机器学习のday_3

分类算法-逻辑回归与二分类

定义

逻辑回归（Logistic Regression）是机器学习中的一种分类模型算法，虽然名字中带有回归，但是它只是与回归有一定的联系。由于算法的简单高效，在实际中应用非常广泛。

原理

输入

$$h(w) = w_1x_1 + w_2x_2 + w_3x_3 \dots + b$$

激活函数

$$g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

- 回归的结果输入到sigmoid函数当中
- 输出结果：[0, 1]区间中的一个概率值，默认为0.5为阈值

解释

假设有两个类别A，B，并且假设我们的概率值为属于A(1)这个类别的概率值。现在有一个样本的输入到逻辑回归输出结果0.6，那么这个概率值超过0.5，意味着我们训练或者预测的结果就是A(1)类别。那么反之，如果得出结果为0.3那么，训练或者预测结果就为B(0)类别

损失及优化

$$cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{if } y=1 \\ -log(1 - h_\theta(x)) & \text{if } y=0 \end{cases}$$

$$cost(h_\theta(x), y) = \sum_{i=1}^{m} -y_i log(h_\theta(x)) - (1 - y_i) log(1 - h_\theta(x))$$

综合完整的损失函数

优化

同样使用梯度下降优化算法，去减少损失函数的值。这样去更新逻辑回归前面对应算法的权重参数，提升原本属于1类别的概率，降低原本是0类别的概率。

API

sklearn.linear_model.LogisticRegression(solver='liblinear', penalty='l2', C = 1.0)

- solver:优化求解方式（默认开源的liblinear库实现，内部使用了坐标轴下降法来迭代优化损失函数）
  - sag：随机平均梯度下降
- penalty：正则化的种类
- C：正则化力度

案例：肿瘤病人预测

默认将类别数量少的当做正例

1.导入数据

2.读取数据处理缺失值

3.去除特征目标值，分割数据集

4.进行标准化

5.逻辑回归进行训练与预测

log.fit(x_train, y_train)

精确率与召回率

混淆矩阵

精确率

预测结果为正例样本中真实为正例的比例（一般不关注）

召回率

真实为正例的样本中预测结果为正例的比例（查的全，对正样本的区分能力）

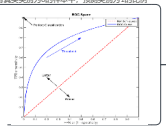
理解

- sklearn.metrics.classification_report(y_true, y_pred, labels=[], target_names=None)
  - y_true：真实目标值
  - y_pred：估计器预测目标值
  - labels:指定类别对应的数字
  - target_names：目标类别名称
  - return：每个类别精确率与召回率

API

衡量样本不均衡下的评估

- TPR = TP / (TP + FN)
  - 所有真实类别为1的样本中，预测类别为1的比例
- FPR = FP / (FP + FN)
  - 所有真实类别为0的样本中，预测类别为1的比例



ROC曲线

- AUC的概率意义是随机取一对正负样本，正样本得分大于负样本的概率
- AUC的最小值为0.5，最大值为1，取值越高越好
- AUC=1，完美分类器，采用这个预测模型时，不管设定什么阈值都能得出完美预测。绝大多数预测的场合，不存在完美分类器。
- 0.5<AUC<1，优于随机猜测。这个分类器（模型）妥善设定阈值的话，能有预测价值。
- AUC=0.5，跟随机猜测一样（例：丢铜板），模型没有预测价值。
- AUC<0.5，比随机猜测还差；但只要总是反预测而行，就优于随机猜测。因此不存在 AUC<0.5 的情况。

最终AUC的范围在[0.5, 1]之间，并且越接近1越好

AUC指标

```
from sklearn.metrics import roc_auc_score
```

- sklearn.metrics.roc_auc_score(y_true, y_score)
  - 计算ROC曲线面积，即AUC值
  - y_true:每个样本的真实类别，必须为0(反例),1(正例)标记
  - y_score:每个样本预测的概率值

API

总结

- AUC只能用来评价二分类
- AUC非常适合评价样本不平衡中的分类器性能
- AUC会比较预测出来的概率，而不仅仅是标签类

模型保存与加载

是什么

当训练或者计算好一个模型之后，那么如果别人需要我们提供结果预测，就需要保存模型（主要是保存算法的参数）

```
from sklearn.externals import joblib
```

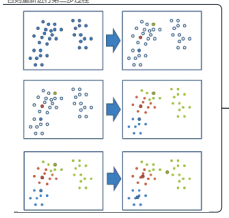
- 保存：joblib.dump(rf, "test.pkl")
- 加载：estimator = joblib.load("test.pkl")

API

无监督学习之K-means算法

步骤

- 1、随机设置K个特征空间内的点作为初始的聚类中心（K值一般是确定的）
- 2、对于其他每个点计算到K个中心的距离，未知的点选择最近的一个聚类中心点作为标记类别
- 3、接着对着标记的聚类中心之后，重新计算出每个聚类的新中心点（平均值）
- 4、如果计算得出的新中心点与原中心点一样，那么结束，否则重新进行第二步过程



```
sklearn.cluster.KMeans(n_clusters=8,init='k-means++')
```

- k-means聚类
- n_clusters:开始的聚类中心数量
- init:初始化方法，默认为'k-means ++'
- labels：默认标记的类型，可以和真实值比较（不是值比较）

API

案例：（待更新）

性能评估指标

$$sc_i = \frac{b_i - a_i}{\max(b_i, a_i)}$$

轮廓系数

对于每个点 为已聚类数据中的样本 ，b_i 为 到其它族群的所有样本的距离最小值，a_i 为: 到本身簇的距离平均值。最终计算出所有的样本点的轮廓系数平均值

```
sklearn.metrics.silhouette_score(X, labels)
```

- 计算所有样本的平均轮廓系数
- X：特征值
- labels：被聚类标记的目标值

API

性能评估指标

线性回归

定义

线性回归(Linear regression)是利用**回归方程(函数)**对一个或**多个自变量(特征值)和因变量(目标值)之间**关系进行建模的一种分析方式。

特点

特点：只有一个自变量的情况称为*单变量回归*，大于一个自变量情况的叫做*多元回归*

损失函数

每个样本的真实值与预测值的差值的平方求和

$$J(\theta) = (h_w(x_1) - y_1)^2 + (h_w(x_2) - y_2)^2 + \cdots + (h_w(x_m) - y_m)^2$$

$$= \sum_{i=1}^{m} (h_w(x_i) - y_i)^2$$

这种方法也叫做最小二乘法

优化算法

其实就相当于数学中的求导功能

法一：正则方程

X为特征值矩阵，y为目标值矩阵

$$w = (X^T X)^{-1} X^T y$$

缺点：当特征过多过复杂时，求解速度太慢并且得不到结果

法二：梯度下降 (Gradient Descent)

$$w_1 := w_1 - \alpha \frac{\partial cost(w_1 + w_1 x_1)}{\partial w_1}$$

$$w_1 := w_1 - \alpha \frac{\partial cost(w_0 + w_1 x_1)}{\partial w_1}$$

α为学习速率，需要手动指定（超参数），α乘以偏导表示方向沿着这个函数下降的方向，最后能找到山谷的最低点，然后更新W值使用：面对训练数据规模十分庞大的任务，能够找到较好的结果

API

sklearn.linear_model.LinearRegression(fit_intercept=True)

- 通过正规方程优化
- fit_intercept：是否计算偏置
- LinearRegression.coef_：回归系数
- LinearRegression.intercept_：偏置

正规方程

sklearn.linear_model.SGDRegressor(loss="squared_loss", fit_intercept=True, learning_rate ='invscaling', eta0=0.01)

- SGDRegressor实现了随机梯度下降学习，它支持不同的loss函数和正则化惩罚项来拟合线性回归模型。
  - loss:损失类型
  - loss="squared_loss": 普通最小二乘法
- fit_intercept：是否计算偏置
- learning_rate : string, optional
  - 学习率填充
  - 'constant': eta = eta0
  - 'optimal': eta = 1.0 / (alpha * (t + t0)) [default]
  - 'invscaling': eta = eta0 / pow(t, power_t)
  - power_t=0.25:存在父类当中
- 对于一个常数值的学习率来说，可以使用learning_rate='constant' ，并使用eta0来指定学习率。
- SGDRegressor.coef_：回归系数
- SGDRegressor.intercept_：偏置

梯度下降

回归当中的数据大小不一致，会导致结果影响较大，所以需要做标准化处理（前面K-近邻算法也进行了标准化，贝叶斯和随机森林没有）。

案例：波士顿房价

- 数据集的分割与标准化处理
- 回归预测
- 线性回归的算法效果评估

分析

回归性能评估（重要）

均方误差(Mean Squared Error) (MSE)评价机制

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y^i - \bar{y})^2$$

y:为预测值，$\bar{y}$为真实值

```
sklearn.metrics.mean_squared_error(y_true, y_pred)
```

- 均方误差回归损失
- y_true:真实值
- y_pred:预测值
- return:浮点数结果

过程

1.获取数据进行数据集分割

2.对数据进行标准化处理

3.使用线性回归的模型进训练预测

正规方程

lr.fit(x_train, y_train)

lr 之后已经得出了需要的结果

梯度下降

```
sgd = SGDRegressor(loss="squared_loss", fit_intercept=True, learning_rate="constant", eta0=0.0001)
```

可以尝试指定学习率（一般都很小）

4.调用predict去预测目标值

5.利用均方误差来评估回归性能

总结

小规模数据

正规方程（不能解决过拟合，一般不用）

岭回归

大规模数据

SGD梯度下降

岭回归

定义

岭回归，其实也是一种线性回归。只不过在算法建立回归方程时候，加上正则化的限制，从而达到解决过拟合的效果

API

sklearn.linear_model.Ridge(alpha=1.0, fit_intercept=True, solver="auto", normalize=False)

- 具有l2正则化的线性回归
- alpha:正则化力度，也叫 λ
  - 正则化力度越大，权重系数会越小
  - 正则化力度越小，权重系数会越大
  - *λ取值：0~1 1~10*
- solver:会根据数据自动选择优化方法
  - *sag:如果数据集、特征都比较大，选择该随机梯度下降优化*
- normalize:数据是否进行标准化
  - normalize=False:可以在fit之前调用preprocessing.StandardScaler标准化数据
- Ridge.coef_:回归权重
- Ridge.intercept_:回归偏置

交叉验证

sklearn.linear_model.RidgeCV(_BaseRidgeCV, RegressorMixin)

正则化

- 正则化力度越大，权重系数会越小
- 正则化力度越小，权重系数会越大

再论波士顿房价

```
# 使用带有L2正则化的线性回归去预测
rd = Ridge(alpha=1.0)
rd.fit(x_train, y_train)
print("岭回归计算出的权重：", rd.coef_)
print("岭回归计算出的偏置：", rd.intercept_)
```

过拟合与欠拟合

过拟合

定义

一个假设在训练数据上能够获得比其他假设更好的拟合，但是在测试数据集上却不能很好地拟合数据，此时认为这个假设出现了过拟合的现象。(模型过于复杂)

解决方法

正则化（尽量减少高次项特征的影响）

欠拟合

定义

一个假设在训练数据上不能获得更好的拟合，并且在测试数据集上也不能很好地拟合数据，此时认为这个假设出现了欠拟合的现象。(模型过于简单)

解决方法

增加数据的特征数量

正则化类别

L2正则化

- 作用：可以使得其中一些W的都很小，都接近于0（不为0），削弱某个特征的影响
- 优点：越小的参数说明模型越简单，越简单的模型则越不容易产生过拟合现象
- Ridge回归

L1正则化

- 作用：可以使得其中一些W的值直接为0，删除这个特征的影响
- LASSO回归

原理

线性回归的损失函数用最小二乘法，等价于当预测值与真实值的误差满足正态分布时的极大似然估计；岭回归的损失函数，是最小二乘法+L2范数，等价于当预测值与真实值的误差满足正态分布，且权重值也满足拉普拉斯分布（先验分布）时的最大后验估计；LASSO的损失函数，是最小二乘法+L1范数，等价于等价于当预测值与真实值的误差满足正态分布，且且权重值满足拉普拉斯分布（先验分布）时的最大后验估计。

总结

- 解决过拟合：决策树---剪枝、线性回归---正则化
- 都有特征选择功能（嵌入式）