SKLearn算法

`sklearn` `LR回归` `随机森林` `GBDT`

逻辑回归算法

Sklearn中与逻辑回归有关的3个类: LogisticRegression、LogisticRegressionCV和logistic_regression_path。LogisticRegressionCV使用了交叉验证来选择正则化系数C。LogisticRegression需要自己每次指定一个正则化系数。logistic_regression_path拟合数据后不能进行预测，只能为拟合数据选择合适逻辑回归的系数和正则化系数，主要用在模型选择时。

参数

- 正则化选择参数: penalty

> penalty可选择的值为l1和l2，分别对应L1正则化和L2正则化。默认为L2正则化。
> penalty参数会影响损失函数优化算法的选择，即solver参数，如果是L2正则化则有newton-cg，lbfgs，liblinear，sag四种可以选择。如果为L1正则化则只能选择liblinear，因为L1正则化的损失函数不是连续可导的，newton-cg、lbfgs、sag三种优化算法需要损失函数一阶或者二阶可导。

- 优化算法选择参数: solver

> liblinear: 使用开源liblinear库实现，内部使用坐标轴下降法迭代优化损失函数
> lbfgs: 拟牛顿法的一种，利用损失函数二阶导数矩阵即海森矩阵迭代优化损失函数。
> newton-cg: 利用损失函数二阶导数矩阵即海森矩阵迭代优化损失函数
> sag: 随机平均梯度下降，是梯度下降法的变种，与普通梯度下降法的区别是每次迭代仅仅使用一部分样本计算梯度，参考文献: https://blog.csdn.net/sun_shengyun/article/details/53811882

- 分类方式选择参数: multi_class

> 有ovr和multinomial两个值可选择，默认为ovr。如果是二元逻辑回归，两者并无区别，区别主要在多元逻辑回归上。

> ovr(one-vs-rest) 将所有的多元回归以二元逻辑回归进行。对于第k类分类决策，把所有第k类样本看做正例，其他样本视作负例，进行训练。
> MvM(many-vs-many)，对第k类要将其他的N-1个分类数据组成正负例进行训练，所以，总共要进行N(N-1)次分类。

- 类型权重参数: class_weight

> 用于标示分类模型中各种类型的权重，如果不输入则不考虑权重。参数选项包括: balanced让类库自己计算类型权重，也可以自己输入各个类型的权重。
> 选择balanced类库会根据训练样本量计算权重，某类型样本量越多，则权重越低，样本量越少则权重越高

- 样本权重参数: sample_weight
- dual参数

> 对偶或者原始方法。dual只适用于正则化项为l2且solver为liblinear的情况，通常样本数大于特征数的情况下，该参数为False。

- C

> C为正则化系数λ的倒数，通常默认为1.

- fit_intercept

> 是否存在截距，默认存在

- intercept_scaling

> 仅在正则化项为liblinear，且fit_intercept设置为True时有用。

- max_iter

> 仅在正则优化算法为newton-cg，sag和lbfgs时才有用，算法收敛的最大迭代次数。

- random_state

> 随机种子数，默认为无，仅在正正则化优化算法为sag，liblinear时有用。

- tol

> 迭代终止判断的误差范围，默认为10的-4次方。

- verbose

> 日志冗长度: 0: 不输出训练过程，1: 偶尔输出 >1 : 对每个子模型都输出

- warm_start

> 是否热启动，如果是，则下一次训练时以追加的形式进行，默认为False

- n_jobs

> 并行数，-1:表示跟CPU核数一致 1: 默认值

衍生知识

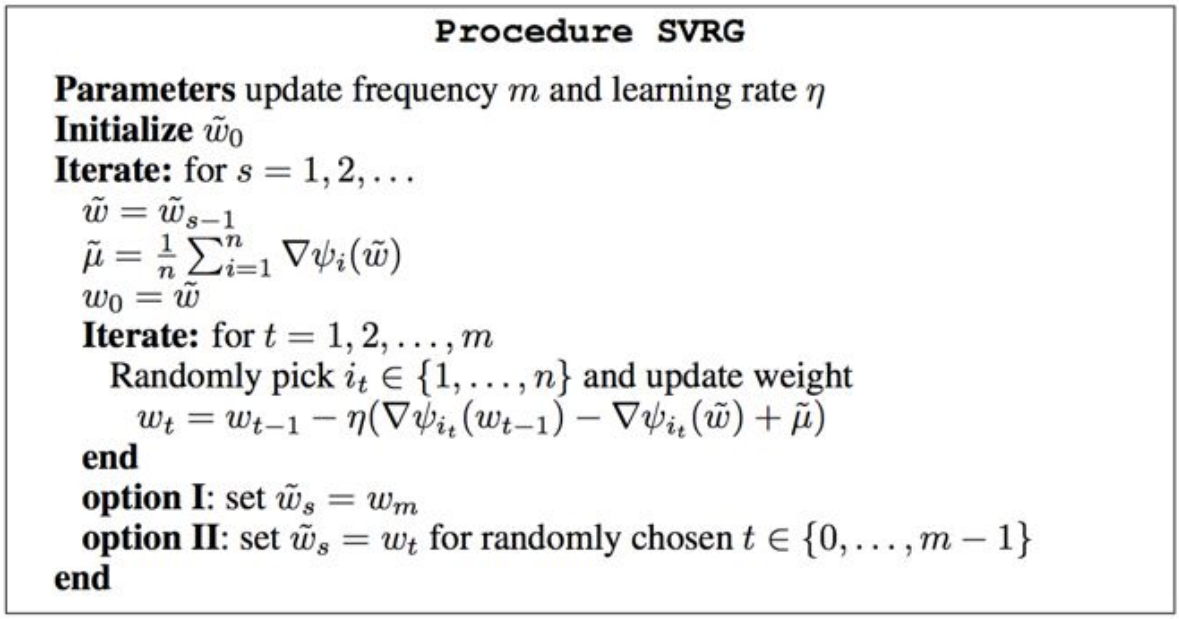
SAG算法

SAG算法在内存中为每个样本维护一个旧的梯度$\boldsymbol{y_i}$，随机选择一个样本$\boldsymbol{i}$更新用于进行梯度下降的梯度值$\boldsymbol{d}$，并用该梯度值$\boldsymbol{d}$对参数进行更新。更新的项$\boldsymbol{d}$来自于新的梯度$\boldsymbol{f_i^{'}(x)}$替换掉$\boldsymbol{d}$中的旧梯度$\boldsymbol{y_i}$。这样每次更新时只选取一个样本计算梯度，其计算开销与SGD基本一致，内存开销较大。

Algorithm 1 Basic SAG method for minimizing $\frac{1}{n}\sum_{i=1}^{n} f_i(x)$ with step size α.

$d = 0$, $y_i = 0$ for $i = 1, 2, \ldots, n$
for $k = 0, 1, \ldots$ **do**
  Sample i from $\{1, 2, \ldots, n\}$
  $d = d - y_i + f_i'(x)$
  $y_i = f_i'(x)$
  $x = x - \frac{\alpha}{n} d$
end for

SVRG算法



Procedure SVRG

Parameters update frequency m and learning rate η
Initialize $\tilde{w}_0$
Iterate: for $s = 1, 2, \ldots$
$\tilde{w} = \tilde{w}_{s-1}$
$\tilde{\mu} = \frac{1}{n}\sum_{i=1}^{n} \nabla\psi_i(\tilde{w})$
$w_0 = \tilde{w}$
Iterate: for $t = 1, 2, \ldots, m$
Randomly pick $i_t \in \{1, \ldots, n\}$ and update weight
$w_t = w_{t-1} - \eta(\nabla\psi_{i_t}(w_{t-1}) - \nabla\psi_{i_t}(\tilde{w}) + \tilde{\mu})$
end
option I: set $\tilde{w}_s = w_m$
option II: set $\tilde{w}_s = w_t$ for randomly chosen $t \in \{0, \ldots, m-1\}$
end

https://blog.csdn.net/sun_shengyun/article/details/53811882

决策树算法

sklearn内部的决策树算法使用了调优过的CART树算法，既可以做分类又可以做回归，分类决策树对应的是DecisionTreeClassifier，回归决策树对应的是DecisionTreeRegressor。

DecisionTreeClassifier既可以用于二分类问题，也可以用于多分类问题。

参数

- criterion

> 表示基于特征划分数据集合时，选择特征的标准，默认为gini，即Gini impurity(基尼不纯度)，其他选项还有entropy，表示通过信息增益进行划分

- spliter

> 表示在构造树时，选择节点的原则，默认为spliter=best，即选择最好的特征点分类。另外一个选择为random。

- max_features

> 表示划分数据集时考虑的最多的特征值数量，数据类型不同意义不同: int值->表示每次split时最大特征数；float->表示百分数。

- max_depth

> 表示树的最大深度

- min_samples_split

> 表示在分解内部结点时最少的样本数

- min_samples_leaf

> 表示每个叶节点最小的样本数目

- min_weight_fraction_leaf

> 这个值限制叶子节点所有样本权重和的最小值，如果小于这个值，则会和兄弟结点一起剪枝，默认为0，表示不考虑权重。

- max_leaf_nodes

> 限制最大叶子节点数，主要为防止过拟合。

- class_weight

> 指定各类别权重，默认值为None。

- min_inpurity_split

> 如果某个节点的不纯度(基尼系数、信息增益、均方差、绝对差)小于该值，则该节点不再生成子节点。

- presort

> 决定是否进行预排序。