官方文档参数：

# 分类

## 1. 逻辑回归

**参数：**

elasticNetParam：

类型：双精度型。

含义：弹性网络混合参数，范围[0,1]。

featuresCol:

类型：字符串型。

含义：特征列名。

fitIntercept:

类型：布尔型。

含义：是否训练拦截对象。

labelCol:

类型：字符串型。

含义：标签列名。

maxIter:

类型：整数型。

含义：最多迭代次数（>=0）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

probabilityCol:

类型：字符串型。

含义：用以预测类别条件概率的列名。

regParam:

类型：双精度型。

含义：正则化参数（>=0）。

standardization:

类型：布尔型。

含义：训练模型前是否需要对训练特征进行标准化处理。

threshold:

类型：双精度型。

含义：二分类预测的阀值，范围[0,1]。

thresholds:

类型：双精度数组型。

含义：多分类预测的阀值，以调整预测结果在各个类别的概率。

tol:

类型：双精度型。

含义：迭代算法的收敛性。

weightCol:

类型：字符串型。

含义：列权重。

## 2. 决策树

**算法介绍：**

  决策树以及其集成算法是机器学习分类和回归问题中非常流行的算法。因其易解释性、可处理类别特征、易扩展到多分类问题、不需特征缩放等性质被广泛使用。树集成算法如随机森林以及boosting算法几乎是解决分类和回归问题中表现最优的算法。

  决策树是一个贪心算法递归地将特征空间划分为两个部分，在同一个叶子节点的数据最后会拥有同样的标签。每次划分通过贪心的以获得最大信息增益为目的，从可选择的分裂方式中选择最佳的分裂节点。节点不纯度有节点所含类别的同质性来衡量。工具提供为分类提供两种不纯度衡量（基尼不纯度和熵），为回归提供一种不纯度衡量（方差）。

  spark.ml支持二分类、多分类以及回归的决策树算法，适用于连续特征以及类别特征。另外，对于分类问题，工具可以返回属于每种类别的概率（类别条件概率），对于回归问题工具可以返回预测在偏置样本上的方差。

**参数：**

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

featuresCol:

类型：字符串型。

含义：特征列名。

impurity:

类型：字符串型。

含义：计算信息增益的准则（不区分大小写）。

labelCol:

类型：字符串型。

含义：标签列名。

maxBins:

类型：整数型。

含义：连续特征离散化的最大数量，以及选择每个节点分裂特征的方式。

maxDepth:

类型：整数型。

含义：树的最大深度（>=0）。

minInfoGain:

类型：双精度型。

含义：分裂节点时所需最小信息增益。

minInstancesPerNode:

类型：整数型。

含义：分裂后自节点最少包含的实例数量。

predictionCol:

类型：字符串型。

含义：预测结果列名。

probabilityCol:

类型：字符串型。

含义：类别条件概率预测结果列名。

rawPredictionCol:

类型：字符串型。

含义：原始预测。

seed:

类型：长整型。

含义：随机种子。

thresholds:

类型：双精度数组型。

含义：多分类预测的阀值，以调整预测结果在各个类别的概率。

## 3. 随机森林（Random Forest）

**随机森林分类器：**

**算法简介：**

  随机森林是决策树的集成算法。随机森林包含多个决策树来降低过拟合的风险。随机森林同样具有易解释性、可处理类别特征、易扩展到多分类问题、不需特征缩放等性质。

  随机森林分别训练一系列的决策树，所以训练过程是并行的。因算法中加入随机过程，所以每个决策树又有少量区别。通过合并每个树的预测结果来减少预测的方差，提高在测试集上的性能表现。

随机性体现：  
1.每次迭代时，对原始数据进行二次抽样来获得不同的训练数据。

2.对于每个树节点，考虑不同的随机特征子集来进行分裂。

除此之外，决策时的训练过程和单独决策树训练过程相同。

   对新实例进行预测时，随机森林需要整合其各个决策树的预测结果。回归和分类问题的整合的方式略有不同。分类问题采取投票制，每个决策树投票给一个类别，获得最多投票的类别为最终结果。回归问题每个树得到的预测结果为实数，最终的预测结果为各个树预测结果的平均值。

   spark.ml支持二分类、多分类以及回归的随机森林算法，适用于连续特征以及类别特征。

**参数：**

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

featureSubsetStrategy:

类型：字符串型。

含义：每次分裂候选特征数量。

featuresCol:

类型：字符串型。

含义：特征列名。

impurity:

类型：字符串型。

含义：计算信息增益的准则（不区分大小写）。

labelCol:

类型：字符串型。

含义：标签列名。

maxBins:

类型：整数型。

含义：连续特征离散化的最大数量，以及选择每个节点分裂特征的方式。

maxDepth:

类型：整数型。

含义：树的最大深度（>=0）。

minInfoGain:

类型：双精度型。

含义：分裂节点时所需最小信息增益。

minInstancesPerNode:

类型：整数型。

含义：分裂后自节点最少包含的实例数量。

numTrees:

类型：整数型。

含义：训练的树的数量。

predictionCol:

类型：字符串型。

含义：预测结果列名。

probabilityCol:

类型：字符串型。

含义：类别条件概率预测结果列名。

rawPredictionCol:

类型：字符串型。

含义：原始预测。

seed:

类型：长整型。

含义：随机种子。

subsamplingRate:

类型：双精度型。

含义：学习一棵决策树使用的训练数据比例，范围[0,1]。

thresholds:

类型：双精度数组型。

含义：多分类预测的阀值，以调整预测结果在各个类别的概率。

## 4. 梯度迭代树（GBDT）算法

**算法简介：**

  梯度提升树是一种决策树的集成算法。它通过反复迭代训练决策树来最小化损失函数。决策树类似，梯度提升树具有可处理类别特征、易扩展到多分类问题、不需特征缩放等性质。Spark.ml通过使用现有[decision tree](http://spark.apache.org/docs/latest/mllib-decision-tree.html)工具来实现。

   梯度提升树依次迭代训练一系列的决策树。在一次迭代中，算法使用现有的集成来对每个训练实例的类别进行预测，然后将预测结果与真实的标签值进行比较。通过重新标记，来赋予预测结果不好的实例更高的权重。所以，在下次迭代中，决策树会对先前的错误进行修正。

   对实例标签进行重新标记的机制由损失函数来指定。每次迭代过程中，梯度迭代树在训练数据上进一步减少损失函数的值。spark.ml为分类问题提供一种损失函数（Log Loss），为回归问题提供两种损失函数（平方误差与绝对误差）。

   Spark.ml支持二分类以及回归的随机森林算法，适用于连续特征以及类别特征。

＊注意梯度提升树目前不支持多分类问题。

**参数：**

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

featuresCol:

类型：字符串型。

含义：特征列名。

impurity:

类型：字符串型。

含义：计算信息增益的准则（不区分大小写）。

labelCol:

类型：字符串型。

含义：标签列名。

lossType:

类型：字符串型。

含义：损失函数类型。

maxBins:

类型：整数型。

含义：连续特征离散化的最大数量，以及选择每个节点分裂特征的方式。

maxDepth:

类型：整数型。

含义：树的最大深度（>=0）。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

minInfoGain:

类型：双精度型。

含义：分裂节点时所需最小信息增益。

minInstancesPerNode:

类型：整数型。

含义：分裂后自节点最少包含的实例数量。

predictionCol:

类型：字符串型。

含义：预测结果列名。

rawPredictionCol:

类型：字符串型。

含义：原始预测。

seed:

类型：长整型。

含义：随机种子。

subsamplingRate:

类型：双精度型。

含义：学习一棵决策树使用的训练数据比例，范围[0,1]。

stepSize:

类型：双精度型。

含义：每次迭代优化步长。

## 5. 多层感知机（MLP）算法

**算法简介：**

  多层感知机是基于反向人工神经网络（[feedforwardartificial neural network](https://en.wikipedia.org/wiki/Feedforward_neural_network)）。多层感知机含有多层节点，每层节点与网络的下一层节点完全连接。输入层的节点代表输入数据，其他层的节点通过将输入数据与层上节点的权重w以及偏差b线性组合且应用一个激活函数，得到该层输出。多层感知机通过方向传播来学习模型，其中我们使用逻辑损失函数以及L-BFGS。

输出层中N代表类别数目。

**参数：**

featuresCol:

类型：字符串型。

含义：特征列名。

labelCol:

类型：字符串型。

含义：标签列名。

layers:

类型：整数数组型。

含义：层规模，包括输入规模以及输出规模。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

seed:

类型：长整型。

含义：随机种子。

stepSize:

类型：双精度型。

含义：每次迭代优化步长。

tol:

类型：双精度型。

含义：迭代算法的收敛性。

## 6. One-vs-Rest算法

**算法介绍：**

[OneVsRest](http://en.wikipedia.org/wiki/Multiclass_classification#One-vs.-rest)将一个给定的二分类算法有效地扩展到多分类问题应用中，也叫做“One-vs-All.”算法。OneVsRest是一个Estimator。它采用一个基础的Classifier然后对于k个类别分别创建二分类问题。类别i的二分类分类器用来预测类别为i还是不为i，即将i类和其他类别区分开来。最后，通过依次对k个二分类分类器进行评估，取置信最高的分类器的标签作为i类别的标签。

**参数：**

featuresCol:

类型：字符串型。

含义：特征列名。

labelCol:

类型：字符串型。

含义：标签列名。

predictionCol:

类型：字符串型。

含义：预测结果列名。

classifier:

类型：分类器型。

含义：基础二分类分类器

## 7. 朴素贝叶斯算法

**算法介绍：**

朴素贝叶斯法是基于贝叶斯定理与特征条件独立假设的分类方法。

朴素贝叶斯的思想基础是这样的：对于给出的待分类项，求解在此项出现的条件下各个类别出现的概率，在没有其它可用信息下，我们会选择条件概率最大的类别作为此待分类项应属的类别。

spark.ml现在支持多项朴素贝叶斯和伯努利朴素贝叶斯。

**参数：**

featuresCol:

类型：字符串型。

含义：特征列名。

labelCol:

类型：字符串型。

含义：标签列名。

modelType:

类型：字符串型。

含义：模型类型（区分大小写）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

probabilityCol:

类型：字符串型。

含义：用以预测类别条件概率的列名。

rawPredictionCol:

类型：字符串型。

含义：原始预测。

smoothing:

类型：双精度型。

含义：平滑参数。

thresholds:

类型：双精度数组型。

含义：多分类预测的阀值，以调整预测结果在各个类别的概率。

# 回归：

## 1. 广义线性回归

Spark的GeneralizedLinearRegression接口提供汇总统计来诊断GLM模型的拟合程度，包括残差、p值、残差、Akaike信息准则及其它。

**参数：**

family:

类型：字符串型。

含义：模型中使用的误差分布类型。

featuresCol:

类型：字符串型。

含义：特征列名。

fitIntercept:

类型：布尔型。

含义：是否训练拦截对象。

labelCol:

类型：字符串型。

含义：标签列名。

link:

类型：字符串型。

含义：连接函数名，描述线性预测器和分布函数均值之间关系。

linkPredictiongCol:

类型：字符串型。

含义：连接函数（线性预测器列名）。

maxIter:

类型：整数型。

含义：最多迭代次数（>=0）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

regParam:

类型：双精度型。

含义：正则化参数（>=0）。

solver:

类型：字符串型。

含义：优化的求解算法。

tol:

类型：双精度型。

含义：迭代算法的收敛性。

weightCol:

类型：字符串型。

含义：列权重。

## 2. 决策树回归算法原理

**算法介绍：**

        决策树以及其集成算法是机器学习分类和回归问题中非常流行的算法。因其易解释性、可处理类别特征、易扩展到多分类问题、不需特征缩放等性质被广泛使用。树集成算法如随机森林以及boosting算法几乎是解决分类和回归问题中表现最优的算法。

       决策树是一个贪心算法递归地将特征空间划分为两个部分，在同一个叶子节点的数据最后会拥有同样的标签。每次划分通过贪心的以获得最大信息增益为目的，从可选择的分裂方式中选择最佳的分裂节点。节点不纯度有节点所含类别的同质性来衡量。工具提供为分类提供两种不纯度衡量（基尼不纯度和熵），为回归提供一种不纯度衡量（方差）。

       spark.ml支持二分类、多分类以及回归的决策树算法，适用于连续特征以及类别特征。另外，对于分类问题，工具可以返回属于每种类别的概率（类别条件概率），对于回归问题工具可以返回预测在偏置样本上的方差。

**参数：**

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

featuresCol:

类型：字符串型。

含义：特征列名。

impurity:

类型：字符串型。

含义：计算信息增益的准则（不区分大小写）。

labelCol:

类型：字符串型。

含义：标签列名。

maxBins:

类型：整数型。

含义：连续特征离散化的最大数量，以及选择每个节点分裂特征的方式。

maxDepth:

类型：整数型。

含义：树的最大深度（>=0）。

minInfoGain:

类型：双精度型。

含义：分裂节点时所需最小信息增益。

minInstancesPerNode:

类型：整数型。

含义：分裂后自节点最少包含的实例数量。

predictionCol:

类型：字符串型。

含义：预测结果列名。

seed:

类型：长整型。

含义：随机种子。

varianceCol:

类型：字符串型。

含义：预测的有偏样本偏差的列名。

## 3. 随机森林回归（Random Forest）算法

**随机森林回归**

**算法介绍：**

       随机森林是决策树的集成算法。随机森林包含多个决策树来降低过拟合的风险。随机森林同样具有易解释性、可处理类别特征、易扩展到多分类问题、不需特征缩放等性质。

       随机森林分别训练一系列的决策树，所以训练过程是并行的。因算法中加入随机过程，所以每个决策树又有少量区别。通过合并每个树的预测结果来减少预测的方差，提高在测试集上的性能表现。

       随机性体现：  
1.每次迭代时，对原始数据进行二次抽样来获得不同的训练数据。

2.对于每个树节点，考虑不同的随机特征子集来进行分裂。

        除此之外，决策时的训练过程和单独决策树训练过程相同。

        对新实例进行预测时，随机森林需要整合其各个决策树的预测结果。回归和分类问题的整合的方式略有不同。分类问题采取投票制，每个决策树投票给一个类别，获得最多投票的类别为最终结果。回归问题每个树得到的预测结果为实数，最终的预测结果为各个树预测结果的平均值。

        spark.ml支持二分类、多分类以及回归的随机森林算法，适用于连续特征以及类别特征。

**参数：**

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

featureSubsetStrategy:

类型：字符串型。

含义：每次分裂候选特征数量。

featuresCol:

类型：字符串型。

含义：特征列名。

impurity:

类型：字符串型。

含义：计算信息增益的准则（不区分大小写）。

labelCol:

类型：字符串型。

含义：标签列名。

maxBins:

类型：整数型。

含义：连续特征离散化的最大数量，以及选择每个节点分裂特征的方式。

maxDepth:

类型：整数型。

含义：树的最大深度（>=0）。

minInfoGain:

类型：双精度型。

含义：分裂节点时所需最小信息增益。

minInstancesPerNode:

类型：整数型。

含义：分裂后自节点最少包含的实例数量。

numTrees:

类型：整数型。

含义：训练的树的数量。

predictionCol:

类型：字符串型。

含义：预测结果列名。

seed:

类型：长整型。

含义：随机种子。

subsamplingRate:

类型：双精度型。

含义：学习一棵决策树使用的训练数据比例，范围[0,1]。

thresholds:

类型：双精度数组型。

含义：多分类预测的阀值，以调整预测结果在各个类别的概率。

## 4. 梯度迭代树回归（GBDT）算法

**算法简介：**

        梯度提升树是一种决策树的集成算法。它通过反复迭代训练决策树来最小化损失函数。决策树类似，梯度提升树具有可处理类别特征、易扩展到多分类问题、不需特征缩放等性质。Spark.ml通过使用现有[decision tree](http://spark.apache.org/docs/latest/mllib-decision-tree.html)工具来实现。

       梯度提升树依次迭代训练一系列的决策树。在一次迭代中，算法使用现有的集成来对每个训练实例的类别进行预测，然后将预测结果与真实的标签值进行比较。通过重新标记，来赋予预测结果不好的实例更高的权重。所以，在下次迭代中，决策树会对先前的错误进行修正。

       对实例标签进行重新标记的机制由损失函数来指定。每次迭代过程中，梯度迭代树在训练数据上进一步减少损失函数的值。spark.ml为分类问题提供一种损失函数（Log Loss），为回归问题提供两种损失函数（平方误差与绝对误差）。

       Spark.ml支持二分类以及回归的随机森林算法，适用于连续特征以及类别特征。

＊注意梯度提升树目前不支持多分类问题。

**参数：**

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

featuresCol:

类型：字符串型。

含义：特征列名。

impurity:

类型：字符串型。

含义：计算信息增益的准则（不区分大小写）。

labelCol:

类型：字符串型。

含义：标签列名。

lossType:

类型：字符串型。

含义：损失函数类型。

maxBins:

类型：整数型。

含义：连续特征离散化的最大数量，以及选择每个节点分裂特征的方式。

maxDepth:

类型：整数型。

含义：树的最大深度（>=0）。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

minInfoGain:

类型：双精度型。

含义：分裂节点时所需最小信息增益。

minInstancesPerNode:

类型：整数型。

含义：分裂后自节点最少包含的实例数量。

predictionCol:

类型：字符串型。

含义：预测结果列名。

seed:

类型：长整型。

含义：随机种子。

subsamplingRate:

类型：双精度型。

含义：学习一棵决策树使用的训练数据比例，范围[0,1]。

stepSize:

类型：双精度型。

含义：每次迭代优化步长。

## 5. 生存回归(加速失效时间模型)算法

 import org.apache.spark.ml.linalg.Vectors

 import org.apache.spark.ml.regression.AFTSurvivalRegression

**算法介绍：**

 在spark.ml中，我们实施加速失效时间模型（[Acceleratedfailure time](https://en.wikipedia.org/wiki/Accelerated_failure_time_model)），对于截尾数据它是一个参数化生存回归的模型。它描述了一个有对数生存时间的模型，所以它也常被称为生存分析的对数线性模型。与比例危险模型不同，因AFT模型中每个实例对目标函数的贡献是独立的，其更容易并行化。

当使用无拦截的连续非零列训练AFTSurvivalRegressionModel时，Spark MLlib为连续非零列输出零系数。这种处理与R中的生存函数survreg不同。

**参数：**

censorCol:

类型：字符串型。

含义：检查器列名。

featuresCol:

类型：字符串型。

含义：特征列名。

fitIntercept:

类型：布尔型。

含义：是否训练拦截对象。

labelCol:

类型：字符串型。

含义：标签列名。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

quantileProbabilities:

类型：双精度数组型。

含义：分位数概率数组。

quantilesCol:

类型：字符串型。

含义：分位数列名。

stepSize:

类型：双精度型。

含义：每次迭代优化步长。

tol:

类型：双精度型。

含义：迭代算法的收敛性。

## 6. 保序回归算法

org.apache.spark.ml.regression.IsotonicRegression

 训练返回一个保序回归模型，可以被用于来预测已知或者未知特征值的标签。保序回归的结果是分段线性函数，预测规则如下：

1.如果预测输入与训练中的特征值完全匹配，则返回相应标签。如果一个特征值对应多个预测标签值，则返回其中一个，具体是哪一个未指定。

2.如果预测输入比训练中的特征值都高（或者都低），则相应返回最高特征值或者最低特征值对应标签。如果一个特征值对应多个预测标签值，则相应返回最高值或者最低值。

3.如果预测输入落入两个特征值之间，则预测将会是一个分段线性函数，其值由两个最近的特征值的预测值计算得到。如果一个特征值对应多个预测标签值，则使用上述两种情况中的处理方式解决。

**参数：**

featuresIndex:

类型：整数型。

含义：当特征列维向量时提供索引值，否则不进行处理。

featuresCol:

类型：字符串型。

含义：特征列名。

isotonic:

类型：布尔型。

含义：输出序列为保序/增序（真）或者反序/降序（假）。

labelCol:

类型：字符串型。

含义：标签列名。

predictionCol:

类型：字符串型。

含义：预测结果列名。

weightCol:

类型：字符串型。

含义：列权重。

# 聚类

## 1. K均值(K-means)算法

**算法介绍：**

K-means是一个常用的聚类算法来将数据点按预定的簇数进行聚集。K-means算法的基本思想是：以空间中k个点为中心进行聚类，对最靠近他们的对象归类。通过迭代的方法，逐次更新各聚类中心的值，直至得到最好的聚类结果。

假设要把样本集分为c个类别，算法描述如下：

（1）适当选择c个类的初始中心；

（2）在第k次迭代中，对任意一个样本，求其到c个中心的距离，将该样本归到距离最短的中心所在的类；

（3）利用均值等方法更新该类的中心值；

（4）对于所有的c个聚类中心，如果利用（2）（3）的迭代法更新后，值保持不变，则迭代结束，否则继续迭代。

MLlib工具包含并行的K-means++算法，称为kmeans||。Kmeans是一个Estimator，它在基础模型之上产生一个KMeansModel。

**参数：**

featuresCol:

类型：字符串型。

含义：特征列名。

k:

类型：整数型。

含义：聚类簇数。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

seed:

类型：长整型。

含义：随机种子。

tol:

类型：双精度型。

含义：迭代算法的收敛性。

## 2. 文档主题生成模型(LDA)算法

**算法介绍：**

LDA（Latent Dirichlet Allocation）是一种文档主题生成模型，也称为一个三层贝叶斯概率模型，包含词、主题和文档三层结构。所谓生成模型，就是说，我们认为一篇文章的每个词都是通过“以一定概率选择了某个主题，并从这个主题中以一定概率选择某个词语”这样一个过程得到。文档到主题服从多项式分布，主题到词服从多项式分布。

LDA是一种非监督机器学习技术，可以用来识别大规模文档集（document collection）或语料库（corpus）中潜藏的主题信息。它采用了词袋（bag of words）的方法，这种方法将每一篇文档视为一个词频向量，从而将文本信息转化为了易于建模的数字信息。但是词袋方法没有考虑词与词之间的顺序，这简化了问题的复杂性，同时也为模型的改进提供了契机。每一篇文档代表了一些主题所构成的一个概率分布，而每一个主题又代表了很多单词所构成的一个概率分布。

**参数：**

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

docConcentration:

类型：双精度数组型。

含义：文档关于主题（"theta"）的先验分布集中参数（通常名为“alpha"）。

featuresCol:

类型：字符串型。

含义：特征列名。

k:

类型：整数型。

含义：需推断的主题（簇）的数目。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

optimizer:

类型：字符串型。

含义：估计LDA模型时使用的优化器。

含义：类别条件概率预测结果列名。

seed:

类型：长整型。

含义：随机种子。

subsamplingRate:

类型：双精度型。

含义：仅对在线优化器（即optimizer=”online”）。

topicConcentration:

类型：双精度型。

含义：主题关于文字的先验分布集中参数（通常名为“beta"或"eta"）。

topicDistributionCol:

类型：字符串型。

含义：每个文档的混合主题分布估计的输出列（文献中通常名为"theta"）。

## 3. 二分K均值算法

import org.apache.spark.ml.clustering.BisectingKMeans

**算法介绍：**

二分K均值算法是一种层次聚类算法，使用自顶向下的逼近：所有的观察值开始是一个簇，递归地向下一个层级分裂。分裂依据为选择能最大程度降低聚类代价函数（也就是误差平方和）的簇划分为两个簇。以此进行下去，直到簇的数目等于用户给定的数目k为止。二分K均值常常比传统K均值算法有更快的计算速度，但产生的簇群与传统K均值算法往往也是不同的。

BisectingKMeans是一个Estimator，在基础模型上训练得到BisectingKMeansModel。

**参数：**

featuresCol:

类型：字符串型。

含义：特征列名。

k:

类型：整数型。

含义：聚类簇数。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

seed:

类型：长整型。

含义：随机种子。

## 4. 混合高斯模型(GMM)

import org.apache.spark.ml.clustering.GaussianMixture

**算法原理：**

    混合高斯模型描述数据点以一定的概率服从k种高斯子分布的一种混合分布。Spark.ml使用EM算法给出一组样本的极大似然模型。

**参数：**

featuresCol:

类型：字符串型。

含义：特征列名。

k:

类型：整数型。

含义：混合模型中独立的高斯数目。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

probabilityCol:

类型：字符串型。

含义：用以预测类别条件概率的列名。

seed:

类型：长整型。

含义：随机种子。

tol:

类型：双精度型。

含义：迭代算法的收敛性。

## 5. 协同过滤(ALS)算法

import org.apache.spark.ml.evaluation.RegressionEvaluator

import org.apache.spark.ml.recommendation.ALS

**算法介绍：**

    协同过滤常被用于推荐系统。这类技术目标在于填充“用户－商品”联系矩阵中的缺失项。Spark.ml目前支持基于模型的协同过滤，其中用户和商品以少量的潜在因子来描述，用以预测缺失项。Spark.ml使用交替最小二乘（ALS）算法来学习这些潜在因子。

＊注意基于DataFrame的ALS接口目前仅支持整数型的用户和商品编号。

**显式与隐式反馈**

    基于矩阵分解的协同过滤的标准方法中，“用户－商品”矩阵中的条目是用户给予商品的显式偏好，例如，用户给电影评级。然而在现实世界中使用时，我们常常只能访问隐式反馈（如意见、点击、购买、喜欢以及分享等），在spark.ml中我们使用“隐式反馈数据集的协同过滤“来处理这类数据。本质上来说它不是直接对评分矩阵进行建模，而是将数据当作数值来看待，这些数值代表用户行为的观察值（如点击次数，用户观看一部电影的持续时间）。这些数值被用来衡量用户偏好观察值的置信水平，而不是显式地给商品一个评分。然后，模型用来寻找可以用来预测用户对商品预期偏好的潜在因子。

**正则化参数**

    我们调整正则化参数regParam来解决用户在更新用户因子时产生新评分或者商品更新商品因子时收到的新评分带来的最小二乘问题。这个方法叫做“ALS-WR”它降低regParam对数据集规模的依赖，所以我们可以将从部分子集中学习到的最佳参数应用到整个数据集中时获得同样的性能。

**参数：**

alpha:

类型：双精度型。

含义：隐式偏好中的alpha参数（非负）。

checkpointInterval:

类型：整数型。

含义：设置检查点间隔（>=1），或不设置检查点（-1）。

implicitPrefs:

类型：布尔型。

含义：特征列名。

itemCol:

类型：字符串型。

含义：商品编号列名。

maxIter:

类型：整数型。

含义：迭代次数（>=0）。

nonnegative:

类型：布尔型。

含义：是否需要非负约束。

numItemBlocks:

类型：整数型。

含义：商品数目（正数）。

numUserBlocks:

类型：整数型。

含义：用户数目（正数）。

predictionCol:

类型：字符串型。

含义：预测结果列名。

rank:

类型：整数型。

含义：分解矩阵的排名（正数）。

ratingCol:

类型：字符串型。

含义：评分列名。

regParam:

类型：双精度型。

含义：正则化参数（>=0）。

seed:

类型：长整型。

含义：随机种子。

userCol:

类型：字符串型。

含义：用户列名。