TÀI LIỆU

HƯỚNG DẪN SỬ DỤNG ZEPPELIN

|  |  |
| --- | --- |
| **Tên dự án** | DATALAKE RAL |
| **Mã hợp đồng** |  |
| **Mã tài liệu** | DLRAL-ZEPPELIN-MANUAL |
| **Phiên bản** | 0.1 |
| **Ngày cập nhật** | 05/09/2023 |

**QUẢN LÝ THAY ĐỔI**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ngày thay đổi** | **Mục, bảng, sơ đồ được thay đổi** | **Mô tả thay đổi** | **Phiên bản** |
| 05/09/2023 | N/A | Tạo mới | 0.1 |
|  |  |  |  |

**MỤC LỤC**

[1. THÔNG TIN CHUNG 4](#_Toc146707650)

[1.1. Mục đích tài liệu 4](#_Toc146707651)

[1.2. Phạm vi tài liệu 4](#_Toc146707652)

[1.3. Định nghĩa và các từ viết tắt 4](#_Toc146707653)

[1.4. Các tài liệu liên quan 4](#_Toc146707654)

[2. HƯỚNG DẪN SỬ DỤNG 5](#_Toc146707655)

[2.1. Thông tin hệ thống 5](#_Toc146707656)

[2.2. Lưu ý chung 5](#_Toc146707657)

[2.3. Truy cập vào Zeppelin 6](#_Toc146707658)

[2.4. Cách lưu dữ liệu về CSDL MariaDB 8](#_Toc146707659)

[2.5. Show HDFS và tìm kiếm file theo tên 9](#_Toc146707660)

[2.6. Code xử lý dữ liệu thô 9](#_Toc146707661)

[2.7. Thông tin nguồn dữ liệu đồng bộ hàng tháng (n-1) 10](#_Toc146707662)

[2.8. Thêm library cho Interpreter 11](#_Toc146707663)

[2.9. Thêm module Python cho Zeppelin 12](#_Toc146707664)

[2.10. Thêm lib học máy cho python 13](#_Toc146707665)

[2.11. Chạy Pyspark trên Zeppelin 20](#_Toc146707666)

1. THÔNG TIN CHUNG
   1. Mục đích tài liệu

* **Mục đích**: Tài liệu này cung cấp hướng dẫn sử dụng Zeppelin cho mục tiêu demo dịch vụ
* **Đối tượng sử dụng**: Team DS, DE.
  1. Phạm vi tài liệu
* Tập trung tập trung vào hướng dẫn cấu hình và hướng dẫn sử dụng các tính năng quan trọng, cần lưu tâm trong dự án DataLake RAL.
* Tài liệu này không tập trung vào hướng dẫn chi tiết cách sử dụng Zeppelin cùng các tính năng cơ bản, tham khảo thêm tài liệu từ trang chủ Zeppelin để biết thêm chi tiết.
  1. Định nghĩa và các từ viết tắt

|  |  |  |
| --- | --- | --- |
| **STT** | **Từ viết tắt/ Thuật ngữ** | **Ý nghĩa** |
|  | HDSD | Hướng dẫn sử dụng |
|  | Library | Thư viện |
|  | OS | Operating System, hệ điều hành |
|  | Interpreter | Trình thông dịch |

* 1. Các tài liệu liên quan

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên tài liệu** | **Ghi chú** |
|  | <https://zeppelin.apache.org/docs/0.10.1/> | Trang tài liệu của Zeppelin |

1. HƯỚNG DẪN SỬ DỤNG
   1. Thông tin hệ thống

* Node: ral-edge-01 (public IP: 116.103.227.210)
* Apache Spark: 3.3
* Zeppelin: 0.11.0-SNAPSHOT
* MariaDB: 10.6.11
* GCC 4.8.5
* Python 3: 3.9.13

A screenshot of a computer

Description automatically generated

* 1. Lưu ý chung
* Toàn bộ dữ liệu lưu về database **dsral** trên mysql 5.7
* Khi cần lưu dữ liệu xuống database, nếu bảng lớn (quy ước khoảng 10 nghìn dòng trở lên), khuyến nghị sử dụng DataFrameWriter với PySpark và sử dụng JDBC Driver của MariaDB (hướng dẫn phía dưới).
  + <https://spark.apache.org/docs/latest/api/python/reference/pyspark.sql/api/pyspark.sql.DataFrameWriter.html>
* Sử dụng PySpark/Spark hoặc các công cụ trong hệ thống nói chung, người sử dụng nên tự trang bị các kiến thức từ cơ bản đến nâng cao để sử dụng chúng được hiệu quả.
* Mỗi người cụ thể sử dụng 1 tài khoản Zeppelin cụ thể
* Mỗi người khi sử dụng, các notebooks của cá nhân sẽ được tổ chức trong 1 thư mục, ví dụ người dùng dattd6, sẽ có các notebooks trong thư mục dattd6/

A screenshot of a phone

Description automatically generated

* Tên notebook là viết liền không dấu, sử dụng kí tự trong bảng chữ cái tiếng Anh.

Ví dụ: day\_la\_ten\_notebook\_hop\_le

* 1. Truy cập vào Zeppelin

Sau đó dùng trình duyệt, truy cập vào đường dẫn https://116.103.227.210:9995

A screenshot of a computer

Description automatically generated

Chọn **Accept the Risk and Continue**

A screenshot of a computer

Description automatically generated

**Bước 3**:

Đăng nhập với thông tin username và password tương ứng.

* 1. CSDL MySQL

Hiện tại đã cấu hình Spark Interpreter (bao gồm cả PySpark) load được library cho MariaDB Connector/J driver lên runtime của Spark Session, cấu hình ví dụ kèm link tới notebook như phía dưới:

Thông tin db:

User: dsteam, pass: Ral@2804, link database: 116.103.226.245

* 1. Show HDFS và tìm kiếm file theo tên

%sh

hdfs dfs -ls /demo\_bank/raw\_zone

Kết quả:

* 1. Thông tin nguồn dữ liệu

hdfs dfs -ls /demo\_bank/raw\_zone

|  |
| --- |
| -rwxrwxrwx 3 hdfs hdfs 216591900 2023-09-27 10:26 /demo\_bank/raw\_zone/Base.csv -rwxrwxrwx 3 hdfs hdfs 216570280 2023-09-27 10:27 /demo\_bank/raw\_zone/Variant\_I.csv -rwxrwxrwx 3 hdfs hdfs 216680189 2023-09-27 10:28 /demo\_bank/raw\_zone/Variant\_II.csv -rwxrwxrwx 3 hdfs hdfs 255837541 2023-09-27 10:28 /demo\_bank/raw\_zone/Variant\_III.csv -rwxrwxrwx 3 hdfs hdfs 216676711 2023-09-27 10:28 /demo\_bank/raw\_zone/Variant\_IV.csv -rwxrwxrwx 3 hdfs hdfs 255840563 2023-09-27 10:28 /demo\_bank/raw\_zone/Variant\_V.csv |

* 1. Thêm library cho Interpreter

Xem thêm Interpreter là gì: <https://zeppelin.apache.org/docs/latest/usage/interpreter/overview.html>

Truy cập vào Zeppelin

Ở góc trên bên phải của trang, click vào tên người dùng

Chọn “Interpreter”

Tìm “spark”

Chọn “edit” ở góc bên phải

Tìm “spark.jars” và điền đường dẫn đến các gói jars (phân cách bằng dấu phẩy)

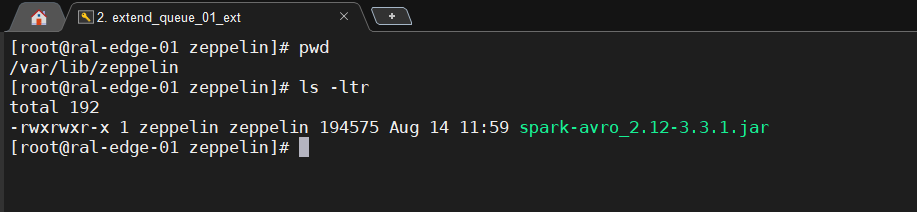
A screenshot of a computer

Description automatically generated

Tìm “spark.files” và điền đường dẫn đến các file (phân cách bằng dấu phẩy)

Chọn “Save”

Phần library tương ứng trên OS:



* 1. Thêm module Python cho Zeppelin

B1: Sftp thư mục lên hệ thống

Sử dụng câu lệnh **sftp -i ral\_zeppelin.pem zeppelin@116.103.227.210**

Ảnh có chứa văn bản, Phần mềm đa phương tiện, ảnh chụp màn hình

Mô tả được tạo tự động

* Put file từ local lên thư mục:
  + Nhập câu lệnh put {đường dẫn file tại local} {đường dẫn file trên máy cài zeppelin}

**put C:\Users\abc\Downloads\abc\testmodule.py \home\zeppelin\libext**

A screen shot of a computer

Description automatically generated

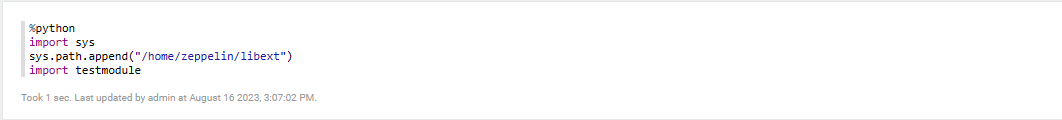
B2: Trỏ đường dẫn vào thư mục chứa module trên zeppelin

%python

import sys

sys.path.append("/home/zeppelin/libext")

import testmodule



B3: Test

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

* 1. Thêm lib học máy cho python

|  |
| --- |
| python3.9 --version  pip3.9 install numpy==1.24.3  pip3.9 install pandas==1.5.3  pip3.9 install catboost==1.2  pip3.9 install lightgbm==3.3.2  pip3.9 install optuna==3.2.0  pip3.9 install prophet==1.1.4  pip3.9 install pymssql==2.2.7  pip3.9 install python\_dateutil==2.8.2  pip3.9 install scikit\_learn==1.2.2  pip3.9 install statsmodels==0.14.0  pip3.9 install tqdm==4.65.0  pip3.9 install pyodbc  pip3.9 install sqlalchemy  pip3.9 install geopandas  pip3.9 install seaborn=0.12.2  pip3.9 install seaborn==0.12.2  pip3.9 install scipy  pip3.9 install xgboost  pip3.9 install dotenv  pip3.9 install dotenv  pip3.9 install yellowbrick  pip3.9 install xlsxwriter  pip3.9 install matplotlib  pip3.9 install plotly  pip3.9 install click  pip3.9 install Sphinx  pip3.9 install coverage  pip3.9 install awscli  pip3.9 install flake8  pip3.9 install python-dotenv==0.5.1  //add to run ipython:  pip3.11 install jupyter-client==6.1.12  pip3.11 install matplotlib==3.4.3  pip3.11 install ipykernel==6.2.0  pip3.11 install ipython==7.26.0  pip3.11 install protobuf==3.16.0  pip3.11 install grpcio==1.38.1  pip3.11 install ipython-genutils==0.2.0 |

**Cấu hình Python Interpreter**:

A screenshot of a computer

Description automatically generated

**Test**:

A screenshot of a computer

Description automatically generated

|  |
| --- |
| %python  #Chúng ta sẽ sử dụng một tập dữ liệu viễn thông để dự đoán tình trạng khách hàng rời đi.  # Đây là tập dữ liệu khách hàng lịch sử, trong đó mỗi hàng đại diện cho một khách hàng. Dữ liệu tương đối  # dễ hiểu và bạn có thể khám phá thông tin chi tiết có thể sử dụng ngay lập tức.  # Thông thường, việc giữ khách hàng sẽ ít tốn kém hơn là tìm các khách hàng mới, vì vậy trọng tâm của phân tích này  # là dự đoán những khách hàng sẽ ở lại với công ty.  #Tập dữ liệu này cung cấp thông tin để giúp bạn dự đoán hành vi nào sẽ giúp bạn giữ chân khách hàng.  # Bạn có thể phân tích tất cả dữ liệu khách hàng có liên quan và phát triển các chương trình tập trung giữ chân khách hàng.  #Tập dữ liệu bao gồm thông tin về:  #- Những khách hàng đã rời đi trong tháng trước - cột Churn  #- Các dịch vụ mà mỗi khách hàng đã đăng ký - điện thoại, nhiều đường truyền, internet, bảo mật trực tuyến,  # sao lưu trực tuyến, bảo vệ thiết bị, hỗ trợ công nghệ, truyền trực tuyến TV và phim  #- Thông tin tài khoản khách hàng - họ đã là khách hàng trong bao lâu, hợp đồng, phương thức thanh toán,  # thanh toán không cần giấy tờ, các khoản phí hàng tháng và tổng các khoản phí  #- Thông tin nhân khẩu học về khách hàng - giới tính, độ tuổi và liệu họ đã kết hôn và có người phụ thuộc  # Load du lieu  import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  churn\_df = pd.read\_csv("/home/zeppelin/ChurnData.csv")  # Trích chọn các đặc trưng  churn\_df = churn\_df[['tenure', 'age', 'address', 'income', 'ed', 'employ', 'equip', 'callcard', 'wireless','churn']]  # chuyen doi kieu du lieu dich thanh so nguyen  churn\_df['churn'] = churn\_df['churn'].astype(int)  print(churn\_df.head(2))  # mo ta du lieu --> 200 row  # so cot 10  print(churn\_df.shape)  # Xac dinh X, y cho tap du lieu --> chuyen sang numpy  X = np.asanyarray(churn\_df[['tenure', 'age', 'address', 'income', 'ed', 'employ', 'equip']])  y = np.asanyarray(churn\_df['churn'])  # chuan hoa tap du lieu mean = 0  from sklearn import preprocessing  X = preprocessing.StandardScaler().fit(X).transform(X)  print("Gia tri trung binh sau khi chuan hoa%.2d" % np.mean(X))  # chi teop du lieu thanh train, test  from sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=4)  print('Train set',X\_train.shape, y\_train.shape)  print('Test set',X\_test.shape, y\_test.shape)  # mô hình hóa sử dụng logistic regression  # LogisticRegression từ gói Scikit-learning. Hàm này thực hiện hồi quy logistic và có thể sử dụng các trình  # tối ưu hóa số khác nhau để tìm các tham số, bao gồm các bộ giải 'newton-cg’, ‘lbfgs’, ‘liblinear’, ‘sag’, ‘saga’  #Phiên bản của Hồi quy Logistic trong Scikit-learning hỗ trợ điều chuẩn (regularization).  # Điều chuẩn là một kỹ thuật được sử dụng để giải quyết bài toán quá khớp (overfitting) trong các mô hình học máy.  # Tham số \*\*C\*\* cho biết \*\*nghịch đảo của cường độ điều chuẩn\*\* phải là một số thực dương.  # Giá trị nhỏ hơn xác định điều chuẩn mạnh hơn. Bây giờ, hãy fit mô hình với train set:  from sklearn.linear\_model import LogisticRegression  LR = LogisticRegression(C=0.01, solver='liblinear').fit(X\_train,y\_train)  # du doan tap test set  yhat = LR.predict(X\_test)  #\*\*predict\_proba\*\* trả về ước lượng cho tất cả các lớp, được sắp xếp theo nhãn của các lớp.  # Vì vậy, cột đầu tiên là xác suất của lớp 0, P (Y = 0|X) và cột thứ hai là xác suất của lớp 1, P (Y = 1|X):  yhat\_prob = LR.predict\_proba(X\_test)  print(yhat\_prob.shape)  # danh gia:  # Hãy thử jaccard index để đánh giá độ chính xác. Chúng ta có thể xác định jaccard là kích thước của giao điểm chia  # cho kích thước hợp của hai tập nhãn. Nếu toàn bộ  # tập các nhãn được dự đoán cho một mẫu hoàn toàn khớp với tập các nhãn thực,  # thì độ chính xác của tập con là 1,0; nếu không thì là 0,0.  from sklearn.metrics import jaccard\_score  print('Do chinh xac theo jaccard:',jaccard\_score(y\_test,yhat, pos\_label=0))  # Co the su dung ma tran nham lan de danh gia  import conFusionMetric  from sklearn.metrics import classification\_report, confusion\_matrix  cnf\_matrix = confusion\_matrix(y\_test, yhat, labels=[1,0])  np.set\_printoptions(precision=2)  # Plot non-normalized confusion matrix  plt.figure()  plt.show()  import sys  sys.path.append("/home/zeppelin/conFusionMetric")  conFusionMetric.plot\_confusion\_matrix(cnf\_matrix, classes=['churn=1','churn=0'],normalize= False, title='Confusion matrix')  print (classification\_report(y\_test, yhat))  #Dựa trên số lượng của từng phần, chúng ta có thể tính precision và recall của từng nhãn:  #Precision là thước đo độ chính xác với điều kiện là nhãn lớp đã được dự đoán, được tính: precision = TP / (TP + FP)  #Recall là tỷ lệ true positive, được tính: Recall = TP / (TP + FN)  #Vậy chúng ta có thể tính toán precision và recall của từng lớp.  #F1 score: Bây giờ chúng ta đang ở vị trí để tính F1 score cho mỗi nhãn dựa trên precision và recall của nhãn đó.  #F1 score là điểm trung bình hài hòa của precision và recall, trong đó F1 score đạt giá trị tốt nhất ở 1  # (precision và recall hoàn hảo) và thấp nhất là 0. Đó là một cách tốt để chứng tỏ rằng bộ phân loại có giá trị tốt  # cho cả recall và precision.  #Và cuối cùng, chúng ta có thể cho biết độ chính xác trung bình cho bộ phân loại này là trung bình của  # F1 score cho cả hai nhãn, trong trường hợp này là 0,72.  #Bây giờ, hãy thử log loss để đánh giá. Trong hồi quy logistic, đầu ra có thể là xác suất khách hàng rời đi (hoặc bằng 1).  # Xác suất này là một giá trị từ 0 đến 1.  #Log loss đo lường hiệu suất của bộ phân loại, trong đó đầu ra dự đoán là giá trị xác suất từ 0 đến 1  from sklearn.metrics import log\_loss  print(log\_loss(y\_test,yhat\_prob)) |

**Kết quả**:

A white rectangular object with blue edges

Description automatically generated

* 1. Chạy Pyspark trên Zeppelin

%pyspark

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName('SparkByExamples.com').getOrCreate()

#Creates Empty RDD

emptyRDD = spark.sparkContext.emptyRDD()

print(emptyRDD)

#Create Schema

from pyspark.sql.types import StructType,StructField, StringType

schema = StructType([

StructField('firstname', StringType(), True),

StructField('middlename', StringType(), True),

StructField('lastname', StringType(), True)

])

#Create empty DataFrame from empty RDD

df = spark.createDataFrame(emptyRDD,schema)

df.printSchema()

A screenshot of a computer

Description automatically generated