Monash University

FIT5202 - Data processing for Big Data

Assignment 2B: Using real-time streaming data to predict retail sales

Due: Wednesday, Feb 8, 2023, 11:55 PM (Local Campus Time)
Worth: 10% of the final marks

Background

MelbourneGig is a start-up incubated at Monash University to provide services to performers in the retail industry. The team would like to hire us as the Analytics Engineers. The job includes:

- Analysing the retail sales data using big data tools.
- Developing machine learning models to predict future sales.
- Integrating the models into the streaming platform using Apache Kafka and Apache Spark Streaming to perform prediction to predict the real-time sales data in the future.

In part A of the assignment, we have already developed the machine learning models. **In this part B**, we need to create proof-of-concept streaming applications to demonstrate the integration of the machine learning model, Kafka, and Spark streaming and create a visualisation to predict future sales.

Available files in Moodle:

- CSV files:
  - produce_data.csv
  - stores.csv
- A zip file:
  - sales_estimation_pipeline_model.zip
- A Metadata file that contains information about the datasets:
  - metadata.pdf

Detailed Information of Files

1. The stores.csv is the same data from A2A.
2. The **produce_data.csv** is a combined version of the original **features.csv** and **sales.csv**. In this file (produce_data.csv), there is one more column compared with **features.csv**, namely *last_weekly_sales* representing the current store's last weekly sales. The original data is available on the website https://www.kaggle.com/datasets/manjeetsingh/retaildataset.

3. The provided model, **sales_estimation_pipeline_model**, is a simplified version to predict the current weekly sales of the store.
    a. To use the model, please unzip the zip file, and the resulting **sales_estimation_pipeline_model** folder should contain two subfolders - **metadata** and **stages**.
    b. You can put the folder **sales_estimation_pipeline_model** in the same notebook directory before loading the model into Spark.

What you need to achieve

The MelbourneGig company requires a proof-of-concept application to ingest the new count data and predict the potential sales data. 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 the weekly sales to predict the current week's sales.

A compulsory interview would also be arranged in Week 6 (session 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.

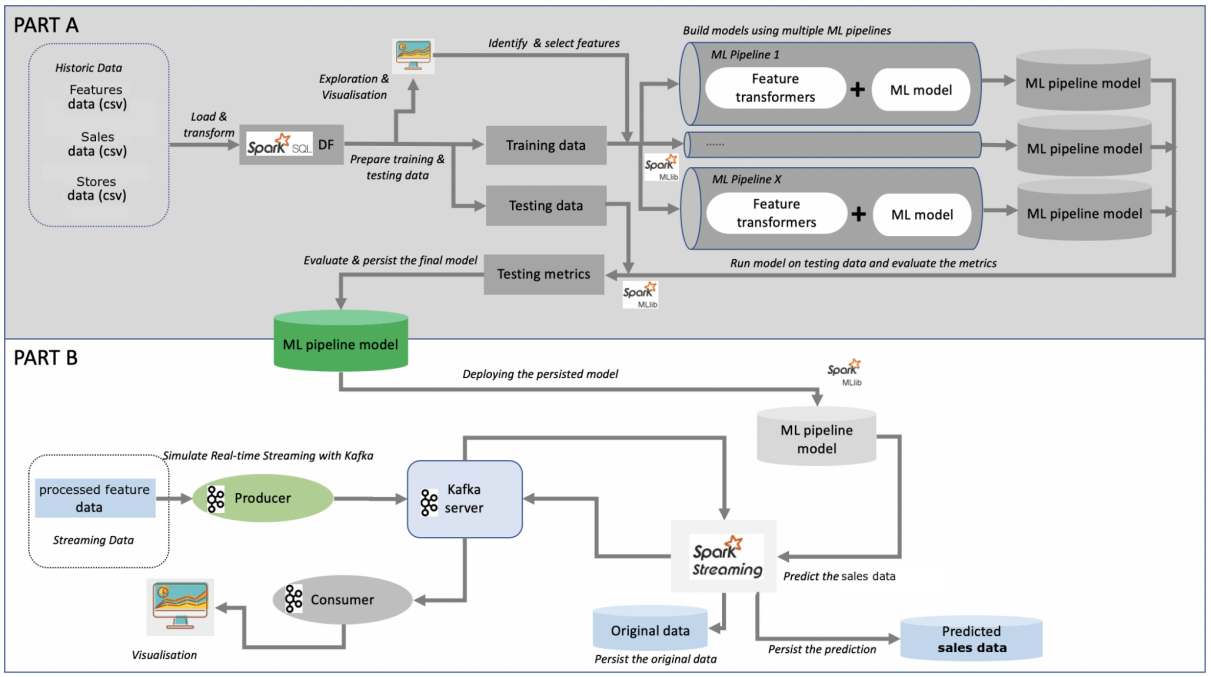


Fig 1: Overall architecture for assignment 2 (part B components updated)

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

1. In task 1 for producing the streaming data for stores in 2011, you can use 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 either csv module, Pandas library, or other libraries to read the data from the Kafka stream and visualise it.

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

Getting Started

- Download the data and models from moodle.
- Download three ipynb template files from moodle
    - ***A2B-Task1_username.ipynb*** file for data production
    - ***A2B-Task2_username.ipynb*** file for consuming and processing data using Spark Structured Streaming
    - ***A2B-Task3_username.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
There is **no documentation task** in this assignment A2B, as the **interview** will replace it in session 12
The interview is compulsory as part of A2B. Zero mark for A2B if the student did not involve in the interview.

1. Producing the data (8%)

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.**

Your program should send one batch of all store's data at one day in 2011 records every 5 seconds to the Kafka stream.
- For example, for the first batch of data transmission, your program should send all stores' data captured on the first date (1/7/2011) in the **produce_data.csv** to the Kafka stream; after 5 seconds, your program should send all stores' data on the next week (1/14/2011) to the Kafka stream, and so on.
- You should send data with time order in 2011. After sending the last date's data in 2011, restart from the first date (1/7/2011).
- Inside the **produce_data.csv,** it has a new **last_weekly_sales** column which represents the store's sales in the last week compared with **feature.csv** in A2a, which will be used in our future prediction.

- For each row, add a timestamp column named *ts* which is real-time when you send the streaming data, which should be in *int* format. Their *ts* should be the same for the data sent in one batch (every 5 seconds).
- All the data except for the 'ts' column should be sent in the original *string* format without changing to any *datetime* format.

- Save your code in ***A2B-Task1_username.ipynb***.

2. Streaming application using Spark Structured Streaming (40%)

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 current week's sales data and persist the prediction.**

1. Write code to SparkSession is created using a SparkConf object, which would use four local cores with a proper application name and also make sure a checkpoint location has been set.

2. Similar to assignment 2A, write code to define the data schema for the store.csv file, following the data types suggested in the metadata file.

3. Using the same topic name from the Kafka producers in Task 1, ingest the streaming data into Spark Streaming, assuming all data coming in the **String** format. Except for the 'ts' column, you can receive it as a **Long** type.

4. Persist the raw streaming data in parquet format. After that, read the parquet result and show the results. (You can stop the parquet streaming after you showed the results)

5. Then the streaming data format should be transformed into the proper formats following the metadata file schema, similar to assignment 2A. For the 'ts' column, transfer them to the timestamp format.

6. As the purpose of the recommendation is to predict the current week's sales data, write code to perform the following transformations to prepare the columns for model prediction.
    a. Create the column named "Month" based on the column "Date" to represent the month of the date
    b. Create the column named "day_of_month" based on the column "Date" to represent the day of the month of the date
    c. Create the column named "day_of_year" based on the column "Date" to represent the day of the year of the date

d. Create the column named “week_of_year” based on the column “Date” to represent the week of the year of the date

7. Join the **stores** dataframe with our (stream) dataframe from Q2.6 as our final data for prediction. Then print out the Schema.
8. Load the machine learning models given, and use the model to predict the week’s sales. Persist the prediction result in parquet format, then read the parquet result and show the results. (You can stop the parquet streaming after you showed the results)
9. Using the prediction result, write code to process the data following the requirements below.
    a. Show how many stores of different store types achieved the goal (weekly sales divided by the store size greater than 8.5) in every 10 seconds.
    b. The slide duration of the window should be 5 seconds.
    c. Abandon the data received later after 3 seconds.
    d. Do not show the empty data/dataframe; only show the count when there is the data.
    e. Process each batch of data every 5 seconds. For example, it should be like the following screenshot. (You can stop this stream after some results were showed)

```
+------------------------------------------+----+-----+
|window                                    |Type|count|
+------------------------------------------+----+-----+
|{2023-01-11 16:30:25, 2023-01-11 16:30:35}|A   |5    |
|{2023-01-11 16:30:25, 2023-01-11 16:30:35}|B   |7    |
|{2023-01-11 16:30:25, 2023-01-11 16:30:35}|C   |5    |
+------------------------------------------+----+-----+
```

10. Find the average weekly sales predictions of different types of stores and write the stream back to Kafka sink using a different topic name.
    a. Use the same 'window rule' in Q9 (every 10 seconds, slide duration should be 5 seconds, and abandon the data later after 3 seconds.)
    b. Use the start time of the window as the 'key' column; the time should be in Unix timestamp format.
    c. The 'value' column should be in JSON format.
    d. The data format should be like this:

key	value
timestamp of window start	JSON of store type and avg sales
'1673233646'	'{"Type":"A","predict_weekly_sales":20000}'

Save your code in ***A2B-Task2_username*.ipynb**.

3. Consuming data using Kafka (12%)

In this task, we will implement an Apache Kafka consumer to consume the data from task 2.10.

Important:

- **In this task, use Kafka consumer to consume the streaming data published from task 2.10.**
- **Do not use Spark in this task.**

Draw a line chart using the data you received. Use the timestamp (the key of the data you received) as the x-axis and the average weekly sales of each type of store as the y-axis. The plot should be updated after each time you receive a new batch of data.

Save your code in ***A2B-Task3_username.ipynb***.

Interview (40%)

IMPORTANT: Interview is compulsory. No marks will be awarded if the interview is not attended. For the Interview, Camera, Mic and Screen Sharing must be working.

Assignment Marking

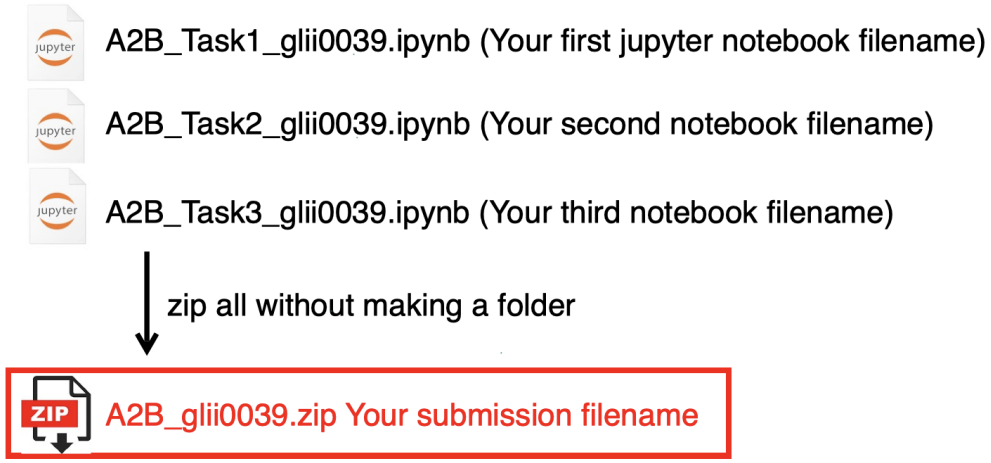
The marking of this assignment is based on the quality of work you have submitted rather than just quantity. The marking starts from zero and goes up based on the tasks you have successfully completed and their quality, for example, how well the code submitted follows *programming standards, code documentation,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

An interview would also be required to demonstrate your knowledge and understanding of the assignment. The interview would be run during the session 12 lab. For the online classes, an audio + camera connection are required.

Submission

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

- The assignment submission should be uploaded and finalised by Wednesday, Feb 8, 2023, 11:55 PM (Local Campus Time).
- Your assignment will be assessed based on the zip file you submitted via Moodle. When marking your assignments, we will use the Docker environment as provided to you.
- The submission is in the form of **Zip file** of your three jupyter notebook files. A zip file named based on your authcate name (e.g., glii0039). And the zip file should contain
    - A2B-Task1_glii0039.ipynb
    - A2B-Task2_glii0039.ipynb
    - A2B-Task3_glii0039.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.
  Below is the illustration of the zip process:



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 offences 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 **5% penalty per day including weekends** for the late submission.

Note: Assessment submitted more than 4 calendar days after the due date will receive a zero (0) mark for that assessment task. Students may not receive feedback on any assessment that receives a mark of zero due to a 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.