Project 6: Evaluation Batch Job
CSC 6605 ML Production Systems

Overview
In this project, you are going to create a batch job that checks for new cleaned, joined, and labeled home sale events, make REST requests to the prediction service to get predicted sale prices, and deposits the results in a PostgreSQL table. You'll use a notebook to query the data, calculate evaluation metrics, and plot the metrics. This system is used to monitor how the model performs as the incoming data changes.

Instructions
Part I: Design Schema and Create Tables
Like the previous batch job, you will need two tables. One table will contain the home sale event id, sale date, and true and predicted prices. The second table will be used for storing ids of evaluated home sale event ids to maintain the state of the pipeline. Create the tables and indices in your database.

Part II: Design and Implement Batch Pipeline
In this project, you're going to implement an evaluation pipeline using the same pattern as your other batch pipeline. Your pipeline will:

1. Query for unprocessed raw home sale events
2. Make RESTful calls to the prediction service to get model predictions for the sale prices. (The requests library is useful for this.)
3. Insert the home sale event id, sale date, and true and predicted prices into a PostgreSQL table
4. Add the home sale event id to a table for recording processed events

As before, you should have a standalone batch job script. It should take the PostgreSQL credentials as environmental variables. You should also have a job runner script that uses APScheduler to run the batch job. And lastly, you need a Dockerfile to package the batch job as a Docker image. These files should be in an "evaluation-batch-job" directory in your GitHub repository. Make sure the directory has a README.md file documenting the schema, components, and how to run everything.

Part III: Deploy Pipeline
After you confirmed that your pipeline works, deploy it in a Docker container.

1. Build a Docker image.

2. Run the container on your group's virtual machine.

3. Check the running Docker containers.

4. Check PostgreSQL to ensure that entries are being added to the two tables you created.

Part IV: Notebook
You will create Jupyter notebook to download and visualize the data. The psycopg library can be used to query the database. Your notebook should calculate the RMSE, R^2, and MAPE metrics. Since there will be

multiple home sale events per day, you should average the results for each day (e.g., using Pandas' groupby() methods). Plot the three lines over time.

Part V: Report
Write a short 2-page report analyzing the metrics. Include plots. Are the errors constant? Do they fluctuate day-to-day? Do they show a trend or any seasonality?

In the appendix of your report, include screenshots showing that your pipeline is setup and running properly:

- `docker ps` output showing the pipeline container running for at least 10 minutes
- PostgreSQL `\d tablename` output for your tables
- Output from selecting 10 records from your tables

Demonstration and Submission
Submit a PDF of your report and HTML file of your Jupyter Notebook to Canvas. Your instructor will review your code in your GitHub repository.

Rubric

Description	Percentage
Development Process • Commits are small and self-contained with reasonable messages. • Commits are contributed through feature branches and pull requests rather than direct commits to the main branch. • Commits were peer-reviewed and approved by other group members. • All source code is committed to the repository.	10%
Documentation • The cleaning and enrichment job has a separate README.md file • The README.md file describes how to build and run the batch job script (standalone) • The README.md file describes how to build and run the Docker image • The README.md describes the schemas for the tables used by the batch job.	5%
PostgreSQL • Tables for cleaned events are present. • Tables for processed ids are present. • Tables follow correct schemas including definition of appropriate fields, types, and nullability. • Id fields are indexed.	5%
Evaluation Batch Job Script • Downloads the records from the database. • Generates models predictions on records. • Inserts home sale event id, sale date, and true and predicted prices into the database. • Inserts ids into the processed table. • Takes credentials as environmental parameters.	20%

• Can be run standalone on the command-line. • Job is robust to failures that occur for individual records.	
Job Runner • APScheduler library is employed. • Creates a BlockingScheduler. • Adds batch job to scheduler. • Sets the job to run every 15 minutes, once maximum instance, and to coalesce jobs that overrun their scheduled times. • Schedules batch job to run immediately. • Starts scheduler.	20%
Deployment • Dockerfile is present in the repository. • Can successfully build a Docker image. • Docker container is running on the VM. • Predictions are written to output tables in PostgreSQL.	15%
Jupyter Notebook • Correctly queries data from PostgreSQL. • Calculates the RMSE, R^2, and MAPE metrics. • Computes the averages of the metrics by day. • Has plots for the metrics.	10%
Written Report • Report is written in a professional manner using proper grammar and spelling. • Report is a useful standalone document that can be shared with a business partner. • Appropriate types of plots were chosen for each analysis. • Axes are properly labeled. • Used legends if appropriate. • Chose appropriate axis limits to make plots readable and avoid misleading interpretations. • Font sizes are legible. • Figures are saved at high resolutions.	15%