1. 逻辑回归思想

逻辑回归用 sigmod 函数：

$$g(z) = \frac{1}{1 + e^{-z}}$$

训练集

$$\{x_i: y_i\}_{i=1}^{m}$$

对于二分类模型

“正类”为：

$$P(y = 1|x) = \frac{e^{w^T + b}}{1 + e^{w^T + b}}$$

“负类”为：

$$P(y = 0|x) = \frac{1}{1 + e^{w^T + b}}$$

2. 用极大似然法估计对多分类对数做几率回归

可以得到极大似然函数：

$$L(w, b) = \sum_{i=1}^{m} lnP(y_i|x_i, w, b)$$

要选择合适的参数w, b使输入参数x_i和输出y_i关系更为紧密，即使得$L(w, b)$最大，

$$(w^*, b^*) = argmaxL(w, b)$$

将这两类模型统一起来：

$$P(y_i|x_i, w, b) = y_i P_1(x_i; w, b) + (1 - y_i) P_0(x_i; w, b)$$

其中$y_i = 0,1$

令$x = (x; 1)$，$w^T x + b$算符为$\beta^T \hat{x}$，将极大似然函数代入可得，

$$L(\beta) = \sum_{i=1}^{m} \ln{(y_i \cdot \frac{e^{\beta^T \hat{x}}}{1 + e^{\beta^T \hat{x}}} + (1 - y_i) \cdot \frac{1}{1 + e^{\beta^T \hat{x}}})}$$

分离分母，

$$L(\beta) = \sum_{i=1}^{m} \ln\left(y_i \cdot e^{\beta^T \hat{x}} + (1 - y_i)\right) - \ln\left(1 + e^{\beta^T \hat{x}}\right)$$

又已知$y_i = 0 \text{ 或 } 1$，该问题等价于极小化下式，

$$L(\beta) = \sum_{i=1}^{m} -y_i \cdot e^{\beta^T \hat{x}} + \ln\left(1 + e^{\beta^T \hat{x}}\right)$$

对于二分类，目标函数为

$$L(\beta)=\sum_{i=1}^{m}(y_i\cdot\ln\left(e^{\beta^T\hat{x}}\right)-\ln\left(1+e^{\beta^T\hat{x}}\right))$$

等价于最小化

$$L(\beta)=\sum_{i=1}^{m}(-y_i\,\beta^T\hat{x}+\ln\left(1+e^{\beta^T\hat{x}}\right))$$

3. 二分类算法思路：

1. 输入训练集$\{x_i\colon c_i\}_{i=1}^{m}$
2. 初始化w,b，令$\beta=(w,b)$ $\hat{x}=(x;1)$
3. 求解参数w,b

1）牛顿法：

计算：

$$L(\beta)=\sum_{i=1}^{m}(-y_i\,\beta^T\hat{x_i}+\ln\left(1+e^{\beta^T\hat{x_i}}\right))$$

$$P_1(\hat{x_i};\beta^T)=\frac{1}{1+e^{\beta^T\hat{x_i}}}$$

$$\beta^*=argminL(\beta)$$

$$\frac{\partial L(\beta)}{\partial\beta}=-\sum_{i=1}^{m}\hat{x_i}(y_i-P_1(\hat{x_i};\beta))$$

$$\frac{\partial^2L(\beta)}{\partial\beta\partial\beta^T}=-\sum_{i=1}^{m}\hat{x_i}\hat{x_i}^TP_1(\hat{x_i};\beta)\left(1-P_1(\hat{x_i};\beta)\right)$$

$$\beta^{t+1}=\beta^t-\left(\frac{\partial^2L(\beta)}{\partial\beta\partial\beta^T}\right)^{-1}\cdot\frac{\partial L(\beta)}{\partial\beta}$$

迭代，得到w,b。

2）梯度下降法：

$$L(\beta)=\sum_{i=1}^{m}(y_i\cdot\ln P_1(\hat{x})+(1-y_i)\cdot\ln\left(1-P_1(\hat{x})\right))$$

等价于最小化：

$$L(\beta)=\sum_{i=1}^{m}(-y_i\,\beta^T\hat{x}+\ln\left(1+e^{\beta^T\hat{x}}\right))$$

$$h(\hat{x})=\frac{e^{\beta^T\hat{x}}}{1+e^{\beta^T\hat{x}}}$$

$$\frac{\partial L(\beta)}{\partial\beta_j}=-\sum_{i=1}^{n}(y_i-h(\hat{x_i}))\cdot\hat{x_i}$$

$$\beta_j \coloneqq \beta_j + \Delta\beta_j$$

$$\Delta\beta_j = -\alpha \cdot \frac{\partial L(\beta)}{\partial \beta_j}$$

$$\beta_j \coloneqq \beta_j + \alpha \cdot \sum_{i=1}^{n} (y_i - h(\widehat{x_i})) \cdot \widehat{x_i}$$

迭代，得到w, b。

4. 用测试样本集去通过概率大小分类，计算正确率。

4. 代码实现：

选用包含 LogisticRegression 函数的 sklearn 进行逻辑回归分类

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression,SGDRegressor,Ridge,LogisticRegression
from sklearn.metrics import mean_squared_error, classification_report
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
import pandas as pd
import numpy as np

def logistic():
    """
    逻辑回归做二分类进行乳腺癌预测，用数据集的 0.7 作为训练集，用数据集的 0.3
作为测试集。Pytorch 进行二分类要分每个维度分别进行，这里有 sklearn 内置逻辑回归
函数对数据进行分类。首先读取在线数据集，然后进行数据分割，按 0.7/0.3 分割为训练
集和测试集。
    """
    # 构造列标签名字
    column = ['Sample code number', 'Clump Thickness', 'Uniformity of Cell Size', 'Uniformity
of Cell Shape',
              'Marginal Adhesion', 'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland
Chromatin', 'Normal Nucleoli',
              'Mitoses', 'Class']
    # 读取在线数据集
    data = pd.read_csv(

"https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-
cancer-wisconsin.data",
        names=column)
    # print(data)
    # 对存在缺失值进行处理，替换为 nan
    data = data.replace(to_replace='?', value=np.nan)
    data = data.dropna()
    # 输出 data 的数据量和维度。
    print("数据量，维度：\n", data.shape)
    print("数据集：\n", data)
    # 进行数据的分割，用数据集的 0.7 作为训练集，用数据集的 0.3 作为测试集
    x_train, x_test, y_train, y_test = train_test_split(data[column[1:10]], data[column[10]],
test_size=0.3)
```

```
    # 输出训练样本两个类分别的样本量。
    print("训练样本类，数据量：\n", y_train.value_counts())
    # 输出测试样本两个类分别的样本量。
    print("测试样本类，数据量：\n", y_test.value_counts())
    # 进行标准化处理
    std = StandardScaler()
    x_train = std.fit_transform(x_train)
    x_test = std.transform(x_test)
    # 逻辑回归来对训练集进行训练
    lg = LogisticRegression(C=1.0)
    # 求得训练集 X 的均值、方差、最大值、最小值等固有属性
    lg.fit(x_train, y_train)
    print('回归系数\n', lg.coef_)
    # 训练后返回预测结果
    y_predict = lg.predict(x_test)
    # 输出预测结果计算出的决定系数 R^2
    print("拟合优度：", lg.score(x_test, y_test))
    # classification_report 函数用于显示主要分类指标．显示每个类的精确度，召回率，F1 值等
    print("分类指标：", classification_report(y_test, y_predict, labels=[2, 4], target_names=["良性", "恶性"]))

if __name__ == "__main__":
        logistic()
```

5. 输出结果：

```
数据量，维度：
 (683, 11)
数据集：
      Sample code number  Clump Thickness  ...  Mitoses  Class
0               1000025                5  ...        1      2
1               1002945                5  ...        1      2
2               1015425                3  ...        1      2
3               1016277                6  ...        1      2
4               1017023                4  ...        1      2
..                  ...              ...  ...      ...    ...
694              776715                3  ...        1      2
695              841769                2  ...        1      2
696              888820                5  ...        2      4
697              897471                4  ...        1      4
698              897471                4  ...        1      4

[683 rows x 11 columns]
训练样本类，数据量：
 2    312
4    166
Name: Class, dtype: int64
测试样本类，数据量：
 2    132
4     73
Name: Class, dtype: int64
```

```
回归系数
 [[ 1.19967514  0.59630373  1.06276511  0.39187684 -0.14101706  1.38601652
   0.65826599  0.71886856  0.49763396]]
拟合优度： 0.9560975609756097
分类指标：               precision    recall  f1-score   support

          良性       0.95      0.98      0.97       132
          恶性       0.97      0.90      0.94        73

    accuracy                           0.96       205
   macro avg       0.96      0.94      0.95       205
weighted avg       0.96      0.96      0.96       205
```