逻辑回归算法实现

实验目标

通过本案例的学习和课后作业的练习：

1. 了解逻辑回归算法的基本原理；
2. 掌握如何通过调用Sklearn框架实现逻辑回归。

你也可以将本案例相关的 ipynb 学习笔记分享到 AI Gallery Notebook (https://marketplace.huaweicloud.com/markets/aihub/notebook/list/) 版块获得成长值 (https://marketplace.huaweicloud.com/markets/aihub/article/detail/?content_id=9b8d7e7a-a150-449e-ac17-2dcf76d8b492)，分享方法请查看此文档 (https://marketplace.huaweicloud.com/markets/aihub/article/detail/?content_id=8afec58a-b797-4bf9-acca-76ed512a3acb)。

案例内容介绍

logistic回归又称logistic回归分析，是一种广义的线性回归分析模型，常用于数据挖掘，疾病自动诊断，经济预测等领域。例如，探讨引发疾病的危险因素，并根据危险因素预测疾病发生的概率等。以胃癌病情分析为例，选择两组人群，一组是胃癌组，一组是非胃癌组，两组人群必定具有不同的体征与生活方式等。因此因变量就为是否胃癌，值为“是”或“否”，自变量就可以包括很多了，如年龄、性别、饮食习惯、幽门螺杆菌感染等。自变量既可以是连续的，也可以是分类的。然后通过logistic回归分析，可以得到自变量的权重，从而可以大致了解到底哪些因素是胃癌的危险因素。同时根据该权值可以根据危险因素预测一个人患癌症的可能性。

本案例推荐的理论学习视频：

- 《AI技术领域课程--机器学习》 逻辑回归 (https://education.huaweicloud.com/courses/course-v1:HuaweiX+CBUCNXE086+Self-paced/courseware/5961d6a3d17e432cb516f37f4ff06941/cf683fa8cc5e497088b94af185ef8c12/)

注意事项

1. 如果您是第一次使用 JupyterLab，请查看《ModelArts JupyterLab使用指导》(https://marketplace.huaweicloud.com/markets/aihub/article/detail/?content_id=03676d0a-0630-4a3f-b62c-07fba43d2857)了解使用方法；
2. 如果您在使用 JupyterLab 过程中碰到报错，请参考《ModelArts JupyterLab常见问题解决办法》(https://marketplace.huaweicloud.com/markets/aihub/article/detail/?content_id=9ad8ce7d-06f7-4394-80ef-4dbf6cfb4be1)尝试解决问题。

实验步骤

1、导入相关数据模块

逻辑回归的调用

In [1]:
```
import numpy as np
# 从sklearn导入LogisticRegression方法
from sklearn.linear_model import LogisticRegression
# 导入划分训练集和测试集的方法
from sklearn.model_selection import train_test_split
import moxing as mox
import os
```

```
INFO:root:Using MoXing-v1.17.3-

INFO:root:Using OBS-Python-SDK-3.20.7
```

2、读取数据

In [2]:
```
if not os.path.exists('info.txt'):
    mox.file.copy('obs://modelarts-labs-bj4-v2/course/hwc_edu/machine_learning/datasets/logistic_regression/info.txt', 'info.txt')
data = np.loadtxt("./info.txt",
                  delimiter=",")  # 此处需要将数据上传至obs内，并将其放置在与项目同一个工作路径下。本地路径直接写文件名称即可，在ModelArts中前面需要增加"./work/"便于ModelArts检测数据位置。
print(data)
```

```
[[2.697e+03 6.254e+03 1.000e+00]

 [1.872e+03 2.014e+03 0.000e+00]

 [2.312e+03 8.120e+02 0.000e+00]

 [1.983e+03 4.990e+03 1.000e+00]

 [9.320e+02 3.920e+03 0.000e+00]

 [1.321e+03 5.583e+03 1.000e+00]

 [2.215e+03 1.560e+03 0.000e+00]

 [1.659e+03 2.932e+03 0.000e+00]

 [8.650e+02 7.316e+03 1.000e+00]

 [1.685e+03 4.763e+03 0.000e+00]

 [1.786e+03 2.523e+03 1.000e+00]]
```

3、划分训练集及测试集

In [3]:
```
# 划分训练集和测试集，测试集占数据集的30%，训练集占数据集的70%
train_x, test_x, train_y, test_y = train_test_split(data[:, 0:2], data[:, 2], test_size=0.3)
# train_test_split(x,y,test_size=0.3)      #参数test_size测试集占比；x:数据集；y:数据集的目标值
```

4、调用sklearn里面的方法

In [4]:
```
model = LogisticRegression()
```

5、通过训练集得到训练后的模型