# 一、过滤式特征选取

## 1.1 VarianceThreshold

1、VarianceThreshold用于剔除方差很小的特征，原型为sklearn.feature_selection.VarianceThreshold(threshold=0.0)

- threshold：一个浮点数，指定方差的阈值。低于此阈值的特征将被剔除。

2、属性有variances_

- variances_：一个数组，元素分别是各特征的方差。

3、方法有fit、transform、fit_transform、get_support、inverse_transform。

- fit(X[, y])：从样本数据中学习每个特征的方差。
- transform(X)：执行特征选择，删除低于指定阈值的特征。
- fit_transform(X[, y])：从样本数据中学习每个特征的方差并删除低于指定阈值的特征。
- get_support([indices])：返回保留的特征。indices=True返回被选出的特征的索引；indices=False返回一个布尔值组成的数组。
- inverse_transform(X)：根据被选出来的特征还原原始数据（特征选取的逆操作），但是对于被删除的特征的值全部用0代替。

## 1.2 SelectKBest

1、SelectKBest用于保留统计得分最高的k个特征，原型为sklearn.feature_selection.SelectKest(score_func=<function f_classif>, k=10)

- score_func：一个函数，用于给出统计指标。该函数参数为(X, y)，返回值为(scores, pvalues)。X为样本集合；y为标签集合；scores为样本的得分集合，与X的每一行相对应；pvalues为样本得分的p值。
- k：一个整数或者字符串'all'，指定要保留的最佳的几个特征。若为'all'，保留所有特征。

2、sklearn中提供的常用统计指标函数score_func有f_regression、chi2、f_classif。

- sklearn.feature_selection.f_regression：基于线性回归分析来计算统计指标，适用于回归问题。
- sklearn.feature_selection.chi2：计算卡方统计量，适用于分类问题。
- sklearn.feature_selection.f_classif：根据方差分析Analysis of variance：ANOVA的原理，依靠F-分布为机率分布的依据，利用方法和与自由度所计算的组间与组内均方估计出F值。适用于分类问题。

3、属性有scores_、pvalues_

- scores_：一个数组，给出了所有特征的得分。
- pvalues_：一个数组，给出了所有特征得分的p-values。

4、方法参考VarianceThreshold

## 1.3 SelectPercentile

1、SelectPercentile用于保留统计得分最高的k%比例的特征，原型为sklearn.feature_selection.SelectPercentile(score_func=<function f_classif>, percentile=10)

- score_func：一个函数，用于给出统计指标。参考SelectKBest。
- percentile：一个整数，指定要保留最佳的百分之几的特征。

2、属性参考SelectKBest

3、方法参考SelectKBest

# 二、包裹式特征选取

## 2.1 RFE

1、RFE类用于实现包裹式特征选取，原型为sklearn.feature_selection.RFE(estimator, n_features_to_select=None, step=1, verbose=0)

- estimator：一个学习器，它必须提供一个.fit方法和一个.coef_特征。其中.coef_特征中存放的是学习到的各特征的权重系数。通常用SVM和广义线性模型作为estimator参数。
- n_feature_to_select：一个整数或None，指定要选出几个特征。如果未None，则默认选取一般的特征。
- step：一个整数或浮点数，指定每次迭代要剔除权重最小的几个特征。若为大于等于1的整数，则指定每次迭代要剔除特征的数量；若为0.0~1.0之间的浮点数，则指定每次迭代要剔除特征的比例。
- verbose：一个整数，控制输出日志。

2、RFE要求学习器能够学习特征的权重（如线性模型）的原理

- 首先学习器在初始特征集合上训练。
- 然后学习器学得每个特征的权重，剔除当前权重的一批特征，构成新的训练集。
- 再将学习器在新的训练集上训练，直至剩下的特征的数量满足条件。

3、属性有n_features_、support_、ranking_、estimator_

- n_features_：一个整数，给出了被选出的特征的数量。
- support_：一个数组，给出了特征是否被选择的mask。
- ranking_：特征权重排名，原始第i个特征的排名为ranking_[i]。
- estimator_：外部提供的学习器。

4、方法有fit、transform、fit_transform、get_support、inverse_transform等：

- fit(X[, y])：训练RFE模型。
- transform(X)：执行特征选择。
- fit_transform(X[, y])：从样本数据中学习RFE模型，然后执行特征选择。
- get_support([indices])：返回保留的特征。indices=True返回被选出的特征的索引；indices=False返回一个布尔值组成的数组。
- inverse_transform(X)：根据被选出来的特征还原原始数据（特征选取的逆操作），但是对于被删除的特征的值全部用0代替。
- predict(X)/predict_log_proba(X)/predict_proba(X)：将X进行特征选择之后，再使用内部的estimator来预测。

- score(X, y)：将X进行特征选择之后，训练内部estimator并对内部的estimator进行评分。

### 2.2 RFECV

1、RFECV是RFE的一个变体，执行一个交叉验证来寻找最优的剩余特征向量，因此不需要指定保留多少个特征，原型为sklearn.feature_selection.RFECV(estimator, step=1, cv=None, scoring=None, verbose=0)

- cv：一个整数或者交叉验证生成器或者一个可迭代对象，它决定了交叉验证策略。若为None，则使用默认的3折交叉验证；若为整数k，则使用k折交叉验证；若为交叉验证生成器，则直接使用该对象；若为可迭代对象，则使用该可迭代对象迭代生成训练-测试集合。
- 其他参数参考RFE。

2、其他属性参考RFE

- grid_scores_：一个数组，给出了交叉验证的预测性能得分。其元素为每个特征子集上执行交叉验证后的预测得分。

3、方法参考RFE

## 三、嵌入式特征选择

### 3.1 SelectFromModel

1、SelectFromModel用于实现嵌入式特征选取，原型为sklearn.feature_selection.SelectFromModel(estimator, threshold=None, prefit=False)

- estimator：一个学习器，它可以是未训练的(prefit=False)，也可以是已经训练好的(prefit=True)。estimator必须有coef_或feature_importances_属性，给出每个特征的重要性。当某个特征的重要性低于某个阈值时，该特征将被移除。
- threshold：一个字符串或者浮点数或者None，指定特征重要性的一个阈值。低于此阈值的特征将被剔除。若为字符串，可以是'mean'（阈值为特征重要性的均值）、'median'（阈值为特征重要性的中值），如果是'1.5*mean'则表示阈值为1.5倍的特征重要性的均值；若为浮点数则指定阈值的绝对大小；若为None阈值默认为'mean'，当estimaor有一个penalty参数且设置为'l1'时阈值默认为1e-5。
- prefit：一个布尔值，指定estimator是否已经训练好了。默认为False未训练。

2、属性有threshold_

- threshold_：一个浮点数，存储了用于特征选取重要性的阈值。

3、方法有fit、transform、fit_transform、get_support、inverse_transform、partial_fit

- fit(X[, y])：训练SelectFromModel模型。
- transform(X)：执行特征选择。
- fit_transform(X[, y])：从样本数据中学习SelectFromModel模型，然后执行特征选择。
- get_support([indices])：返回保留的特征。indices=True返回被选出的特征的索引；indices=False返回一个布尔值组成的数组。
- inverse_transform(X)：根据被选出来的特征还原原始数据（特征选取的逆操作），但是对于被删除的特征的值全部用0代替。

- partial_fit(X[, y])：通过部分数据来学习SelectFromModel模型。它支持批量学习，这样对于内存更友好。即训练数据并不是一次性学习，而是分批学习。