‹ Return to Classroom

DISCUSS ON STUDENT HUB

Evaluate Human Balance with Spark Streaming

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Hope this message finds you well.

Congratulations on completing the project! You should be very proud of your accomplishments. We cherish your feedback (by that I mean bragging rights among colleagues :D) and also your feedback of our review helps us be a small part of our students' success. Please stay safe and keep learning

Initialize Spark Components

The log `/home/workspace/spark/logs/spark-[username]-org.apache.spark.deploy.worker.Worker-1-[hostname].out` contains a statement “Successfully registered with master spark”, like this:

```
20/09/12 22:19:15 INFO Worker: Successfully registered with master spark:
//Seans-MBP:7077
```

Worker logs provide evidence of successful registration with master

The log `/home/workspace/spark/logs/kafkajoin.log` does not have any Python, Java, or Scala errors.

No Python, Java, or Scala errors are present in the log file

Continuous Application

As evidence, provide at least two screenshots of the STEDI risk graph showing a change in the data captured. The graph should have different data points.

Awesome !! two screenshots of the STEDI risk graph showing a change in the data captured is provided

The sink statement at the end of the sparkpykafkajoin.py script uses the proper method to ensure that the application runs continuously (until either killed or the computer shuts down).

`.awaitTermination()` is used to ensure that the application runs continuously

Consume and Process Data

A streaming dataframe is instantiated in the `sparkpykafkajoin.py` script with the redis-server Kafka topic as a source.

`redis-server` is consumed as a kafka topic

A streaming dataframe is instantiated in the `sparkpykafkajoin.py` script with the stedi-events Kafka topic as a source.

`stedi-events` is consumed as a kafka topic

The Python script `sparkpykafkajoin.py` properly invokes the appropriate function to parse JSON and decode Base64 encoded data from Kafka.

Appropriate function to parse JSON and decode Base64 encoded data is used.

Aggregation of Streaming Dataframes

The two distinct streaming dataframes joined for the aggregate streaming dataframes contain the following:

emailAndBirthYear: Email and Birth Year
emailAndRiskScore: Email and Risk Score

Neat !! two distinct streaming dataframes joined for the aggregate streaming dataframes

In the Python script `sparkpykafkajoin.py`, the streaming dataframe on which `.join()` is invoked contains information about either the customer risk or the customer record. The streaming dataframe passed inside the `.join()` function call contains information about either the customer risk or the customer record including the birth year.

The Python script `sparkpykafkajoin.py` calls the appropriate function to join two separate dataframes to create riskScoreByBirthYear.

`.join()` is used to two separate dataframes

Sink Datastream

In the Python script `sparkpykafkajoin.py`, the application sinks a streaming dataframe to a Kafka topic which is defined in the `/home/workspace/stedi-application/application.conf` file (example below):

```
kafka{
  riskTopic=customer-risk
}
```

Meets Specs !!

In the Python script `sparkpykafkajoin.py`, the appropriate function is called to sink the riskScoreByBirthYear streaming dataframe to Kafka in JSON format below:

```
{
"customer":"Santosh.Fibonnaci@test.com",
"Score":"28.5",
"email":"Santosh.Fibonnaci@test.com",
"birthYear":"1963"
}
```

Meets Specs !!

Validation

The Python script `sparkpyeventskafkastreamtoconsole.py` sinks a streaming dataframe to the console which contains customer risk information in a human-readable format. (Example below):

```
+--------------------+-----
|customer            |score|
+--------------------+-----+
|Spencer.Davis@tes...| 8.0|
+--------------------+-----
```

Meets Specs !!

The Python script `sparkpyrediskafkastreamtoconsole.py` sinks a streaming dataframe to the console which contains customer birth year information in a human-readable format. (Example below):

```
+--------------------+-----
| email          |birthYear|
+--------------------+-----
|Gail.Spencer@test...|1963|
|Craig.Lincoln@tes...|1962|
|  Edward.Wu@test.com|1961|
|Santosh.Phillips@...|1960|
|Sarah.Lincoln@tes...|1959|
|Sean.Howard@test.com|1958|
|Sarah.Clark@test.com|1957|
+--------------------+-----
```

Meets Specs !!

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START