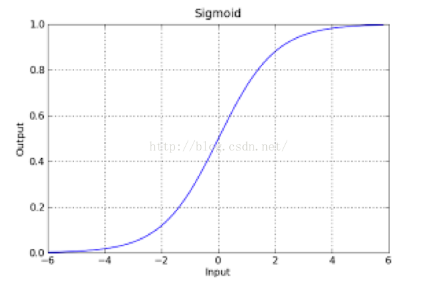
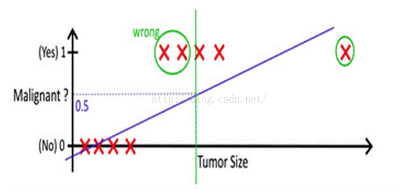
逻辑回归

1. 简介

逻辑回归是一种常见的分类算法，最常见的应用场景就是预测和判别，例如癌症的预测、某人患某种疾病的概率等等，这些都是一种两分类问题，即输出的结果只有两种，分别代表两个类别。

1. 逻辑回归与线性回归

在上述的两分类问题中，线性回归通过设定阈值的方式很难完成一个鲁棒性很好的分类器，对于这种不按套路出牌的数据点，线性回归实在是无能为力，而逻辑回归所用到的sigmoid函数就可以很好的解决这一问题。



Sigmod函数的输入范围是，而输出结果在0到1之间，这恰好满足了概率的分布要求，而且这是一个单调上升的函数，具有良好的连续性。

1. 逻辑回归的求解过程
2. 寻找预测函数
3. 构造代价函数
4. 通过特定算法使代价函数最小并求得回归参数
5. sklearn中的代码示例

由于sklearn中我们所需函数都已经被定义好了，所以我们直接使用就可以，这大大简化了我们的工作量

4.1 实验数据:

鸢尾花数据集或许是最有名的模式识别测试数据。早在1936年，模式识别的先驱Fisher

就在论文"The use of multiple measurements in taxonomic problems"中使用了它（直

至今日该论文仍然被频繁引用）。该数据集包括 3 个鸢尾花类别，每个类别有 50 个样本。

其中一个类别是与另外两类线性可分的，而另外两类不能线性可分。

由于 Fisher 的最原始数据集存在两个错误(35 号和 38 号样本)，因此，我们实验中使

用的是修正过的数据。

该数据可以从这个链接直接下载得到：<http://archive.ics.uci.edu/ml/datasets/Iris>

4.2 数据描述

该数据集共包括 150 行，每行为 1 个样本。每个样本有 5 个字段，分别是：花萼长度 (单位 cm)、花萼宽度(单位：cm)、花瓣长度(单位：cm)、花瓣宽度(单位：cm)、类别（共 3类，分别是：Iris Setosa、Iris Versicolour、Iris Virginica）

* 1. 实验过程

import numpy as np

from sklearn.linear\_model import LogisticRegression

import matplotlib.pyplot as plt

def iris\_type(s):

it = {b'Iris-setosa': 0,b 'Iris-versicolor': 1,b 'Iris-virginica': 2}

return it[s]

if \_\_name\_\_ == "\_\_main\_\_":

path = 'E:\bigdata\iris.data' # 数据文件路径

# 路径，浮点型数据，逗号分隔，第 4 列使用函数 iris\_type 单独处理

data = np.loadtxt(path, dtype=float, delimiter=',', converters={4:iris\_type})

# 将数据的 0 到 3 列组成 x ，第 4 列得到 y

x, y = np.split(data, (4,), axis=1)

# 为了可视化，仅使用前两列特征

x = x[:, :2]

logreg = LogisticRegression() # Logistic 回归模型



logreg.fit(x, y.ravel()) # 根据数据 [x,y] ，计算回归参数

# 画图

N, M = 500, 500 # 横纵各采样多少个值

x1\_min, x1\_max = x[:, 0].min(), x[:, 0].max() # 第 0 列的范围

x2\_min, x2\_max = x[:, 1].min(), x[:, 1].max() # 第 1 列的范围

t1 = np.linspace(x1\_min, x1\_max, N)

t2 = np.linspace(x2\_min, x2\_max, M)

x1, x2 = np.meshgrid(t1, t2) # 生成网格采样点

x\_test = np.stack((x1.flat, x2.flat), axis=1) # 测试点

y\_hat = logreg.predict(x\_test) # 预测值

y\_hat = y\_hat.reshape(x1.shape) # 使之与输入形状相同

plt.pcolormesh(x1, x2, y\_hat, cmap=plt.cm.prism) # 预测值的显示

plt.scatter(x[:, 0], x[:, 1], c=np.squeeze(y), edgecolors='k', cmap=plt.cm.prism)

# 样本的显示

plt.xlabel('Sepal length')

plt.ylabel('Sepal width')

plt.xlim(x1\_min, x1\_max)

plt.ylim(x2\_min, x2\_max)

plt.grid()

plt.show()

# 训练集上的预测结果

y\_hat = logreg.predict(x)

y = y.reshape(-1)

print(y\_hat.shape)

print(y.shape)

result = y\_hat == y

print(y\_hat)

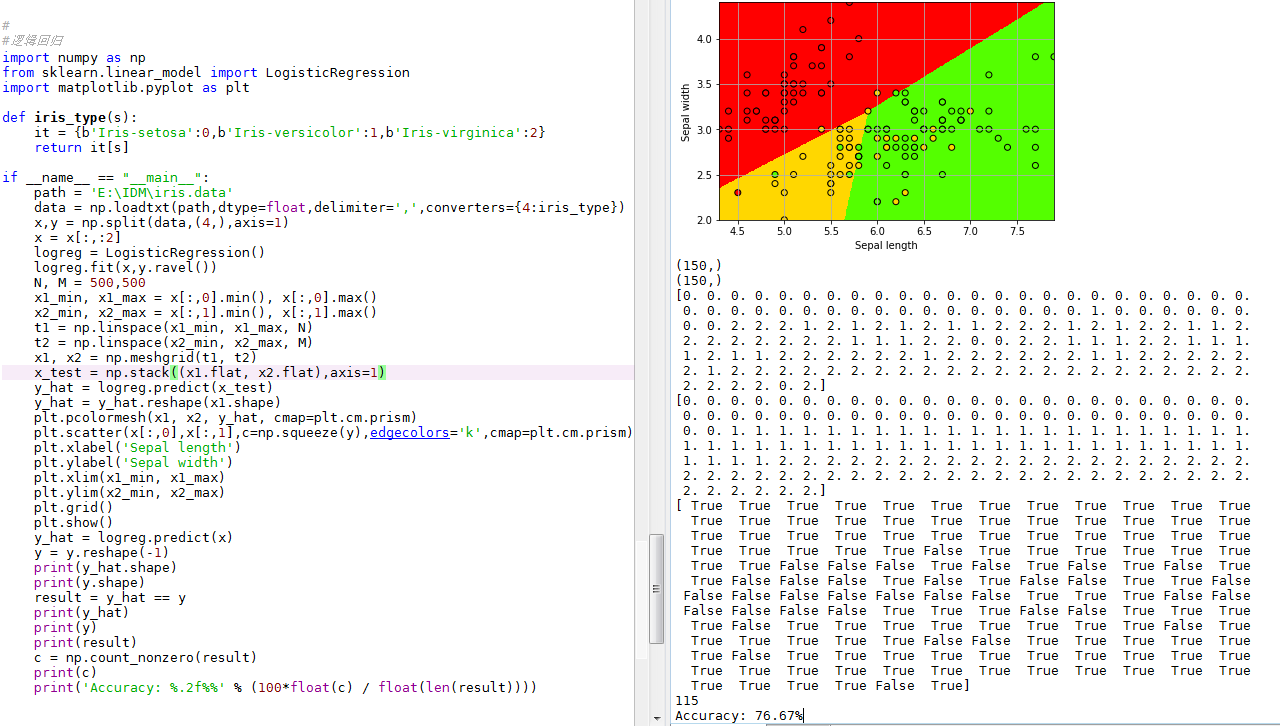
print(y)

print(result)

c = np.count\_nonzero(result)

print(c)

print('Accuracy: %.2f%%' % (100 \* float(c) / float(len(result))))



1. 分析与思考

（1）仅仅使用两个特征：花萼长度和宽度，在 150 个样本中，有 115 个分类正确，正确率为 76.67%——分类效果基本堪用，但并不十分令人满意。

（2）请尝试使用更多的特征（如 4 个全部使用），再次运行该程序，比较正确率

