# 学习目标

|  |
| --- |
|  |
|  |

# 1.逻辑回归的应用场景

|  |
| --- |
|  |

# 2.逻辑回归的原理

|  |
| --- |
|  |
| 结果大于阈值,就属于正规类别,小于阈值,就不属于这个类别 |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

## 扩展:为什么逻辑回归可以用来做分类处理,是因为sigmoid函数的作用.把回归结果转换为分类结果

|  |
| --- |
|  |

# 3.逻辑回归API模块

|  |
| --- |
|  |

# 4.案例:癌症分类预测-良/恶性乳腺癌肿瘤预测

|  |
| --- |
|  |

## 代码实现: LogisticRegression\_breast\_cancer\_wisconsin.py

|  |
| --- |
| """   案例:使用逻辑回归算法进行威斯康辛州癌症分类预测  """  import pandas as pd  import numpy as np  from sklearn.model\_selection import train\_test\_split  from sklearn.linear\_model import LogisticRegression  from sklearn.preprocessing import StandardScaler  # 让pandas显示所有列  pd.set\_option("display.max\_columns", None)  # 1.获取数据,这个数据没有头 ,我们需要给他添加头,也就是特征名称  column\_name = ['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']  url = "https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data"  wisconsin = pd.read\_csv(url,names=column\_name)  # print(wisconsin)  # ok  # 2.数据处理,处理缺失值  #  1> 把?替换成np.nan  wisconsin = wisconsin.replace(to\_replace="?",value=np.nan)  #  2> 用pandas来处理nan ,这里直接删除  wisconsin.dropna(inplace=True) # 就地处理  # print(wisconsin.isnull().any())  # 全部返回False,说明所有的nan都已经处理完毕  # 3.数据集划分  #  1>确定特征值和目标值  features= wisconsin.iloc[:,1:-1]  targets = wisconsin["Class"]  #  2>数据集分割  x\_train,x\_test,y\_train,y\_test = train\_test\_split(features,targets,random\_state=6)  # 4.特征工程-标准化处理  transfer = StandardScaler()  x\_train = transfer.fit\_transform(x\_train)  x\_test  = transfer.transform(x\_test)  # 5.逻辑回归预估器流程  estimator = LogisticRegression()  estimator.fit(x\_train,y\_train)  # 查看模型  print("逻辑回归-权重系数为：\n", estimator.coef\_)  print("逻辑回归-偏置为：\n", estimator.intercept\_)  # 6.模型评估  # 1)直接对比预测值与真实值  y\_predict = estimator.predict(x\_test)  print("对比真实值与预测值\n",y\_predict == y\_test)  # 2)计算准确率  score = estimator.score(x\_test,y\_test)  print("准确率:",score) #准确率: 0.9766081871345029 |

### 效果,使用以前的评估方法:

|  |
| --- |
|  |

### 其实这样子还不够,我们需要使用新的对于这一类问题的评估方法

# 5.分类的评估方法

|  |
| --- |
|  |
|  |
|  |
| 精确率和召回率虽然好,但是也是有局限性的,就是当样本不均衡时效果不好召回率是很常用的,F1指标是衡量模型的稳健性的.,两个F1-score很高,下面精确率和召回率都很高,那么模型就很稳定. |
| 分类评估API  |  | | --- | |  | |
|  |
|  |
|  |
| 注意:当auc<0.5的时候是反向预测 |
|  |

# 6.总结

|  |
| --- |
|  |