Monash University

FIT5202 - Data processing for Big Data

Assignment 2B: Using real-time streaming data to predict potential customers

Due: Tuesday, Oct 18, 2022, 11:55 PM (Local Campus Time)
Worth: 10% of the final marks

Background

MonPG provides its loan services to its customers and is interested in selling more of its Top-up loan services to its existing customers. They hired us as the Analytics Engineer to develop a model to identify the potential customers that may have any Top Up services in the future. In addition, they want us to help them integrate the machine learning models into the streaming platform using Apache Kafka and Apache Spark Streaming to handle real-time data from the company to recommend our services.

In part A assignment, we only process the static data and form the machine learning model. **In this part B**, we would need to create proof-of-concept streaming applications to demonstrate the integration of the machine learning models, Kafka and Spark streaming, and create a visualization to provide some decision support.

File Structure

The files required for this assignment are available in moodle under Assessment 2B section. The description of the files is summarized in the table below:

File name	Description
bureau.csv	The bureau data includes the behavioral and transactional attributes of the customers, such as current balance, loan amount, overdue, etc., for various tradelines of a given customer. You can refer to this link for more details of the dataset: The original data is available on the website: https://www.kaggle.com/datasets/rizdelhi/analytics-vidya-ltfs-finhack-3?select=ltfs3_train_bureau.csv
customer.csv	The customer data contains variables related to basic service information. For example, frequency of the loan, tenure of the loan, disbursal amount for a loan & LTV.
metadata.pdf	The information about the dataset.

topup_pipeline_model.zip	The provided model “topup_pipeline_model” is a simplified version to predict whether customers are willing to join the Top-up services. - To use the model, please unzip the zip file and the resulting folder " topup_pipeline_model " should contain two subfolders - "metadata" and "stages". - You can put the " topup_pipeline_model " folder in the same directory of your notebook before loading the model into the Spark

What you need to achieve

The MonPG requires a proof-of-concept application to ingest the new data and predict the new top-up customers. To achieve this, you need to simulate the streaming data production using Kafka, and then build a streaming application that ingests the data and integrates the machine learning model (provided to you) to monitor the number of possible real-time top-up services.

A compulsory interview would also be arranged in Week 12 after the submission to discuss your proof-of-concept application.

Architecture

The overall architecture of the assignment setup is represented by the following figure.

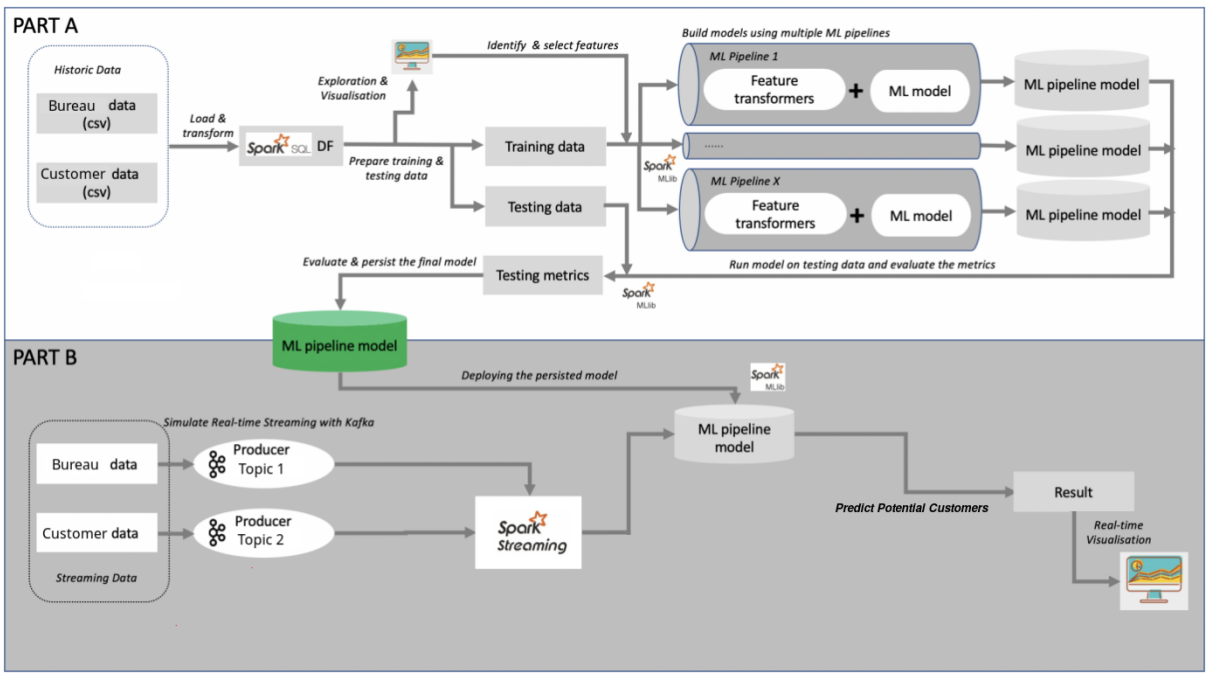

Fig 1: Overall architecture for assignment 2 (part B components updated)

In Part B of assignment 2, you have three main tasks - producing streaming data, processing the streaming data, and visualizing the data.

1. In task 1 for producing the streaming for both data files, you can use the CSV module or Pandas library, or other libraries to read and publish the data to the Kafka stream.
2. In task 2 for streaming data application, you need to use Spark Structured Streaming together with PySpark ML / DataFrame to process the data streams.
3. For task 3, you can use the CSV module or Pandas library, or other libraries to read the data from the Kafka stream and visualize it.

Please follow the steps to document the processes and write the codes in Jupyter Notebook.

Getting Started

- Download the data and models from moodle.
- Create an ***Assignment-2B-Task1_producer.ipynb*** file for data production
- Create an ***Assignment-2B-Task2_spark_streaming.ipynb*** file for consuming and processing data using Spark Structured Streaming
- Create an ***Assignment-2B-Task3_consumer.ipynb*** file for consuming the count data using Kafka

IMPORTANT:

- Please answer each question using BOTH codes and notebook markdown descriptions.
- In-line reference is required to acknowledge any ideas or codes that you referenced from others, or no marks would be awarded.
- Please do not print out an excessive amount of output in notebook cells

1. Producing the data (10%)

In this task, we will implement **one** Apache Kafka producer to simulate the real-time data transfer from one repository to another.

Important:

- **Do not use Spark in this task**
- **In this part, all columns should be string type**

Your program should send a random number (10~30, including 10 and 30) of client data every 5 seconds to the Kafka stream in 2 different topics based on their origin files.

- For example, if the first random batch of customers' IDs is 1,2, and 3, you should also send bureau data of them to the bureau topic. For every batch of data, you need to add a new column 'ts', the current timestamp. The data in the same batch should

have the same timestamp.
For instance: batch1: [{ID=xxx,...,ts=123456}, {ID=xxx,...,ts=123456},.....]
↑
one row

Save your code in ***Assignment-2B-Task1_producer.ipynb***.

2. Streaming application using Spark Structured Streaming (55%)

In this task, we will implement Spark Structured Streaming to consume the data from task 1 and perform predictive analytics.

Important:

- **In this task, use PySpark Structured Streaming together with PySpark Dataframe APIs and PySpark ML**
- **You are also provided with a pre-trained pipeline model for predicting the top-up customers. Information on the required inputs of the pipeline model can be found in the Background section.**

1. Write code to SparkSession is created using a SparkConf object, which would use two local cores with a proper application name, and use UTC as the timezone.
2. Use the same topic names from the Kafka producer in Task 1, ingest the streaming data into Spark Streaming and assume all data coming in String format.
3. Then the streaming data format should be transformed into the proper formats following the metadata file schema, similar to assignment 2A. Then use 'ts' column as the watermark and set the delay threshold to 5 seconds.
4. Group the bureau stream based on ID with 30 seconds window duration, similar to assignment 2A(same rule for sum and dist).
   - Transform the “SELF-INDICATOR” column’s values. If the value is true, then convert to 1, if the value is false, then convert to 0.
   - sum the rows for numeric type columns, count distinct values for other columns with other data types, and rename them with the postfix like '_sum' or '_dist'. (For example, we did the sum function based on the 'HIGH CREDIT', and the new column’s name will be 'HIGH CREDIT_sum').
5. Create new columns named 'window_start' and 'window_end' which are the window’s start time and end time in 2.4. Then inner join the 2 streams based on 'ID', and only customer data received between the window time are accepted.
   For example, customer data ID '3' received at 10:00, and only when the window of corresponding bureau data contains 10:00(like window start: 9:59, end: 10:00), then this data is accepted.
6. Persist the above result in parquet format.(When you save the data to parquet format,you need to rename “Top-up Month” to “Top-up_Month” first. And only keep these columns “ID”, “window_start”, “window_end”, “ts”, “Top-up_Month”) **Renaming “Top-up Month” only happen in this question**
7. Load the machine learning models given and use the model to predict whether users will be joining the top-up service. Save the results in parquet format. (When you save the data to parquet format,you need to rename “Top-up Month” to “Top-up_Month” first. And only keep these columns “ID”, “window_start”, “window_end”, “ts”, “prediction”, “Top-up_Month”) **Renaming “Top-up Month” will happen in this question as well**

8. Only keep the customer predicted as our target customers (willing to join the top-up service). Normally, we should only keep “Top-up=1”. But due to the limited performance of our VM, if your process is extremely slow, you can abandon the filter and keep all of the data. Then for each batch, show the epoch id and count of the dataframe.
   If the dataframe is not empty, transform the data to the following key/value format, which key is 'window_end' column and the data are the numbers of top-up services customers in the different states(in JSON format). Then send it to Kafka with a proper topic. These data will be used for the real-time monitoring in task 3.

e.g.:
|key|value|
|”2022-08-2609:06:00”|[{"State":"GUJARAT","count":3},{"State":"MADHYA PRADESH","count":10},{"State":"TELANGANA","count":5},{"State":"MAHARASHTRA","count":4},{"State":"KARNATAKA","count":1},{"State":"CHATTISGARH","count":1},{"State":"RAJASTHAN","count":8},{"State":"UTTARAKHAND","count":3},{"State":"UTTAR PRADESH","count":4},{"State":"HARYANA","count":5},{"State":"PUNJAB","count":2},{"State":"ANDHRA PRADESH","count":7},{"State":"WEST BENGAL","count":3},{"State":"ORISSA","count":2}]|

Save your code in ***Assignment-2B-Task2_spark_streaming*.ipynb**.

3. Consuming data using Kafka (15%)

In this task, we will implement an Apache Kafka consumer to consume the data from task 2.8.

Important:

- **In this task, use Kafka consumer to consume the streaming data published from task 2.8.**
- **Do not use Spark in this task**

Your program should consume the latest data and display them on the map(only for the latest data, abandon the old data). The map should be refreshed whenever a new batch of data is consumed. And the number of top-up customers of each state should be shown on maps (like a Choropleth map or heatmap with legend).

- Hint - you can use libraries like Plotly or folium to show data on a map, please also provide instructions on how to install your plotting library.

Save your code in ***Assignment-2B-Task3_consumer.ipynb***.

Interview (20%)

IMPORTANT: The interview is compulsory for this assignment. No marks will be awarded if the interview is not attended. For the online students, Camera, Mic, and Screen Sharing must be working. For the on-campus students, please bring your student ID card to attend the interview.

Assignment Marking

The marking of this assignment is based on the quality of work that you have submitted rather than just quantity. The marking starts from zero and goes up based on the tasks you have successfully completed and their quality for example how well the code submitted follows *programming standards, code documentation, presentation of the assignment, readability of the code, reusability of the code, organization of code, and so on*. Please find the PEP 8 -- Style Guide for Python Code here for your reference.

An interview would also be required for demonstrating your knowledge and understanding of the assignment. The interview would be run during the week 12 lab, where an audio + camera connection is required for the online students.

Submission

You should submit your final version of the assignment solution online via Moodle; You must submit the following:

- A PDF file containing all codes from the 1st, 2nd, and 3rd notebook to be submitted through the Turnitin submission link
    - Use the browser's print function (NOT Microsoft print) to save each notebook as PDF,
- A zip file named based on your authcate name and student_id (e.g. glii0039_30548470). And the zip file should contain
    - ***Assignment-2B-Task1_producer*.ipynb**
    - ***Assignment-2B-Task2_spark_streaming*.ipynb**
    - ***Assignment-2B-Task3_consumer.ipynb***

  This should be a ZIP file and *not any other kind of compressed folder (e.g. .rar, .7zip, .tar).* Please do not include the data files in the ZIP file.
- The assignment submission should be uploaded and finalised by Tuesday, Oct 18, 2022, 11:55 PM (Local Campus Time).
- Your assignment will be assessed based on the contents of the Assignment 2 folder you have submitted via Moodle. When marking your assignments, we will use the same ubuntu setup (VM) as provided to you.

Other Information

Where to get help

You can ask questions about the assignment on the Assignments section in the Ed Forum accessible from the on the unit's Moodle Forum page. This is the preferred venue for assignment clarification-type questions. It is not permitted to ask assignment questions on commercial websites such as StackOverflow or other forms of forums.

You should check the Ed forum regularly, as the responses of the teaching staff are "official" and can constitute amendments or additions to the assignment specification. Also, you can visit the consultation sessions if the problem and the confusions are still not solved.

Plagiarism and collusion

Plagiarism and collusion are serious academic offenses at Monash University. Students must not share their work with any other students. Students should consult the policy linked below for more information.

https://www.monash.edu/students/academic/policies/academic-integrity

See also the video linked on the Moodle page under the Assignment block.
Students involved in collusion or plagiarism will be subject to disciplinary penalties, which can include:

- The work not being assessed
- A zero grade for the unit
- Suspension from the University
- Exclusion from the University

Late submissions

There is a **10% penalty per day including weekends** for the late submission.

Note: The assessment submitted more than 7 calendar days after the due date will receive a mark of zero (0) for that assessment task. Students may not receive feedback on any assessment that receives a mark of zero due to late-submission penalty.

ALL Special Consideration, including within the semester, is now to be submitted centrally. This means that students MUST submit an online Special Consideration form via Monash Connect. For more details, please refer to the **Unit Information** section in Moodle.