OBDX JMS Logs: Analysis and Fine-Tuning Guide

Overview

This guide provides a practical solution for fine-tuning a machine learning model to analyze JMS logs from an OBDX (Oracle Banking Digital Experience) framework-based digital banking application. The goal is to train the model to read log files, extract key parameters, and generate insights, such as the number of successful, failed, and timed-out transactions.

Understanding JMS Logs

In the OBDX application, JMS (Java Message Service) logs contain the request and response details between the OBDX application and the core banking system. A transaction in this context refers to a request sent from the OBDX application to the core system, and the corresponding response from the core system.

Key Log Components

1. **Request**: Sent from OBDX to the core system.
    - **RequestTime**: Timestamp when the request was sent.
    - **UserID**: User ID initiating the transaction.
    - **ECID**: Execution context ID for tracing.
    - **ReferenceNo**: Unique identifier for the transaction.
    - **RequestType**: Type of request (interface ID).
    - **RequestBody**: Contains the details of the transaction.
2. **Response**: Sent from the core system to OBDX.
    - **ResponseTime**: Timestamp when the response was received.
    - **Status**: Indicates the outcome (SUCCESS, FAILURE, or TIMEOUT).
    - **ReferenceNo**: Matches the reference number in the request.
    - **ErrorCode**: Error code in case of failure.
    - **ErrorMessage**: Error message in case of failure.
    - **ResponseBody**: Contains the response details.

Identifying Corresponding Requests and Responses

A response can be matched to its corresponding request using the following parameters:

- **UserID**: Must be the same in both request and response.
- **ECID**: Must match in both request and response.
- **ReferenceNo**: Must be the same in both request and response.

Example Log Entry

Request Example

```
plaintext
Copy code
[2022-03-30T10:36:15.441+05:00] [OBDXManagedServer1] [TRACE] [] [JMSEndpoint]
[tid: '128'] [userId: Abdulwaheedryk] [ecid:
005qqZgZ8HqE8T0pzwl3iW000D6l000z3Q,0:1] [APP: obdx.app.rest.idm] [SRC_METHOD:
sendMessage] [[
Sending request to jms queue.
Request:
<?xml version="1.0" encoding="UTF-8"?>
<PartyToTermDepositAccountRelationRequest>
   <referenceNo>PI1741473206155702</referenceNo>
   <interfaceId>PARTY_TD_ACCOUNTS_LIST</interfaceId>
   <entity>OBDX_BU</entity>
   <headersMap/>
   <queryMap/>
   <accountType>TRD</accountType>
</PartyToTermDepositAccountRelationRequest>
]]
```

Response Example

```
plaintext
Copy code
[2022-03-30T10:36:15.469+05:00] [OBDXManagedServer1] [TRACE] [] [JMSEndpoint]
[tid: '128'] [userId: Abdulwaheedryk] [ecid:
005qqZgZ8HqE8T0pzwl3iW000D6l000z3Q,0:1] [APP: obdx.app.rest.idm] [SRC_METHOD:
processRequest] [[
Response received from jms queue.
Response:
<AccountListResponse xmlns:datatype="http://datatype.fc.ofss.com">
   <referenceNo>PI1741473206155702</referenceNo>
   <result>
      <status>SUCCESS</status>
      <errorList>
         <code></code>
         <message></message>
      </errorList>
      <warningList>
         <code></code>
         <message></message>
      </warningList>
   </result>
   <hasMore>false</hasMore>
   <startSequence>0</startSequence>
   <totalRecords>0</totalRecords>
</AccountListResponse>
]]
```

Key Paramters

- **Request Parameters**:
  - RequestTime: `[2022-03-30T10:36:15.441+05:00]`
  - UserID: `Abdulwaheedryk`
  - ECID: `005qqZgZ8HqE8T0pzwl3iW000D6l000z3Q`
  - ReferenceNo: `PI1741473206155702`
  - RequestType: `PARTY_TD_ACCOUNTS_LIST`
  - RequestBody: XML content of the request
- **Response Parameters**:
  - ResponseTime: `[2022-03-30T10:36:15.469+05:00]`
  - UserID: `Abdulwaheedryk`
  - ECID: `005qqZgZ8HqE8T0pzwl3iW000D6l000z3Q`
  - ReferenceNo: `PI1741473206155702`
  - Status: `SUCCESS`
  - ErrorCode (if FAILURE): `EP`
  - ErrorMessage (if FAILURE): `invalid Input Param: CifNo`
  - ResponseBody: XML content of the response
  - Duration: Difference between ResponseTime and RequestTime

Key Parameters to Extract from JMS Logs

From Request (Transaction)

1. **RequestTime**: The timestamp of the request.
2. **UserID**: The user ID associated with the request.
3. **ECID**: The execution context ID associated with the request.
4. **ReferenceNo**: The unique reference number of the request.
5. **RequestType**: The type of the request as indicated by the `interfaceId`.
6. **RequestBody**: The content of the request.

From Response

1. **ResponseTime**: The timestamp of the response.
2. **UserID**: The user ID associated with the response.
3. **ECID**: The execution context ID associated with the response.
4. **ReferenceNo**: The unique reference number of the response.
5. **Status**: The status of the response, which can be `SUCCESS`, `FAILURE`, or if no response is found, it is considered `TIMEOUT`.
6. **ErrorCode and ErrorMessage**: If the status is `FAILURE`, these fields will be present.
7. **ResponseBody**: The content of the response.

Matching Request and Response

- In case of large log files, find the response for each request based on the same **UserID**, **ECID**, and **ReferenceNo**. Always match all three parameters before declaring a response against a request.

- The time difference between **ResponseTime** and **RequestTime** will be the **Duration** of a request.
- If no response is found against a request in the JMS log file, the request (or transaction) status is considered `TIMEOUT`.
- Note that wherever this part appears it is useless its neither request nor response ignore it if found in the log file or chunk the part containg “Request sent on Queue successfully.”

  [2022-03-30T10:36:15.444+05:00] [OBDXManagedServer1] [TRACE] [] [000.com.ofss.digx.extxface.impl.endpoint.JMSEndpoint] [tid: [ACTIVE].ExecuteThread: '128' for queue: 'weblogic.kernel.Default (self-tuning)'] [userId: Abdulwaheedryk] [ecid: 005qqZgZ8HqE8T0pzwl3iW000D6l000z3Q,0:1] [APP: obdx.app.rest.idm] [partition-name: DOMAIN] [tenant-name: GLOBAL] [SRC_CLASS: com.ofss.digx.extxface.impl.endpoint.JMSEndpoint] [SRC_METHOD: processRequest] [[

  Request sent on Queue successfully.

  ]]

Overall, a request is also known as a transaction.

These are the essential parameters and instructions for processing JMS logs to extract and classify transactions.

Dataset Example

```
csv
Copy code
RequestTime,ResponseTime,UserID,ECID,ReferenceNo,RequestType,Status,ErrorCode
,ErrorMessage,Duration
2022-03-30T10:36:15.441+05:00,2022-03-
30T10:36:15.469+05:00,Abdulwaheedryk,005qqZgZ8HqE8T0pzwl3iW000D6l000z3Q,PI174
1473206155702,PARTY_TD_ACCOUNTS_LIST,SUCCESS,,,0.028
```

Conclusion

By following this guide, you can fine-tune your machine learning model to efficiently read, parse, and analyze OBDX JMS logs. This will help in generating actionable insights and improving the monitoring of your digital banking application's transactions.