  
第12章 鳶尾花分類補充密技—簡單版



### 讀入資料檔

|  |  |
| --- | --- |
| 指令 | 說明 |
| 進入Ubuntu環境 | |
| start-all.sh | 啟動hadoop，若已經啟動，則不需要再執行此指令 |
| cd ~ | 回到家目錄 |
| wget http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data | 下載鳶尾花資料集 |
| hdfs dfs -mkdir /iris | 在hdfs上建一目錄 |
| hdfs dfs -put ~/iris.data /iris | 將資料集上傳hdfs |
| hdfs dfs -ls /iris | 查看hdfs資料 |
| 進入spark-shell環境 | |
| spark-shell | 進入spark-shell |
| val lines = sc.textFile("hdfs://master:9000/iris/iris.data").filter(x=> !x.isEmpty).map(x=>x.split(","))  lines.count()  lines.first  val irisIndex=lines.map(line=>line(4)).distinct.sortBy(x=>x).zipWithIndex.collectAsMap  irisIndex  val irisNameMap=irisIndex.toArray.map(x=>(x.\_2,x.\_1)).toMap  irisNameMap  import org.apache.spark.mllib.regression.LabeledPoint  import org.apache.spark.mllib.linalg.Vectors  val irisLP=lines.map(line=>new LabeledPoint(irisIndex(line.last).toDouble, Vectors.dense(line.init.map(x=>x.toDouble))))  irisLP.first  irisLP.first.label  irisLP.first.features  val Array(irisTrainSet, irisValidSet, irisTestSet) = irisLP.randomSplit(Array(0.7, 0.2, 0.1),168)  irisTrainSet.count  irisValidSet.count  irisTestSet.count  import org.apache.spark.mllib.tree.RandomForest  val impurity="gini"  val maxDepth=3  val numTrees=5  val maxBins=5  val irisModel = RandomForest.trainClassifier(irisTrainSet, 3, Map[Int, Int](), numTrees, "auto" , impurity, maxDepth, maxBins)  def getPrecision(model:org.apache.spark.mllib.tree.model.RandomForestModel, dataset:org.apache.spark.rdd.RDD[org.apache.spark.mllib.regression.LabeledPoint])={  val precision = dataset.map { x =>  val pred = model.predict(x.features)  if (x.label == pred) 1.0 else 0.0  }.mean()  precision  }  println("預測準確率＝" + getPrecision(irisModel, irisValidSet))  println("預測準確率＝" + getPrecision(irisModel, irisTrainSet))  def modelAndPrecision(numTrees:Int, impurity:String, maxDepth:Int, maxBins:Int)={  val irisModel = RandomForest.trainClassifier(irisTrainSet, 3, Map[Int, Int](), numTrees, "auto" , impurity, maxDepth, maxBins)  val precision=getPrecision(irisModel, irisValidSet)  println(f"Precision:$precision%2.4f <== numTrees:$numTrees%2d, impurity:$impurity%8s, maxDepth:$maxDepth%2d, maxBins:$maxBins%2d")  (precision, irisModel)  }  var bestPres=Double.MinValue  var bestModel:org.apache.spark.mllib.tree.model.RandomForestModel=null  for(maxBins<-Array(5,10,20); maxDepth<-Array(3,5); numTrees<-Array(5,7); impurity<-Array("gini", "entropy")) {  val (tempPres, tempModel)=modelAndPrecision(numTrees, impurity, maxDepth, maxBins)  if(tempPres > bestPres) {  bestPres=tempPres  bestModel=tempModel  }  }  println("最佳預測準確率＝" + bestPres)  println("測試資料集準確率＝" + getPrecision(bestModel, irisTestSet))  val newFeatures=Array(Vectors.dense(7.9,4.4,6.9,2.5),Vectors.dense(4.3,2,1,0.1))  newFeatures.foreach(x=>println(x + " 預測為： " +irisNameMap(bestModel.predict(x).toInt)))  println("隨機森林決策樹" + bestModel.toDebugString) | 讀入資料  建立分類索引號  建立分類名稱對照表  建立LabeledPoint類別  資料集隨機分割  設定隨機森林分類參數  開始訓練  計算模式預測準確度  模式訓練與計算準確度副程式  開始跑迴圈，計算所有超參數組合的模式和準確度  計算模式準確度  計算測試資料集的準確度  輸入欲預測的資料  輸出預測結果  查看隨機森林的決策樹s |