调整估计器的超参数

原文链接：http://scikit-learn.org/stable/modules/grid_search.html

译文链接：http://cwiki.apachecn.org/pages/viewpage.action?pageId=10813989

贡献者：片刻 ApacheCN Apache中文网

超参数是在估计器内不直接学习的参数。在scikit学习中，他们被作为参数传递给估计器类的构造函数。典型的例子包括C，`kernel`与`gamma`用于支持向量分类，`alpha`为套索等

可以并建议搜索超参数空间以获得最佳的交叉验证：评估估计器性能得分。

可以以这种方式优化构造估计器时提供的任何参数。具体来说，要找到给定估计器的所有参数的名称和当前值，请使用：

```
estimator.get_params()
```

搜索包括：

- 估计器（回归器或分类器`sklearn.svm.SVC()`）;
- 参数空间;
- 一种用于搜索或抽样候选人的方法;
- 交叉验证方案; 和
- 一个得分的功能。

一些模型允许专门，高效的参数搜索策略， 如下所述。在scikit-learn中提供了两种采样搜索候选的通用方法：对于给定值，`GridSearchCV`穷尽地考虑所有参数组合，同时`RandomizedSearchCV`可以从具有指定分布的参数空间中抽取给定数量的候选。在描述这些工具之后，我们将详细介绍 两种方法的最佳实践。

请注意，这些参数的一小部分通常可能会对模型的预测或计算性能产生较大的影响，而其他参数可以保留为默认值。建议阅读估计类的文档，以便更好地了解其预期行为，可能通过阅读文献参考。

穷尽网格搜索

通过`GridSearchCV`穷举地提供的网格搜索从参数指定的参数值的网格生成候选者`param_grid`。例如，以下内容`param_grid`：

```
param_grid = [
  {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
  {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel':
['rbf']},
 ]
```

应该探讨两个网格：一个线性内核和C值在[1,10,100,1000]中，第二个是RBF内核，C值的交叉乘积在[1,10，100,1000]和[0.001，0.0001]中的伽马值。

该`GridSearchCV`实例实现了通用的估计器API：当在数据集上“拟合”时，将评估参数值的所有可能组合，并保留最佳组合。
例子：

- 有关数字数据集上的网格搜索计算示例，请参见使用具有交叉验证的网格搜索的参数估计。
- 参见文本特征提取和文本特征提取和评估的样本管道，用于从文本文档特征提取器（n-gram计数向量化器和TF-IDF变换器）与分类器的网格搜索耦合参数示例（这里是使用具有弹性网格的SGD训练的线性SVM或L2惩罚）`pipeline.Pipeline`。
- 在iris数据集的交叉验证循环中，参见网格搜索示例的嵌套交叉验证和非嵌套交叉验证。这是评估具有网格搜索的模型的性能的最佳做法。

随机参数优化

当使用参数设置网格时，当前使用最广泛的参数优化方法，其他搜索方法具有更有利的特性。 RandomizedSearchCV通过参数实现随机搜索，其中每个设置从可能的参数值的分布中进行采样。这与穷举搜索有两个主要好处：

- 可以独立于参数数量和可能的值来选择预算。
- 添加不影响性能的参数不会降低效率。

使用字典指定参数如何采样，与指定参数非常相似GridSearchCV。另外，使用该n_iter参数来指定作为采样候选的数量或采样迭代的计算预算。对于每个参数，可以指定在可能值上的分布或离散选择（将被均匀采样）的列表：

```
{'C': scipy.stats.expon(scale=100), 'gamma':
scipy.stats.expon(scale=.1),
  'kernel': ['rbf'], 'class_weight':['balanced', None]}
```

此示例使用scipy.stats模块，该模块包含了采样参数许多有用的分布，例如expon，gamma， uniform或randint。原则上，可以传递提供rvs（随机变量样本）方法来获取值的任何函数。对rvs函数的调用应该在连续调用时提供可能参数值的独立随机抽样。

> 警告:
>
> 在*scipy.stats*版本scipy 0.16之前的分布不允许指定随机状态。相反，他们使用全局numpy随机状态，可以通过*np.random.seed*或设置使用种子*np.random.set_state*。然而，开始scikit-learn 0.18，*sklearn.model_selection*如果scipy> = 0.16也可用，模块将设置由用户提供的随机状态。

对于连续参数，C如上所述，重要的是指定连续分布以充分利用随机化。这样，增加n_iter总是会导致更好的搜索。
例子：

- 比较随机搜索和网格搜索超参数估计比较随机搜索和网格搜索的使用和效率。

参考文献：

- Bergstra，J.and Bengio，Y.，Random search for hyper-parameter optimization，The Journal of Machine Learning Research（2012）

参数搜索提示

指定客观量度

默认情况下，参数搜索使用score估计器的功能来评估参数设置。这些是 sklearn.metrics.accuracy_score分类和sklearn.metrics.r2_score回归。对于某些应用，其他评分功能更适合（例如在不平衡分类中，准确度得分常常不知情）。一种替代打分函数可通过指定scoring参数来 GridSearchCV，RandomizedSearchCV和许多下面描述的专门的交叉验证的工具。请参阅评分参数：定义模型评估规则以获取更多详细信息。

综合估计和参数空间

管道：链接估计器描述了可以使用这些工具搜索参数空间的构建复合估计器。

模型选择：开发和评估

通过评估各种参数设置的模型选择可以看作是使用标记数据“训练”网格参数的一种方式。
在评估所产生的模型，可以做到这一点上，在电网搜索过程中均未见持有出来的样品是很重要的：建议将数据分割成一个集开发（被送入GridSearchCV实例）和评价设置 到计算性能指标。

这可以通过使用train_test_split 效用函数来完成。

并行

GridSearchCV并RandomizedSearchCV独立评估每个参数设置。如果您的操作系统支持它，可以使用关键字并行运行计算n_jobs=-1。有关详细信息，请参阅功能签名。

坚强的失败

某些参数设置可能导致fit数据的一个或多个折叠失败。默认情况下，这将导致整个搜索失败，即使某些参数设置可以被完全评估。设置erro

r_score=0（或—= np.NaN）将使程序对于此类故障的稳健性，发出警告并将该折叠的分数设置为0（或—NaN），但完成搜索。

暴力参数搜索的替代方法

模型特定交叉验证

一些模型可以适应一些参数值范围的数据，几乎与为参数的单个值拟合估计器一样高。可以利用此功能来执行用于此参数的模型选择的更有效的交叉验证。

适用于此策略的最常用参数是对正则化程序的强度进行编码的参数。在这种情况下，我们说我们计算估计器的正则化路径。

以下是这些型号的列表：

linear_model.ElasticNetCV([l1_ratio, eps, ...])	弹性网模型沿正则化路径迭代拟合
linear_model.LarsCV([fit_intercept, ...])	交叉验证的最小二乘回归模型
linear_model.LassoCV([eps, n_alphas, ...])	拉索线性模型，沿正则化路径迭代拟合
linear_model.LassoLarsCV([fit_intercept, ...])	使用LARS算法进行交叉验证的Lasso
linear_model.LogisticRegressionCV([Cs, ...])	Logistic回归CV（又名logit，MaxEnt）分类器。
linear_model.MultiTaskElasticNetCV([...])	多任务L1 / L2 ElasticNet内置交叉验证。
linear_model.MultiTaskLassoCV([eps, ...])	多任务L1 / L2 Lasso内置交叉验证。
linear_model.OrthogonalMatchingPursuitCV([...])	交叉验证的正交匹配追踪模型（OMP）
linear_model.RidgeCV([alphas, ...])	里奇回归与内置交叉验证。
linear_model.RidgeClassifierCV([alphas, ...])	里奇分类器内置交叉验证。

信息标准

一些模型可以通过计算单个正则化路径（而不是几个使用交叉验证）来提供正则化参数的最优估计的信息理论闭包公式。

以下是从Aikike信息标准（AIC）或贝叶斯信息标准（BIC）受益的型号列表，用于自动选择模型：

linear_model.LassoLarsIC([criterion, ...])	Lasso模型适合Lars使用BIC或AIC进行型号选择

超出袋估计

当使用基于装袋的集合方法时，即使用具有替换的采样生成新的训练集，部分训练集保持不用。对于集合中的每个分类器，训练集的不同部分被忽略。

这个省略的部分可以用来估计泛化误差，而不必依赖于单独的验证集。这个估计是“免费的”，因为不需要额外的数据，可以用于模型选择。

这是目前在以下类中实现的：

ensemble.RandomForestClassifier([...])	随机森林分类器。
ensemble.RandomForestRegressor([...])	随机森林回归。
ensemble.ExtraTreesClassifier([...])	一个额外的树分类器。
ensemble.ExtraTreesRegressor([n_estimators, ...])	一个额外的树木回归。
ensemble.GradientBoostingClassifier([loss, ...])	梯度提升分类。
ensemble.GradientBoostingRegressor([loss, ...])	渐变提升回归。