TRƯỜNG ĐH CÔNG NGHỆ THÔNG TIN TP.HCM
Khoa Hệ Thống Thông Tin
Chương trình đào tạo: Kỹ sư Hệ thống Thông tin

Bài Thực hành số 5

Tên môn học: ***Dữ liệu lớn***

I. Tóm tắt bài thực hành

1. Yêu cầu lý thuyết

Sinh viên đã được trang bị kiến thức:

- Cấu trúc hệ thống phân tán và framework lập trình Apache Spark
- Đối tượng RDD (Resilient Distributed Dataset) trong Apache Spark
- Lập trình Python với Apache Spark thông qua PySpark
- Các thuật toán máy học

…

2. Nội dung

❖ **Ôn tập lại những kiến thức cần thiết**

❖ **Làm quen với thư viện các thuật toán máy học MLLib**

3. Kết quả cần đạt

✓ Sử dụng được thư viện Spark MLLib để áp dụng các thuật toán máy học trên dữ liệu lớn.

II. Ôn tập lại những kiến thức đã học

Sinh viên tham khảo tài liệu ở buổi học trước về việc sử dụng PySpark để lập trình trong Apache Spark, đối tượng RDD.

Sinh viên tham khảo tài liệu ở các môn học trước, tài liệu trên lớp lý thuyết cũng như trên internet để hiểu rõ về các thuật toán máy học.

III. Yêu cầu bài làm sinh viên

Nội dung thực hành buổi 05 được thực hiện theo từng cá nhân. Sinh viên upload một tập tin **<MSSV>.doc** hoặc **<MSSV>.docx**, nội dung trả lời các bài tập bên dưới.

Lưu ý: Bài nộp không theo đúng quy định này sẽ không được tính.

IV. Làm quen với thư viện các thuật toán máy học (Machine Learning Library – MLLib)

Sử dụng tập tin `data_geo.csv` được cung cấp sẵn để thực hiện bài tập sau.

Bài tập 1: Thao tác với dữ liệu đầu vào

✓ Sử dụng biến SparkSession để đọc dữ liệu đầu vào từ tập tin csv.

```
data = spark.read.format("csv")\
  .option("header", "true")\
  .option("inferSchema", "true")\
  .load("đường dẫn đến tập tin csv")
data.cache()
data.count()
```

✓ Hiển thị thông tin dữ liệu

```
display(data)
```

```
data.printSchema()
#Loại bỏ các dòng có giá trị trống
data = data.dropna()
data.count()
#Tạo View để truy vấn hiển thị dữ liệu
data.createOrReplaceTempView("data_geo")
df1 = spark.sql("select City, `State Code`, `2014
Population estimate`/1000 as `2014 Pop estimate`, `2015
median sales price` from data_geo")
display(df1)
```

✓ Tiền xử lý dữ liệu

```
from pyspark.mllib.regression import LabeledPoint
data = data.select("2014 Population estimate", "2015
median sales price")\
  .map(lambda r: LabeledPoint(r[1], [r[0]]))\
  .toDF()
display(data)
```

✓ Trực quan hóa dữ liệu

```
import numpy as np
import matplotlib.pyplot as plt
x = data.map(lambda p: (p.features[0])).collect()
y = data.map(lambda p: (p.label)).collect()
from pandas import *
from ggplot import *
pydf = DataFrame({'pop':x,'price':y})
p = ggplot(pydf, aes('pop','price')) + \
    geom_point(color='blue')
display(p)
```

Bài tập 2: Sử dụng mô hình hồi quy tuyến tính để dự báo kết quả giá bán

✓ Thêm lớp LinearRegression từ thư viện máy học

```
from pyspark.ml.regression import LinearRegression
```

✓ Tạo biến LinearRegression để sử dụng

```
lr = LinearRegression()
```

✓ Tạo ra hai mô hình tương ứng với hai tham số

```
modelA = lr.fit(data, {lr.regParam:0.0})
modelB = lr.fit(data, {lr.regParam:100.0})
```

✓ Hiển thị thông tin của hai mô hình

```
print(">>>> ModelA intercept: {}, coefficient:
{}".format(modelA.intercept, modelA.coefficients[0]))
print(">>>> ModelB intercept: {}, coefficient:
{}".format(modelB.intercept, modelB.coefficients[0]))
```

✓ Hiển thị thông tin của hai mô hình

```
predictionsA = modelA.transform(data)
display(predictionsA)
predictionsB = modelB.transform(data)
display(predictionsB)
```

Bài tập 3: Đánh giá mô hình dự đoán

✓ Thêm lớp RegressionEvaluator từ thư viện máy học

```
from pyspark.ml.evaluation import RegressionEvaluator
```

✓ Sử dụng phương pháp tính Root Mean Squared Error để đánh giá kết quả mô hình

```
evaluator = RegressionEvaluator(metricName="rmse")
RMSEA = evaluator.evaluate(predictionsA)
print("ModelA: Root Mean Squared Error = " + str(RMSEA))
RMSEB = evaluator.evaluate(predictionsB)
print("ModelB: Root Mean Squared Error = " + str(RMSEB))
```

Bài tập 4: Trực quan hóa kết quả

✓ Thêm vào các thư viện hỗ trợ

```
import numpy as np
from pandas import *
from ggplot import *
```

✓ Lấy ra các thông tin cần thiết

```
pop = data.map(lambda p: (p.features[0])).collect()
price = data.map(lambda p: (p.label)).collect()
predA = predictionsA.select("prediction").map(lambda r:
r[0]).collect()
predB = predictionsB.select("prediction").map(lambda r:
r[0]).collect()
```

✓ Hiển thị trực quan kết quả

```
pydf = DataFrame({'pop':pop,'price':price,'predA':predA,
'predB':predB})
pydf

p = ggplot(pydf, aes('pop','price')) +
geom_point(color='blue') +
geom_line(pydf, aes('pop','predA'), color='red') +
geom_line(pydf, aes('pop','predB'), color='green') +
scale_x_log10() + scale_y_log10()
display(p)
```

~ HẾT ~