链滴

Spark mllib API- classification

作者：Zing

原文链接：https://ld246.com/article/1460106426957

来源网站：链滴

Apark mllib API的翻译 - 分类篇。 对官方文档进行翻译的同时加入了一些常识性知识。

更多分类的相关知识可以查看我的另外一篇博客数据挖掘算法初窥门庭--分类回归

Spark当前提供LogisticRegression、SVM、NaiveBayes。

##LogisticRegression 逻辑回归

###背景知识

LinerRegression是使用线性方程对数据进行两分类（在线的一侧属于同一类）。而LogisticRegressin就是一个被logistic方程归一化后的LinerRegression（归一化后值域为0-1）。LogisticRegression般也用于两分类，预测样本属于某个类别的概率。

LogisticRegression的过程是典型的监督机器学习，也就是在规则化参数的同时最小化误差。最小化差是为了让我们的模型拟合我们的训练数据，而规则化参数是防止我们的模型过分拟合我们的训练数。

大致步骤如下：

- 目标函数为f（f为未知的），我们假定目标函数为h。（假设）
- 构造损失函数cost（基于最大似然估计），表示h的预测结果与实际结果f之间的偏差。（预测并评）
- 通过迭代，调整h，使h与f尽可能接近。（求最优解）

LogisticRegression有很多不同的算法版本，大多数的主要不同在于求最优解。目前，spark提供两种ogisticRegression方法：SGD（随机梯度下降）和LBFGS（改进的拟牛顿法）。

特征选择：

- LogisticRegression假设向量的各个维度是独立不相互影响的。
- 由于LogisticRegression的终止条件是收敛或达到最大迭代次数，因此在数据预处理时进行归一化加快收敛速度。
- 更多具体的变量选择方法，参考 华山大师兄的Logistic Regression--逻辑回归算法汇总

###Spark API

- 类：pyspark.mllib.classification.LogisticRegressionWithSGD
  - 方法：

train(data, iterations=100, step=1.0, miniBatchFraction=1.0, initialWeights=None, regParam=0.01, regType='l2', intercept=False, validateData=True, convergenceTol=0.001)

通过给定数据训练逻辑回归模型。

- data：训练数据，LabeledPoint格式的RDD数据集。
- iterations：迭代次数，默认为100。
- step：SGD的步长，默认为1.0。（太大容易错过最优解，太小导致迭代次数过多）。

- miniBatchFraction：用于每次SGD迭代的数据，默认1.0。（SGD每次迭代选用随机数据）。
    - initialWeights：初始权值，默认None。
    - regParam：规则化参数，默认0.01。
    - regType：用于训练模型的规则化类型，可选为l1或l2（默认）。
    - intercept：布尔值，表示是否使用增强表现来训练数据，默认False。
    - validateData：布尔值，表示算法是否在训练前检验数据，默认True。
    - convergenceTol：终止迭代的收敛值，默认0.001。

- 类： pyspark.mllib.classification.LogisticRegressionWithLBFGS
    - 方法：

train(data, iterations=100, initialWeights=None, regParam=0.01, regType='l2', intercept=False
 corrections=10, tolerance=0.0001, validateData=True, numClasses=2)

通过给定数据训练逻辑回归模型。

- data：训练数据，LabeledPoint格式的RDD数据集。
- iterations：迭代次数，默认为100。
- initialWeights：初始权值，默认None。
- regParam：规则化参数，默认0.01。
- regType：用于训练模型的规则化类型，可选为l1或l2（默认）。
- intercept：布尔值，表示是否使用增强表现来训练数据，默认False。
- corrections：用于LBFGS更新的修正值，默认10。
- tolerance：LBFGS迭代的收敛容忍系数，默认1e-4。
- validateData：布尔值，表示算法是否在训练前检验数据，默认True。
- numClasses：多分类逻辑回归中类别的个数，默认2。

- 类：pyspark.mllib.classification.LogisticRegressionModel

使用多/两逻辑分类方法训练得到的模型。

- 属性：
    - weights：每个向量计算的权值。
    - intercept：该模型的计算截距（只用于两逻辑回归）。
    - numFeatures：向量的维度。
    - numClasses：输出类别的个数。
    - threshold：用于区分正负样本的阈值。
- 方法： clearThreshold()

去除阈值，直接输出预测值，只用于两分类

- 方法：　load(sc, path)

从指定路径加载模型

- 方法：　save(sc, path)

将模型保存到指定路径

- 方法：　predict(x)

预测，输入可以为单个向量或整个RDD

- 方法：　setThreshold(value)

设置用于区分正负样本的阈值。当预测值大于该预置时，判定为正样本。

SVM 支持向量机

###背景知识

SVM是二分类的分类模型。给定包含正负样本的数据集，SVM的目的是寻找一个超平面（WX+b=0对样本进行分割，且使得离超平面比较近的点能有更大的间距。

（待补充）

###Spark API

- 类：class pyspark.mllib.classification.SVMWithSGD
  - 方法：

train(data, iterations=100, step=1.0, regParam=0.01, miniBatchFraction=1.0, initialWeights=one, regType='l2', intercept=False, validateData=True, convergenceTol=0.001)

通过给定的数据训练SVM模型。

- data：训练数据，LabeledPoint格式的RDD数据集。
- iterations：迭代次数，默认为100。
- step：SGD的步长，默认为1.0。
- regParam：规则化参数，默认0.01。
- miniBatchFraction：用于每次SGD迭代的数据，默认1.0。
- initialWeights：初始权值，默认None。
- regType：用于训练模型的规则化类型，可选为l1或l2（默认）。
- intercept：布尔值，表示是否使用增强表现来训练数据，默认False。
- validateData：布尔值，表示算法是否在训练前检验数据，默认True。
- convergenceTol：终止迭代的收敛值，默认0.001。

- 类： pyspark.mllib.classification.SVMModel

支持向量机模型

  - 属性：
    - weights：每个向量计算的权值。
    - intercept：该模型的计算截距。
  - 方法： clearThreshold()

去除阈值，直接输出预测值

  - 方法： load(sc, path)

从指定路径加载模型

  - 方法： save(sc, path)

将模型保存到指定路径

  - 方法： predict(x)

预测，输入可以为单个向量或整个RDD

  - 方法： setThreshold(value)

设置用于区分正负样本的阈值。当预测值大于该预置时，判定为正样本。

##NaiveBayes 朴素贝叶斯

###背景知识

贝叶斯概率公式：

P(B[j]|A[i])=P(A[i]|B[j])P(B[j]) / P(A[i])

朴素贝叶斯分类器是使用贝叶斯概率公式为核心的分类算法，其基本思想为：对于给出的待分类项，解在此项出现的条件下各个类别出现的概率，哪个最大，就认为此待分类项属于哪个类别。

朴素贝叶斯假定样本的不同特征属性对样本的归类影响时相互独立的。

（待补充）

###Spark API

- 类：pyspark.mllib.classification.NaiveBayes
  - 方法：

train(data, lambda_=1.0)

通过给定数据集训练贝叶斯模型

  - data：训练数据，LabeledPoint格式的RDD数据集。
  - lambda：平滑参数，默认1.0

- 类： pyspark.mllib.classification.NaiveBayesModel

朴素贝叶斯分类器模型

  - 属性：
    - labels：label列表
    - pi：每个类别的priors
    - theta：使用矩阵存储每个向量划分到每个类的条件概率
  - 方法： load(sc, path)

从指定路径加载模型

  - 方法： save(sc, path)

将模型保存到指定路径

  - 方法： predict(x)

预测，输入可以为单个向量或整个RDD