MGMT 467 | Final Project Report | Team 8 | Artemii Chirkov, Kanan Gurbanov, Brandon Moss

Architecture & Ops

This report documents the overall architecture of the project and how to spin up / tear down the data and modeling pipeline.

High-Level Architecture

At a high level, the system combines **batch data** from Kaggle with **streaming data** from a public earthquake API (https://www.kaggle.com/datasets/usgs/earthquake-database), then trains a BQML model and surfaces KPIs in Looker Studio.

1. Batch Path – Kaggle → GCS → BigQuery

- **Source**
    - Kaggle earthquakes CSV file from this link: https://www.kaggle.com/datasets/usgs/earthquake-database
- **Storage & raw table**
    - Raw file stored in GCS (e.g., gs://mgmt-467-471119-opensky/earthquakes_raw/).
    - Loaded into a raw BigQuery table mgmt-467-471119.database.earthquakes_raw.
- **Curation**
    - A SQL transformation builds mgmt-467-471119.database.earthquakes_batch which:
        - Casts magnitude, depth, latitude, and longitude to numeric types.
        - Normalizes timestamps into a single TIMESTAMP column.
        - Enforces simple data-quality rules (e.g. 0 ≤ depth_km ≤ 700, magnitude not NULL).
        - Partitions the table by date to support efficient queries.

2. Streaming Path – API → Cloud Function → Pub/Sub → Dataflow → BigQuery

- **Source**
    - Public earthquake API (e.g., USGS) returning JSON for recent events.
- **Cloud Function (2nd gen)**
    - Triggered on a schedule by Cloud Scheduler (e.g., every few minutes).
    - Calls the earthquake API and normalizes JSON into a payload with fields such as event_id, event_time, mag, depth_km, lat, lon, place, and source.
    - Publishes each event to a Pub/Sub topic, for example: earthquake-events.
- **Dataflow streaming job**
    - Consumes messages from the Pub/Sub topic.
    - Applies basic parsing / validation.
    - Writes to a BigQuery streaming table mgmt-467-471119.database.earthquakes_stream.
- **Parsed view for analytics**
    - A SQL view mgmt-467-471119.database.earthquakes_stream_parsed uses JSON_VALUE / SAFE_CAST to turn the JSON payload into typed columns.
    - This view feeds both:
        - The unified earthquakes_live table, and
        - The Looker Studio streaming dashboard.

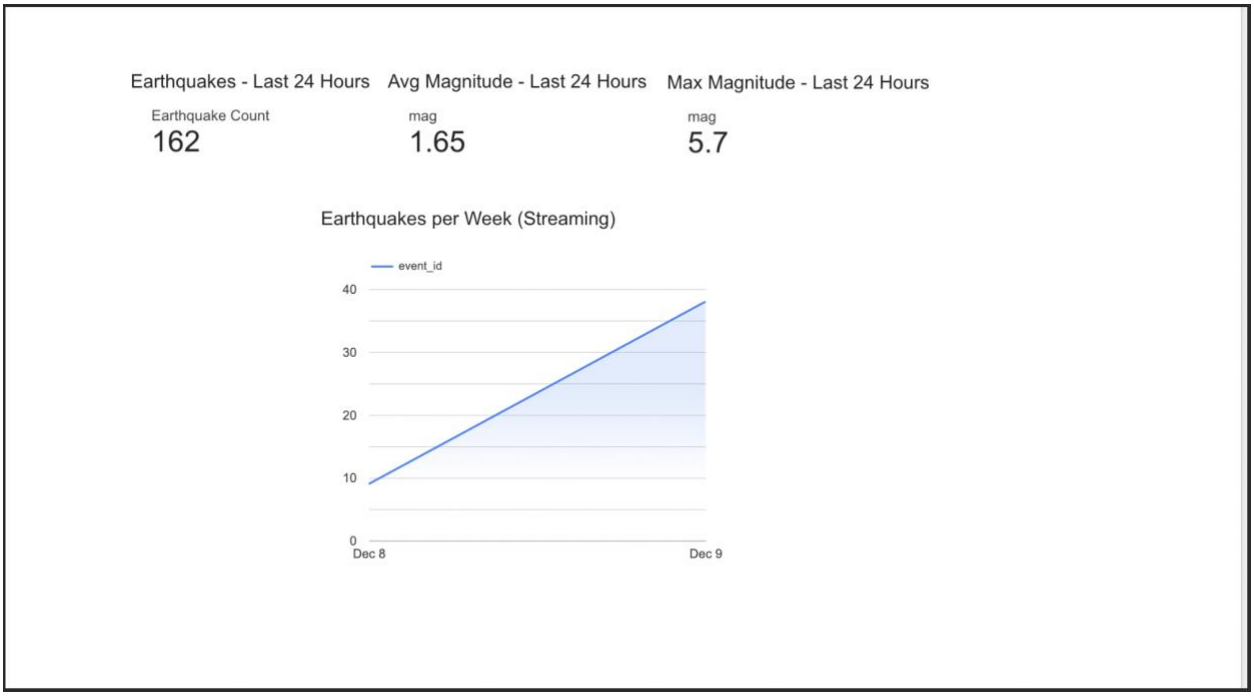
3. Unified Live Table & BQML Model

- **Unified table**
    - mgmt-467-471119.database.earthquakes_live combines:
        - Historical curated batch data from earthquakes_batch.
        - Recent streaming rows from earthquakes_stream_parsed.
    - Ensures the model sees a consistent schema across batch + streaming.
- **BQML model**
    - Logistic regression classifier mgmt-467-471119.database.severity_classifier.

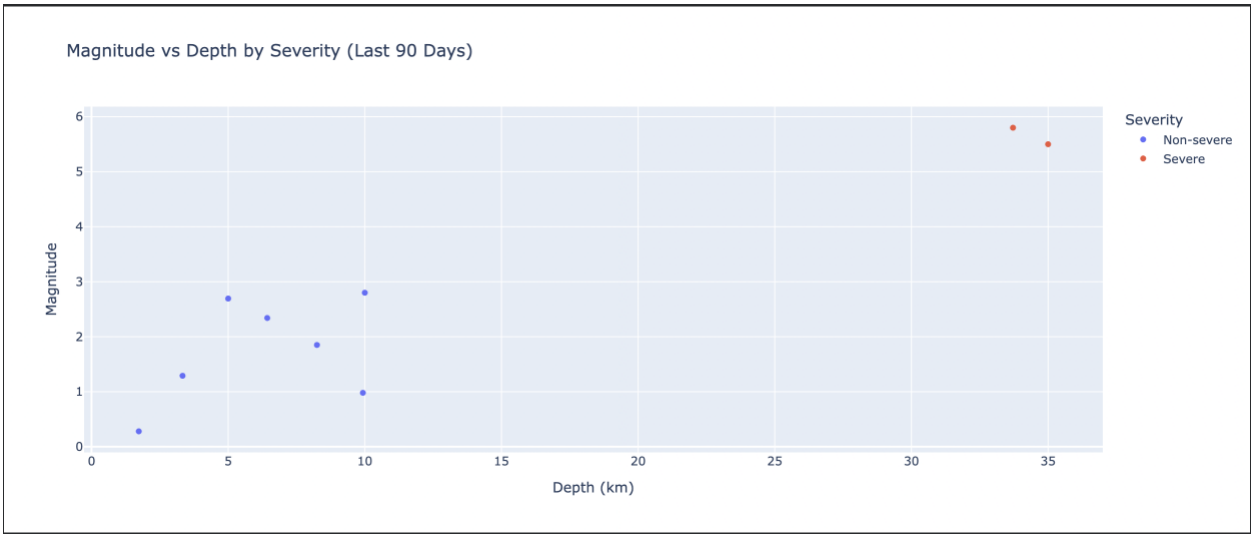
- Features: mag, depth_km, lat, lon.
- Label: label_severe = 1 if mag ≥ 5.5, else 0.
- Evaluation done via ML.EVALUATE, predictions via ML.PREDICT.

4. Analytics & Dashboards

- **Looker Studio dashboard**
  - Data source: earthquakes_stream_parsed.
  - KPIs (Last 24 Hours):
    - Earthquakes – COUNT(event_id),
    - Avg Magnitude – AVG(mag),
    - Max Magnitude – MAX(mag).
  - Time series:
    - "Earthquakes per Day (Streaming)" – count of events over time.



- **Notebook analytics**
  - Additional Plotly visualizations (e.g. magnitude vs depth by severity) for model interpretability and KPI design.



Reproducibility & Operations

This checklist explains how to **spin up** and **tear down** the pipeline in GCP.

Spin-Up Steps

1. **Enable core GCP APIs**
    - BigQuery API
    - Cloud Storage API
    - Cloud Functions API
    - Cloud Scheduler API
    - Pub/Sub API
    - Dataflow API
2. **Create storage and BigQuery datasets**
    - GCS bucket for raw data, e.g. gs://mgmt-467-471119-opensky.
    - BigQuery dataset database in project mgmt-467-471119.
3. **Batch ingestion**
    - Upload the Kaggle earthquakes CSV into the GCS bucket.
    - Load the file into database.earthquakes_raw.

- Run the SQL script that builds the curated, partitioned database.earthquakes_batch with:
  - Correct types,
  - Timestamp normalization,
  - Data-quality filters (e.g., depth and magnitude rules).

4. **Streaming ingestion**
   - Create Pub/Sub topic earthquake-events.
   - Deploy the 2nd-gen Cloud Function that:
     - Calls the earthquake API,
     - Publishes normalized JSON to the earthquake-events topic.
   - Configure a Cloud Scheduler job to trigger the function on a fixed schedule.
   - Start a Dataflow streaming job using the Pub/Sub → BigQuery template:
     - Input: earthquake-events topic,
     - Output: database.earthquakes_stream.
   - Verify that rows appear in database.earthquakes_stream_parsed with recent timestamps.
5. **Model training & scoring**
   - Run the CREATE OR REPLACE MODEL statement for database.severity_classifier.
   - Use ML.EVALUATE to check metrics.
   - Optionally schedule a query that periodically scores new events in earthquakes_live and writes predictions / probabilities to a table for further reporting.
6. **Dashboards & notebooks**
   - In Looker Studio, connect to mgmt-467-471119.database.earthquakes_stream_parsed as a data source and recreate the 3 KPIs + time series.

- Open this notebook and run all cells to regenerate tables, model evaluation, and Plotly visualizations.

Tear-Down Steps

1. **Pause streaming**
    - Stop or drain the Dataflow streaming job.
    - Disable or delete the Cloud Scheduler job.
    - Disable or delete the Cloud Function if you no longer want to call the API.
2. **Clean up GCP resources**
    - Optionally delete the Pub/Subtopic earthquake-events.
    - Optionally remove any temporary GCS staging buckets used by Dataflow / Cloud Functions.
3. **BigQuery cleanup (optional)**
    - If cost is a concern, drop or expire:
        - database.earthquakes_stream,
        - database.earthquakes_batch,
        - database.earthquakes_live,
        - Any temporary scoring tables.

Documenting these steps makes the project **reproducible** for another person (or a future version of ourselves) and shows how the end-to-end pipeline can be deployed and safely shut down.