()
plt.show()
```

图 13 拟合曲线可视化

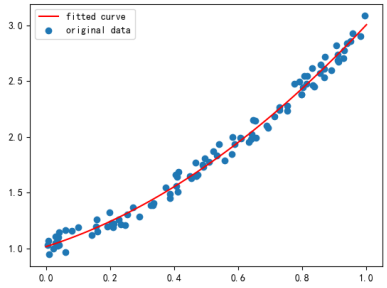


图 14 拟合曲线和原始数据

拟合曲线表达式： $y = 0.9366x^2 + 1.0660x + 0.9786$

3. 鸢尾花是经典的计息学习数据集，由 150 个样本组成，三种鸢尾花、四种花的特征属性，根据题意，提取前两种花、任意两种特征，然后进行**逻辑回归**，并根据回归结果可视化出**分类边界**，综上所述，该题依旧是三步走：数据-模型-可视化：

Step1-1：提取数据

```
iris = load_iris()
# 取鸢尾花数据集前两类的前两维数据
x = iris.data[:100, :2]
y = iris.target[:100]
```

图 15 鸢尾花数据处理

Step1-2:原始数据可视化

```
# 可视化数据集
plt.scatter(x[:50, 0], x[:50, 1], color="red", label="Setosa")
plt.scatter(x[50:, 0], x[50:, 1], color="green", label="Versicolor")
plt.legend()
plt.xlabel("Sepal length")
plt.ylabel("Sepal width")
plt.show()
```

图 16 鸢尾花数据集（部分）可视化

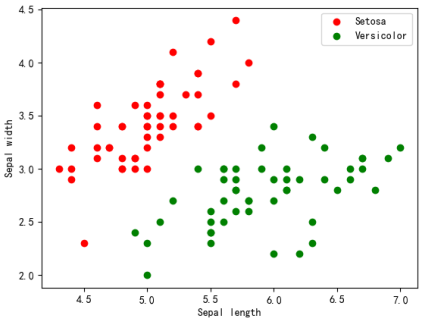


图 17 原始数据可视化

Step2：回归训练

```
# 逻辑回归分类，采用梯度下降法
lr = LogisticRegression(solver="sag", multi_class='ovr')
lr.fit(x, y)
```

图 18 sklearn 库自带逻辑回归训练

在这一步中，采用 sklearn 库中的 **LogisticRegression** 回归函数对模型进行训练，优化算法采用 sag，即用随机梯度下降来更新参数，后面的参数对此题无意义（已经是二分类问题了，采用默认的 ovr（一对多）模式即可）。

Step3：分类边界可视化

```
plt.scatter(x[:50, 0], x[:50, 1], color="red", label="Setosa")
plt.scatter(x[50:, 0], x[50:, 1], color="green", label="Versicolor")

length = x[:, 0]

def f(para1):
    return (-lr.coef_[0][0] * para1 - lr.intercept_[0]) / lr.coef_[0][1]

width = f(length)
plt.plot(length, width, "red")
plt.legend()
plt.xlabel("Sepal length")
plt.ylabel("Sepal width")
plt.show()
```

图 19 边界可视化

在这一步中，根据公式 $h = ax_1 + bx_2 + c$ 可以反向计算出两个属性中的另一个，故而先提取出属性 1，然后代入公式 $x_2 = (-ax_1 - c)/b$ 即可得到属性 2 的预测值，亦即可绘制出分类模型的分类边界如**图 20** 所示。

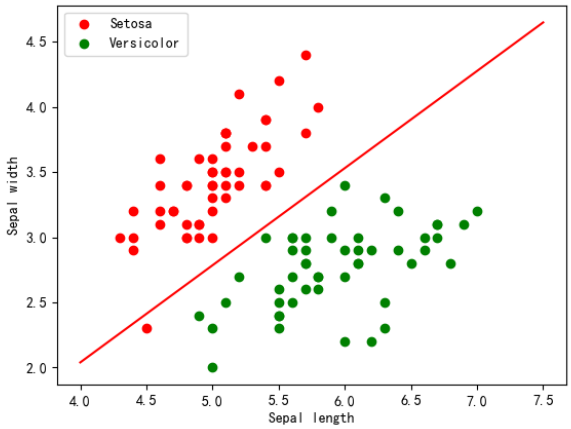


图 20 分类边界可视化

三、总结（心得体会）

在本次实践过程中，我实现了从理论到实践的跨越，在上机操作之前，首先复习了在机器学习课程中学到的几种回归模型，通过老师 ppt 上细致的公式推导，深入理解了几种模型的内部机理，紧接着通过老师留下的 3 个练习题，上机实现不同的模型，在编写代码过程中再次体会几种模型的实现细节（如：梯度下降法的手动实现），最后，可视化作为数据分析的重要工具，在上一次实践中进行了初步学习，在本次习题中，每个模型最后都对拟合结果进行了可视化，在校验拟合结果的同时，也是对上节课所学知识的一次复习和巩固。