厦门大学林子雨，郑海山，赖永炫 编著

《Spark编程基础（Python版）》

教材配套

机房上机实验指南

实验7 Spark机器学习库MLlib编程实践

（版本号：2020年4月版本）

（答案）



主讲教师：林子雨

厦门大学数据库实验室

二零二零年四月

目录

[一、实验目的 1](#_Toc494984827)

[二、实验平台 1](#_Toc494984828)

[三、实验内容和要求 1](#_Toc494984829)

[1.数据导入 1](#_Toc494984830)

[2.进行主成分分析（PCA） 2](#_Toc494984831)

[3.训练分类模型并预测居民收入 4](#_Toc494984832)

[4.超参数调优 6](#_Toc494984833)

[四、实验报告 8](#_Toc494984834)

[**附录1:任课教师介绍** 9](#_Toc494984835)

[**附录2：课程教材介绍** 9](#_Toc494984836)

[**附录3：高校大数据课程公共服务平台介绍** 10](#_Toc494984837)

厦门大学林子雨,赖永炫,陶继平 编著

《Spark编程基础（Python版）》

第8章 Spark MLlib

教材配套机房上机实验指南

实验7 Spark机器学习库MLlib编程实践

（答案）

**主讲教师：林子雨**

E-mail: [ziyulin@xmu.edu.cn](mailto:ziyulin@xmu.edu.cn) 个人主页：<http://www.cs.xmu.edu.cn/linziyu>

# 一、实验目的

（1）通过实验掌握基本的MLLib编程方法；

（2）掌握用MLLib解决一些常见的数据分析问题，包括数据导入、成分分析和分类和预测等。

# 二、实验平台

操作系统：Ubuntu16.04

JDK版本：1.8或以上版本

Spark版本：2.4.0

Python版本：3.4.3

数据集：下载Adult数据集(http://archive.ics.uci.edu/ml/datasets/Adult)，该数据集也可以直接到本教程官网的“下载专区”的“数据集”中下载。数据从美国1994年人口普查数据库抽取而来，可用来预测居民收入是否超过50K$/year。该数据集类变量为年收入是否超过50k$，属性变量包含年龄、工种、学历、职业、人种等重要信息，值得一提的是，14个属性变量中有7个类别型变量。

# 三、实验内容和要求

## 1.数据导入

从文件中导入数据，并转化为DataFrame。

**【参考答案】**

答案：

//导入需要的包

from pyspark.ml.feature import PCA

from pyspark.sql import Row

from pyspark.ml.linalg import Vector,Vectors

from pyspark.ml.evaluation import MulticlassClassificationEvaluator

from pyspark.ml import Pipeline,PipelineModel

from pyspark.ml.feature import IndexToString, StringIndexer, VectorIndexer,HashingTF, Tokenizer

from pyspark.ml.classification import LogisticRegression

from pyspark.ml.classification import LogisticRegressionModel

from pyspark.ml.classification import BinaryLogisticRegressionSummary, LogisticRegression

from pyspark.sql import functions

from pyspark.ml.tuning import CrossValidator, ParamGridBuilder

//获取训练集测试集（需要对测试集进行一下处理，adult.data.txt的标签是>50K和<=50K，而adult.test.txt的标签是>50K.和<=50K.，这里是把adult.test.txt标签的“.”去掉了。另外，确保adult.data.txt和adult.test.txt最后没有多一个空格。）

>>> def f(x):

rel = {}

rel['features']=Vectors.dense(float(x[0]),float(x[2]),float(x[4]),float(x[10]),float(x[11]),float(x[12]))

rel['label'] = str(x[14])

return rel

>>> df = spark.sparkContext.textFile("file:///usr/local/spark/adult.data.txt").map(lambda line: line.split(',')).map(lambda p: Row(\*\*f(p))).toDF()

df: pyspark.sql.DataFrame = [features: vector, label: string]

>>> test = spark.sparkContext.textFile("file:///usr/local/spark/adult.test.txt").map(lambda line: line.split(',')).map(lambda p: Row(\*\*f(p))).toDF()

test: pyspark.sql.DataFrame = [features: vector, label: string]

## 2.进行主成分分析（PCA）

对6个连续型的数值型变量进行主成分分析。PCA（主成分分析）是通过正交变换把一组相关变量的观测值转化成一组线性无关的变量值，即主成分的一种方法。PCA通过使用主成分把特征向量投影到低维空间，实现对特征向量的降维。请通过setK()方法将主成分数量设置为3，把连续型的特征向量转化成一个3维的主成分。

**【参考答案】**

构建PCA模型，并通过训练集进行主成分分解，然后分别应用到训练集和测试集

>>> pca = PCA(k=3, inputCol="features", outputCol="pcaFeatures").fit(df)

pca: pyspark.ml.feature.PCAModel = PCA\_4a668f4a52beccad9526

>>> result = pca.transform(df)

result: pyspark.sql.DataFrame = [features: vector, label: string, pcaFeatures: vector]

>>> testdata = pca.transform(test)

testdata: pyspark.sql.DataFrame = [features: vector, label: string, pcaFeatures: vector]

>>> result.show(truncate=False)

+------------------------------------+------+-----------------------------------------------------------+

|features |label |pcaFeatures |

+------------------------------------+------+-----------------------------------------------------------+

|[39.0,77516.0,13.0,2174.0,0.0,40.0] | <=50K|[77516.0654328193,-2171.6489938846585,-6.9463604765987625] |

|[50.0,83311.0,13.0,0.0,0.0,13.0] | <=50K|[83310.99935595776,2.526033892790795,-3.38870240867987] |

|[38.0,215646.0,9.0,0.0,0.0,40.0] | <=50K|[215645.99925048646,6.551842584546877,-8.584953969073675] |

|[53.0,234721.0,7.0,0.0,0.0,40.0] | <=50K|[234720.99907961802,7.130299808613842,-9.360179790809983] |

|[28.0,338409.0,13.0,0.0,0.0,40.0] | <=50K|[338408.9991883054,10.289249842810678,-13.36825187163136] |

|[37.0,284582.0,14.0,0.0,0.0,40.0] | <=50K|[284581.9991669545,8.649756033705797,-11.281731333793557] |

|[49.0,160187.0,5.0,0.0,0.0,16.0] | <=50K|[160186.99926937037,4.86575372118689,-6.394299355794958] |

|[52.0,209642.0,9.0,0.0,0.0,45.0] | >50K |[209641.99910851708,6.366453450443119,-8.38705558572268] |

|[31.0,45781.0,14.0,14084.0,0.0,50.0]| >50K |[45781.42721110636,-14082.596953729324,-26.3035091053821] |

|[42.0,159449.0,13.0,5178.0,0.0,40.0]| >50K |[159449.15652342222,-5173.151337268416,-15.351831002507415]|

|[37.0,280464.0,10.0,0.0,0.0,80.0] | >50K |[280463.9990886109,8.519356755954709,-11.188000533447731] |

|[30.0,141297.0,13.0,0.0,0.0,40.0] | >50K |[141296.99942061215,4.2900981666986855,-5.663113262632686] |

|[23.0,122272.0,13.0,0.0,0.0,30.0] | <=50K|[122271.9995362372,3.7134109235547164,-4.887549331279983] |

|[32.0,205019.0,12.0,0.0,0.0,50.0] | <=50K|[205018.99929839539,6.227844686207229,-8.176186180265503] |

|[40.0,121772.0,11.0,0.0,0.0,40.0] | >50K |[121771.99934864056,3.6945287780540603,-4.918583567278704] |

|[34.0,245487.0,4.0,0.0,0.0,45.0] | <=50K|[245486.99924622496,7.4601494174606815,-9.75000324288002] |

|[25.0,176756.0,9.0,0.0,0.0,35.0] | <=50K|[176755.9994399727,5.370793765347799,-7.029037217537133] |

|[32.0,186824.0,9.0,0.0,0.0,40.0] | <=50K|[186823.99934678187,5.675541056422981,-7.445605003141515] |

|[38.0,28887.0,7.0,0.0,0.0,50.0] | <=50K|[28886.99946951148,0.8668334219437271,-1.2969921640115318] |

|[43.0,292175.0,14.0,0.0,0.0,45.0] | >50K |[292174.9990868344,8.87932321571431,-11.599483225618247] |

+------------------------------------+------+-----------------------------------------------------------+

only showing top 20 rows

>>> testdata.show(truncate=False)

+------------------------------------+------+-----------------------------------------------------------+

|features |label |pcaFeatures |

+------------------------------------+------+-----------------------------------------------------------+

|[25.0,226802.0,7.0,0.0,0.0,40.0] | <=50K|[226801.99936708904,6.893313042325555,-8.993983821758796] |

|[38.0,89814.0,9.0,0.0,0.0,50.0] | <=50K|[89813.99938947687,2.7209873244764906,-3.6809508659704675] |

|[28.0,336951.0,12.0,0.0,0.0,40.0] | >50K |[336950.99919122306,10.244920104026273,-13.310695651856003]|

|[44.0,160323.0,10.0,7688.0,0.0,40.0]| >50K |[160323.23272903427,-7683.121090489607,-19.729118648470976]|

|[18.0,103497.0,10.0,0.0,0.0,30.0] | <=50K|[103496.99961293535,3.142862309150963,-4.141563083946321] |

|[34.0,198693.0,6.0,0.0,0.0,30.0] | <=50K|[198692.9993369046,6.03791177465338,-7.894879761309586] |

|[29.0,227026.0,9.0,0.0,0.0,40.0] | <=50K|[227025.99932507655,6.899470708670979,-9.011878890810314] |

|[63.0,104626.0,15.0,3103.0,0.0,32.0]| >50K |[104626.09338764261,-3099.8250060692035,-9.648800672052692]|

|[24.0,369667.0,10.0,0.0,0.0,40.0] | <=50K|[369666.99919110356,11.241251385609905,-14.581104454203475]|

|[55.0,104996.0,4.0,0.0,0.0,10.0] | <=50K|[104995.9992947583,3.186050789405019,-4.236895975019816] |

|[65.0,184454.0,9.0,6418.0,0.0,40.0] | >50K |[184454.1939240066,-6412.391589847388,-18.518448307264528] |

|[36.0,212465.0,13.0,0.0,0.0,40.0] | <=50K|[212464.99927015396,6.455148844458399,-8.458640605561254] |

|[26.0,82091.0,9.0,0.0,0.0,39.0] | <=50K|[82090.999542367,2.489111409624171,-3.335593188553175] |

|[58.0,299831.0,9.0,0.0,0.0,35.0] | <=50K|[299830.9989556855,9.111696151562521,-11.909141441347733] |

|[48.0,279724.0,9.0,3103.0,0.0,48.0] | >50K |[279724.0932834471,-3094.495799296398,-16.491321474159864] |

|[43.0,346189.0,14.0,0.0,0.0,50.0] | >50K |[346188.9990067698,10.522518314317386,-13.720686643182727] |

|[20.0,444554.0,10.0,0.0,0.0,25.0] | <=50K|[444553.9991678726,13.52288689604709,-17.47586621453762] |

|[43.0,128354.0,9.0,0.0,0.0,30.0] | <=50K|[128353.99933456781,3.895809826834201,-5.163630508998832] |

|[37.0,60548.0,9.0,0.0,0.0,20.0] | <=50K|[60547.99950268136,1.834388499828796,-2.482228457083787] |

|[40.0,85019.0,16.0,0.0,0.0,45.0] | >50K |[85018.99937940767,2.5751267063691055,-3.4924978737087193] |

+------------------------------------+------+-----------------------------------------------------------+

only showing top 20 rows

## 3.训练分类模型并预测居民收入

在主成分分析的基础上，采用逻辑斯蒂回归，或者决策树模型预测居民收入是否超过50K；对Test数据集进行验证。

**【参考答案】**

训练逻辑斯蒂回归模型，并进行测试，得到预测准确率

>>> labelIndexer = StringIndexer(inputCol="label", outputCol="indexedLabel").fit(result)

labelIndexer: pyspark.ml.feature.StringIndexerModel = StringIndexer\_49fd892bf407764dcffb

>>> for label in labelIndexer.labels:print(label)

<=50K

>50K

>>> featureIndexer = VectorIndexer(inputCol="pcaFeatures", outputCol="indexedFeatures").fit(result)

featureIndexer: pyspark.ml.feature.VectorIndexerModel = VectorIndexer\_48bc920d8af88e337d21

>>> print(featureIndexer.numFeatures)

3

>>> labelConverter = IndexToString(inputCol="prediction", outputCol="predictedLabel",labels=labelIndexer.labels)

labelConverter: pyspark.ml.feature.IndexToString = IndexToString\_40e99a67399e57d7950c

>>> lr = LogisticRegression().setLabelCol("indexedLabel").setFeaturesCol("indexedFeatures").setMaxIter(100)

lr: pyspark.ml.classification.LogisticRegression = LogisticRegression\_44efaefad414357b7c36

>>> lrPipeline = Pipeline().setStages([labelIndexer, featureIndexer, lr, labelConverter])

lrPipeline: pyspark.ml.Pipeline = Pipeline\_49a886038fe4366cb525

>>> lrPipelineModel = lrPipeline.fit(result)

lrPipelineModel: pyspark.ml.PipelineModel = PipelineModel\_43eb8e7d01dae015460c

>>> lrModel = lrPipelineModel.stages[2]

lrModel:pyspark.ml.classification.LogisticRegressionModel = LogisticRegression\_44efaefad414357b7c36

>>> print ("Coefficients: \n " + str(lrModel.coefficientMatrix)+"\nIntercept: "+str(lrModel.interceptVector)+ "\n numClasses: "+str(lrModel.numClasses)+"\n numFeatures: "+str(lrModel.numFeatures))

Coefficients:

DenseMatrix([[-1.98285864e-07, -3.50909247e-04, -8.45150628e-04]])

Intercept: [-1.4525982557843347]

numClasses: 2

numFeatures: 3

>>> lrPredictions = lrPipelineModel.transform(testdata)

lrPredictions: pyspark.sql.DataFrame = DataFrame[features: vector, label: string, pcaFeatures: vector, indexedLabel: double, indexedFeatures: vector, rawPrediction: vector, probability: vector, prediction: double, predictedLabel: string]

>>> evaluator = MulticlassClassificationEvaluator().setLabelCol("indexedLabel").setPredictionCol("prediction")

evaluator: pyspark.ml.evaluation.MulticlassClassificationEvaluator = MulticlassClassificationEvaluator\_44fb8a00fb8868ae541f

>>> lrAccuracy = evaluator.evaluate(lrPredictions)

lrAccuracy: Double = 0.7764235163053484

>>> print("Test Error = %g " % (1.0 - lrAccuracy))

Test Error = 0.223576

## 4.超参数调优

利用CrossValidator确定最优的参数，包括最优主成分PCA的维数、分类器自身的参数等。

**【参考答案】**

>>> pca = PCA().setInputCol("features").setOutputCol("pcaFeatures")

pca: pyspark.ml.feature.PCA = PCA\_465ea3aeee8f823b1cc2

>>> labelIndexer = StringIndexer().setInputCol("label").setOutputCol("indexedLabel").fit(df)

labelIndexer: pyspark.ml.feature.StringIndexerModel = StringIndexer\_4a4caa1f671823df2712

>>> featureIndexer = VectorIndexer().setInputCol("pcaFeatures").setOutputCol("indexedFeatures")

featureIndexer: pyspark.ml.feature.VectorIndexer = VectorIndexer\_4a87a808787866220518

>>> labelConverter = IndexToString().setInputCol("prediction").setOutputCol("predictedLabel").setLabels(labelIndexer.labels)

labelConverter: pyspark.ml.feature.IndexToString = IndexToString\_444190300664cc71e5b5

>>> lr = LogisticRegression().setLabelCol("indexedLabel").setFeaturesCol("indexedFeatures").setMaxIter(100)

lr: pyspark.ml.classification.LogisticRegression = LogisticRegression\_4ff3b577b810fd21ab1b

>>> lrPipeline = Pipeline().setStages([pca, labelIndexer, featureIndexer, lr, labelConverter])

lrPipeline: pyspark.ml.Pipeline = Pipeline\_4165a34a906306ee044a

>>> paramGrid = ParamGridBuilder().addGrid(pca.k, [1,2,3,4,5,6]).addGrid(lr.elasticNetParam, [0.2,0.8]).addGrid(lr.regParam, [0.01, 0.1, 0.5]).build()

paramGrid: Array[pyspark.ml.param.ParamMap] =

{Param(parent=u'LogisticRegression\_4ff3b577b810fd21ab1b', name='elasticNetParam', doc='the ElasticNet mixing parameter, in range [0, 1]. For alpha = 0, the penalty is an L2 penalty. For alpha = 1, it is an L1 penalty.'): 0.2, Param(parent=u'LogisticRegression\_4ff3b577b810fd21ab1b', name='regParam', doc='regularization parameter (>= 0).'): 0.01, Param(parent=u'PCA\_465ea3aeee8f823b1cc2', name='k', doc='the number of principal components'): 1}

{Param(parent=u'LogisticRegression\_4ff3b577b810fd21ab1b', name='elasticNetParam', doc='the ElasticNet mixing parameter, in range [0, 1]. For alpha = 0, the penalty is an L2 penalty. For alpha = 1, it is an L1 penalty.'): 0.2, Param(parent=u'LogisticRegression\_4ff3b577b810fd21ab1b', name='regParam', doc='regularization parameter (>= 0).'): 0.01, Param(parent=u'PCA\_465ea3aeee8f823b1cc2', name='k', doc='the number of principal components'): 2}

{Param(parent=u'LogisticRegression\_4ff3b577b810fd21ab1b', name='elasticNetParam', doc='the ElasticNet mixing parameter, in range [0, 1]. For alpha = 0, the penalty is an L2 penalty. For alpha = 1, it is an L1 penalty.'): 0.2, Param(parent=u'LogisticRegression\_4ff3b577b810fd21ab1b', name='regParam', doc='regularization parameter (>= 0).'): 0.01, Param(parent=u'PCA\_465ea3ae……

>>> cv = CrossValidator().setEstimator(lrPipeline).setEvaluator(MulticlassClassificationEvaluator().setLabelCol("indexedLabel").setPredictionCol("prediction")).setEstimatorParamMaps(paramGrid).setNumFolds(3)

cv: pyspark.ml.tuning.CrossValidator = CrossValidator\_4d4eaeb04035ccae91e2

>>> cvModel = cv.fit(df)

cvModel: pyspark.ml.tuning.CrossValidatorModel = CrossValidatorModel\_4601a7d61debbfd3544e

>>> lrPredictions=cvModel.transform(test)

lrPredictions: pyspark.sql.DataFrame = [features: vector, label: string, pcaFeatures: vector, indexedLabel: double, indexedFeatures: vector, rawPrediction: vector, probability: vector, prediction: double, predictedLabel: string]

>>> evaluator = MulticlassClassificationEvaluator().setLabelCol("indexedLabel").setPredictionCol("prediction")

evaluator: pyspark.ml.evaluation.MulticlassClassificationEvaluator = MulticlassClassificationEvaluator\_40bfa39a6a73931437c8

>>> lrAccuracy = evaluator.evaluate(lrPredictions)

lrAccuracy: Double = 0.7833268290041506

>>> print("准确率为"+str(lrAccuracy))

准确率为0.7833268290041506

>>> bestModel= cvModel.bestModel

bestModel: pyspark.ml.PipelineModel = PipelineModel\_47388ab70ca452562894

>>> lrModel = bestModel.stages[3]

lrModel: pyspark.ml.classification.LogisticRegressionModel = LogisticRegression\_46d894d2cea1ed552ec5

>>> print ("Coefficients: \n " + str(lrModel.coefficientMatrix)+"\nIntercept: "+str(lrModel.interceptVector)+ "\n numClasses: "+str(lrModel.numClasses)+"\n numFeatures: "+str(lrModel.numFeatures))

Coefficients:

DenseMatrix([[-1.50035172e-07, -1.68933655e-04, -8.83869475e-04,

4.92262006e-02, 3.10992712e-02, -2.81742804e-01]])

Intercept: [-7.459195847829245]

numClasses: 2

numFeatures: 6

>>> pcaModel = bestModel.stages[0]

pcaModel: pyspark.ml.feature.PCAModel = PCA\_423c88604bc4e9c371f3

>>> print("Primary Component: " + str(pcaModel.pc))

Primary Component: -9.905077142269292E-6 -1.435140700776355E-4 ... (6 total)

0.9999999987209459 3.0433787125958012E-5 ...

-1.0528384042028638E-6 -4.2722845240104086E-5 ...

3.036788110999389E-5 -0.9999984834627625 ...

-3.9138987702868906E-5 0.0017298954619051868 ...

-2.1955537150508903E-6 -1.3109584368381985E-4 ...

可以看出，PCA最优的维数是6。

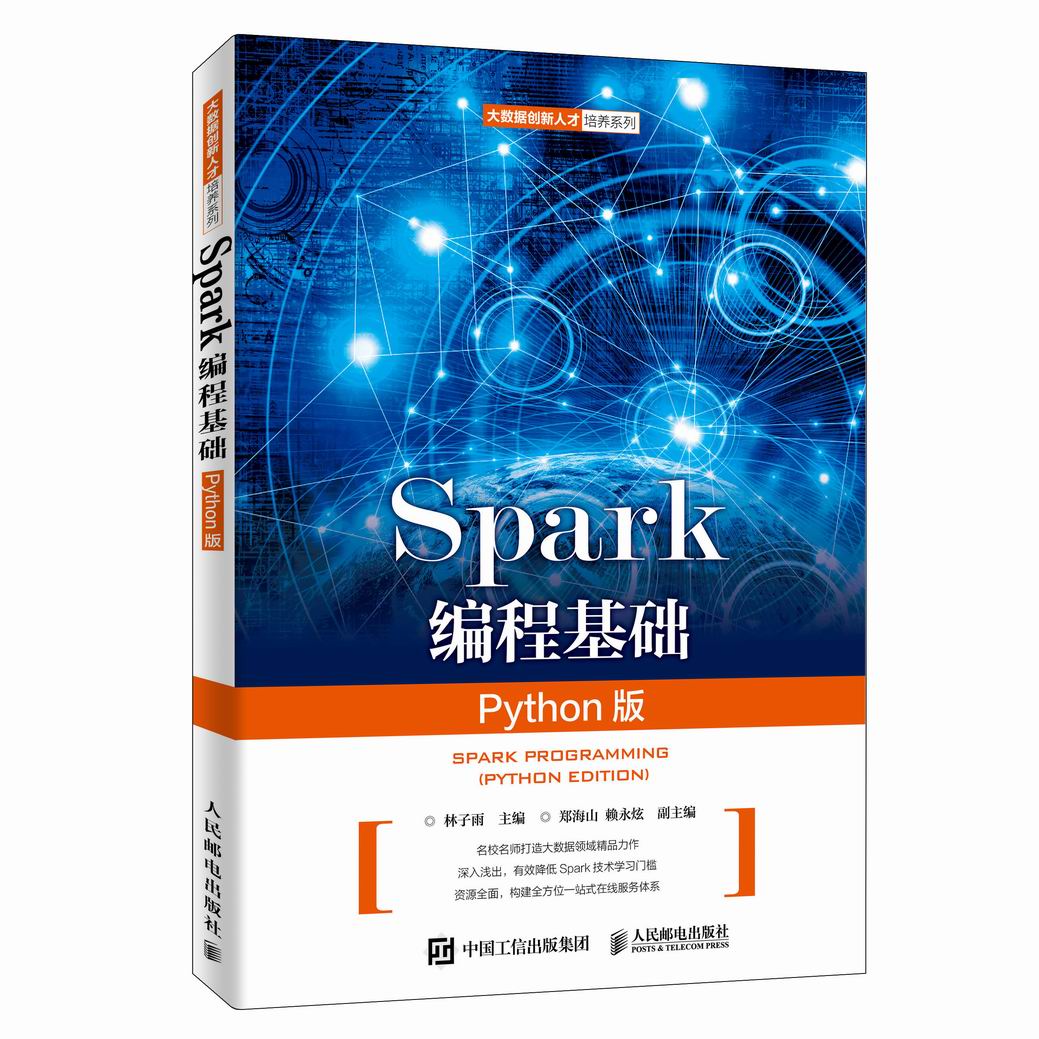
# 四、实验报告

|  |  |  |
| --- | --- | --- |
| 《Spark编程基础（Python版）》实验报告 | | |
| 题目： | 姓名： | 日期： |
| 实验环境： | | |
| 实验内容与完成情况： | | |
| 出现的问题： | | |
| 解决方案（列出遇到的问题和解决办法，列出没有解决的问题）： | | |

**附录1:任课教师介绍**

|  |  |
| --- | --- |
| [http://www.cs.xmu.edu.cn/database/linziyu/images/linziyu2016.jpg](http://dblab.xmu.edu.cn/post/linziyu/) | 林子雨（1978－），男，博士，国内高校知名大数据教师，厦门大学计算机科学系副教授，厦门大学云计算与大数据研究中心创始成员，厦门大学数据库实验室负责人，中国计算机学会数据库专委会委员，中国计算机学会信息系统专委会委员，中国高校首个“数字教师”提出者和建设者。2013年开始在厦门大学开设大数据课程，建设了国内高校首个大数据课程公共服务平台，平台累计网络访问量超过1000万次，成为全国高校大数据教学知名品牌，并荣获“2018年福建省教学成果二等奖”，主持的课程《大数据技术原理与应用》获评“2018年国家精品在线开放课程”。  E-mail: ziyulin@xmu.edu.cn  个人主页：http://dblab.xmu.edu.cn/post/linziyu  数据库实验室网站：http://dblab.xmu.edu.cn |

**附录2：课程教材介绍**



林子雨、郑海山、赖永炫编著《Spark编程基础（Python版）》

人民邮电出版社 ISBN:978-7-115-52439-3 定价：49.80元

厦门大学林子雨、郑海山、赖永炫老师编著《Spark编程基础（Python版）》，以Python作为开发Spark应用程序的编程语言，系统介绍了Spark编程的基础知识。全书共8章，内容包括大数据技术概述、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Structured Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作，以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源，包括讲义PPT、习题、源代码、软件、数据集、授课视频、上机实验指南等。

本书可以作为高等院校计算机、软件工程、数据科学与大数据技术等专业的进阶级大数据课程教材，用于指导Spark编程实践，也可供相关技术人员参考。

欢迎访问《Spark编程基础(Python版)》教材官方网站：http://dblab.xmu.edu.cn/post/spark-python/



扫一扫访问教材官网

**附录3：高校大数据课程公共服务平台介绍**



高校大数据课程公共服务平台，由中国高校首个“数字教师”的提出者和建设者——林子雨老师发起，由厦门大学数据库实验室全力打造，由厦门大学云计算与大数据研究中心携手共建。这是国内第一个服务于高校大数据课程建设的公共服务平台，旨在促进国内高校大数据课程体系建设，提高大数据课程教学水平，降低大数据课程学习门槛，提升学生课程学习效果。平台服务对象涵盖高校、教师和学生。平台为高校开设大数据课程提供全流程辅助，为教师开展教学工作提供一站式服务，为学生学习大数据课程提供全方位辅导。平台重点打造“11个1工程”，即1本教材（含官网）、1个教师服务站、1个学生服务站、1个公益项目、1堂巡讲公开课、1个示范班级、1门在线课程、1个交流群（QQ群、微信群）、1个保障团队、1个培训基地和1个实验平台。目前平台每年访问量已经超过200万次，累计访问量超过1000万次，成为国内高校大数据教学知名品牌。

平台主页：http://dblab.xmu.edu.cn/post/bigdata-teaching-platform/



扫一扫访问平台主页