Monash University

FIT5202 - Data processing for Big Data 2025 S2

Assignment 2B: Using Streaming and ML for Building Energy Consumption Prediction and Visualisation

Due: **(23:55 Friday 24/Oct/2025 End of Week 12)**

Weight: 15%

Background (The same as A2A)

Accurate energy consumption forecasting is critical for modern power grids. It enables utility companies to balance supply and demand, prevent outages, and integrate renewable energy sources more effectively. For consumers, understanding energy usage patterns can lead to significant cost savings and a reduced carbon footprint.

The way we get and use electricity is changing in a big way. For a long time, large power plants made electricity and sent it to our homes. This was a one-way street. Now, we have new goals. We need our power to be:

1. **Reliable:** We want to have electricity whenever we need it.
2. **Affordable:** We want the cost of power to be fair and not too expensive.
3. **Clean:** We need to use more clean energy, like solar and wind, to protect the environment.

To meet these goals, we are building a "Smart Grid." A smart grid is a modern power system that utilises computers and the internet to enhance its efficiency. A key part of this system is the **smart meter**. These devices are in people's homes, and they measure how much electricity is being used. They send this information back to the power company very often. This creates enormous amounts of data.

This data is very useful. By studying it, power companies can predict how much electricity people will need at any given time (**energy forecasting)**. Good forecasting is essential for several reasons:

- **Making the Right Amount of Power:** It helps companies make just enough electricity to meet everyone's needs, without wasting any or running short.
- **Using Clean Energy:** Power from the sun and wind can change a lot. Forecasting helps companies plan for these changes and use more clean energy.
- **Preventing Problems:** It helps companies manage busy "peak hours" when everyone is using a lot of power at once.

Big data processing enables us to analyse vast datasets from smart meters, weather sensors, and building characteristics, thereby building precise predictive models. By identifying key drivers of energy consumption—such as time of day, weather conditions, and appliance usage—we can forecast future demand with high accuracy.

In this assignment, you will assume the role of a data scientist working for a power company. You will use a real dataset that shows how much electricity different buildings use over time.

Your job will be to use **Apache Spark** to study this data and build a program. This program will learn from the old data to predict how much electricity a building will use in the future.

Key Information

This is a two-part assignment (A2A and **A2B**) that requires staged submissions. In part A2A, you are going to use the provided dataset, complete the assignment tasks, and build your ML model; then, in part A2B, the trained ML model will be used in combination with streaming data to make real-time predictions.

A2B Due Date: **(23:55 Friday 24/Oct/2025, End of Week 12)**

Submission links can be found in Moodle.

Weight: 30% of Final Marks (15% each for 2A and 2B) A2A and A2B will be marked separately.

A2B has a compulsory interview/demo component, which will be conducted in week 14. **The teaching team only marks A2B submissions during your demo session. Failure to attend this demo will result in 0 marks (for A2B).**

(Please pay attention to the unit announcement in the final teaching week. If you have an extension/special consideration, more demo sessions will be arranged after the exam.)

The Datasets:

- **weather.csv (From A2A)**
- **new_meters.csv**
- **new_building_information.csv**
- **Metadata is the same as A2A.**

What you need to achieve

A2B Use Case	Use streaming data to perform real-time prediction and visualise the results	Spark Structured Streaming

Architecture

The following figure represents the overall architecture of the assignment setup.

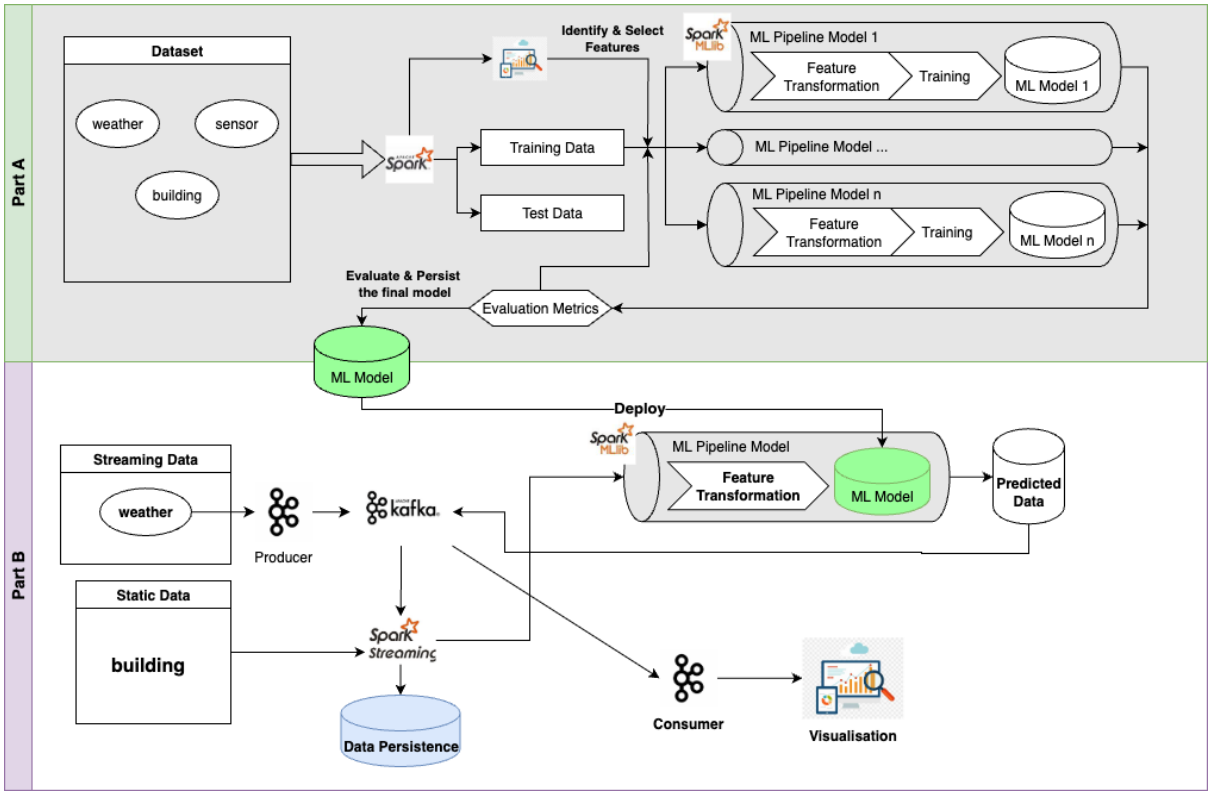

Fig 1: Overall Architecture for Assignment 2

In both parts, you must implement the solutions using PySpark DataFrame/MLlib for the data pre-processing and machine learning pipelines. Please follow the steps to document the processes and write the code in your Jupyter Notebook.
In **Part B,** you will utilise the model from Part A and generate operational insights for the grid operator.

Getting Started

- Download the datasets from Moodle.
- Download a template file for submission purposes:
    - ***A2B-Task1_producer.ipynb*** file for streaming data production
    - ***A2B-Task2_spark_streaming.ipynb*** file for consuming and processing data using Spark Structured Streaming
    - ***A2B-Task3_consumer.ipynb*** file for consuming the data using Kafka and visualising
    - For submission, please append your authcate ID at the end of the filename. ***(e.g. A2B-Task1_producer_xxxx0000.ipynb*** **xxxx0000** is your authcate ID.
- You will use Python 3+ and PySpark 3.5.0+ for this assignment (This environment is the same as we used in labs.)

IMPORTANT:

Please answer each question in your Jupyter Notebook file using code/markdown cells. Acknowledge any ideas or codes you referenced from others in the markdown cell or reference list.

If you use generative AI tools, all prompts you use should also be included in the reference section or appendix.

A2 Part B Specification

Your task in this application is to build a visualisation dashboard for the power grid operator to oversee buildings' and sites' energy consumption, so that they can optimise their power generation.

Part 1. Producing the data (10%)

In this task, we will implement Apache Kafka producers to **simulate** real-time data streaming. Spark and parallel data processing should not be used in this section, as we are simulating sensors that often lack processing capabilities.

1. Every 5 seconds, load 5 days of **weather** data from the CSV file. We refer to this as **weather5s** to explain the tasks; feel free to use your own variable name. You should keep a pointer in the file reading process and advance it per read. The data reading should be in chronological order.
2. Add the current timestamp (**weather_ts**) to the **weather5s** and spread your batch out evenly for 5 seconds for each day. Since the weather data is hourly readings, each day you shall have 24 records (120 records in total for 5 days).
   For example, assume you send the records at 2025-01-26 00:00:00 (ISO format: YYYY-MM-DD HH:MM:SS) -> (ts = 1737810000):
   Day 1(records 1-24): ts = 1737810000
   Day 2(records 25-48): ts = 1737810001
   Day 3(records 49-72): ts = 1737810002
   …
3. Send your batch of weather data to a Kafka topic with an appropriate name.

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

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

In this task, you will implement Spark Structured Streaming to consume the data from task 1 and perform a prediction.

Important:

- **This task uses PySpark Structured Streaming with PySpark Dataframe APIs and PySpark ML.**
- **You also need your pipeline model from A2A to make predictions and persist the results.**

1. Write code to create a SparkSession, which 1) uses **four cores** with **a proper application name**; 2) use the Melbourne timezone; 3) ensure a checkpoint location has been set.
2. Write code to define the data schema for the data files, following the data types suggested in the metadata file. Load the static datasets (e.g. building information) into data frames. (You can reuse your code from 2A.)
3. Using the Kafka topic from the producer in Task 1, ingest the streaming data into Spark Streaming, assuming all data comes in the **String** format. Except for the 'weather_ts' column, you shall receive it as an **Int** type. Load the new building information CSV file into a dataframe. Then, the data frames should be transformed into the proper formats following the metadata file schema, similar to assignment 2A.
4. Use a watermark on **weather_ts**, if data points are received 5 seconds late, discard the data.
5. Perform the necessary transformation you used in A2A. (note: every student may have used different features, feel free to reuse the code you have written in A2A. If you built an end-to-end pipeline, you can ignore this task.)
6. Load your pipeline model and perform the following aggregations:
    a) Print the prediction from your model as a stream comes in.
    b) Every 7 seconds, print the total energy consumption for each **6-hour interval**, aggregated by **building**, and print 20 records. (Note: This is simulating energy data each day in a week**)**
    c) Every 14 seconds, for each **site**, print the **daily** total energy consumption.
7. Save the data from 6 to Parquet files as streams. (Hint: Parquet files support streaming writing/reading. The file keeps updating while new batches arrive.)
8. Read the parquet files from task 7 as **data streams** and send them to Kafka topics with appropriate names.
   (Note: **You shall read the parquet files as a streaming data frame and send messages to the Kafka topic when new data appears in the parquet file.)**

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

Part 3. Consuming data using Kafka and Visualise (30%)

In this task, we will implement an Apache Kafka consumer to consume the data from Part 2.
Important:

- **In this part, Kafka consumers are used to consume the streaming data published from task 2.8.**

1. **Load the new meters CSV file into a data frame.**
2. Plot two diagrams to show data from 6b and 6c. You are free to choose the type of plot.
3. Plot a diagram to visualise the **daily** shortfall/excess energy in each **site**. The shortfall/excess energy is defined as the predicted total sum of energy in each site, minus the metered data (the value can be positive or negative, depending on the model and data quality). Again, you're free to choose the type of plot that is the best to represent this data.

You can refer to the following sites:
https://samplecode.link/**(**a mirror of **https://python-graph-gallery.com/**)

https://matplotlib.org/

https://plotly.com/python/

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

Part 4: Demo and Interview (20%)

> **IMPORTANT**: The interview is compulsory, and we only mark your A2B during the interview. No marks will be awarded if the interview is not attended. (0 marks for the whole A2B, not just this section).

The demo/interview session details will be announced/arranged in the last teaching week. Please pay attention to the unit announcement email and Ed forum.

Each demo is roughly 10 minutes. You have 5-6 minutes to show your application; then, your marker will ask 3-4 questions to assess your understanding. Please come to your allocated demo session a few minutes early and ensure your laptop/application works correctly.

Demo/Interview is marked on a 5-level scale:

The demo is working, and the student has a competent understanding	20
Working demo, partial understanding	15
The demo is not working, partial understanding	10
The demo is not working, low understanding	5
No attendance or can't answer most of the questions	0

For those students with special consideration and extension, more demo sessions will be arranged after the exam.

Submission A2B

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

- A **zip** file named based on your authcate name (e.g. abcd1234). The zip file should contain
  - ***Assignment-2B-Task1_producer_authcate*.ipynb**
  - ***Assignment-2B-Task2_spark_streaming_authcate*.ipynb**
  - ***Assignment-2B-Task3_consumer_authcate.ipynb***

  The file in submission 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 A2B due date is **23:55 Friday 24/Oct/2025 End of Week 12**

Assignment Marking Rubric

Detailed mark allocation is available in each task. For complex tasks and explanation questions, you will receive marks based on the quality of your work.

In your submission, the Jupyter Notebook file should contain the **code and its output**. It should follow *programming standards, readability of the code, and organisation of code*. Please find the PEP 8 -- Style Guide for Python Code for your reference. Here is the link: https://peps.python.org/pep-0008/ Penalty applies if your code is hard to understand with insufficient comments.

Other Information

Where to get help

You can ask questions about the assignment in the Assignments section in the Ed Forum, which is accessible on the unit's Moodle Forum page. This is the preferred venue for assignment clarification-type questions. You should check this forum regularly, as the responses of the teaching staff are "official" and can constitute amendments or additions to the assignment specification. Also, you can attend scheduled consultation sessions if the problem and the confusion are still unresolved.

Searching and learning on commercial websites/forums (e.g. Quora, Stack Overflow) is allowed. However, you should not post/ask assignment questions on those forums.

Plagiarism and collusion

Plagiarism and collusion are serious academic offences at Monash University. Students must not share their work with 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 is not being assessed
- A zero grade for the unit
- Suspension from the University
- Exclusion from the University

Late submissions and Special Consideration

ALL Special Consideration, including within the semester, is now handled 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.

There is a **5% penalty per day, including weekends,** for a late submission. Also, the cut-off date is 7 days after the due date. No submission will be accepted (i.e. zero mark) after the cut-off date unless you have a special consideration.

Mark Release and Review

- Mark will be released within 10 business days after the submission deadline.
- Reviews and disputes regarding the mark will be accepted a maximum of 7 days after the release date (including weekends).

Generative AI Statement

As per the University's policy on the guidelines and practices pertaining to the usage of Generative AI:

AI & Generative AI tools may be used SELECTIVELY within this assessment.

Where used, AI must be used responsibly, clearly documented and appropriately acknowledged (see Learn HQ).

Any work submitted for a mark must:

1. Represent a sincere demonstration of your human efforts, skills, and subject knowledge for which you will be accountable.
2. Adhere to the guidelines for AI use set for the assessment task.
3. Reflect the University's commitment to academic integrity and ethical behaviour.

Inappropriate AI use and/or AI use without acknowledgement will be considered a breach of academic integrity.

The teaching team encourages students to apply their own critical thinking and reasoning skills when working on the assessments with assistance from GenAI. Generative AI tools may produce inaccurate content, which could have a negative impact on students' comprehension of big data topics.

Appendix: Metadata of the Dataset Schema

This table contains the time-series data for energy consumption for each meter in each building.

Column Name	Data Type	Description
building_id	Integer	A unique identifier for the building where the meter is located. Foreign key to the buildings table.
meter_type	Char(1)	The type of energy being measured from 4 different type of meters. (e.g., 'e', ‘c’, 's', 'h').
ts	Timestamp	The exact timestamp of the meter reading.
value	Decimal	The energy consumption reading value. The aggregated value of 4 meters is target variable for prediction.
row_id	Integer	A unique identifier for each individual reading in the table.

2. Buildings Table (buildings.csv)

This table contains the static metadata and descriptive features for each building.

Column Name	Data Type	Description
site_id	Integer	An ID for the geographical location of the building. Foreign key to the weather table.
building_id	Integer	A unique identifier for the building.
primary_use	String	The primary function of the building (e.g., 'Education', 'Office', 'Retail').
square_feet	Integer	The gross floor area of the building in square feet.
floor_count	Integer	The number of floors in the building.
row_id	Integer	A unique identifier for each building record in the table.
year_built	Integer	The year the building was constructed.
latent_y	Decimal	A latent feature with unknown meaning.
latent_s	Decimal	A latent feature with unknown meaning.

latent_r	Decimal	A latent feature with unknown meaning.

3. Weather Table (weather.csv)

This table contains the time-series weather data for each geographical site.

Column Name	Data Type	Description
site_id	Integer	A unique identifier for the geographical location/site. Each site may reside in different country with different seasons.
timestamp	Timestamp	The timestamp of the weather measurement.
air_temperature	Decimal	The temperature of the air in degrees Celsius.
cloud_coverage	Integer	The portion of the sky covered by clouds (e.g., from 0-8 oktas). May contain missing values.
dew_temperature	Decimal	The dew point temperature in degrees Celsius.
sea_level_pressure	Decimal	The air pressure in millibars, adjusted to sea level. May contain missing values.
wind_direction	Integer	The direction of the wind in degrees from true north (0-360). May contain missing values.
wind_speed	Decimal	The speed of the wind in meters per second.