Data Science at Scale Final Project Description - DSCC 202-402

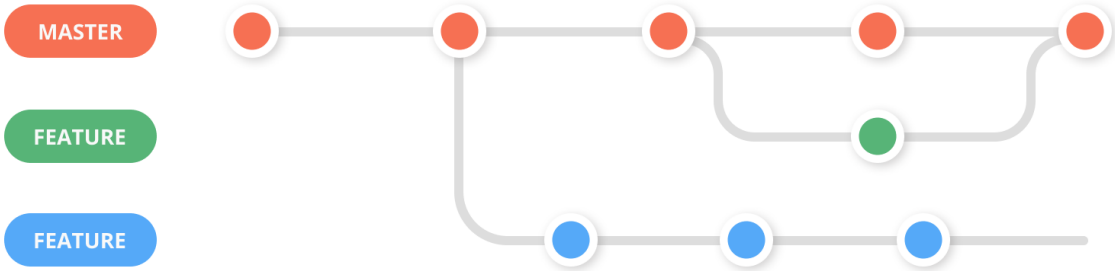
You and your group will develop an end-to-end data-intensive application that tracks the inventory of bikes and available docks at a given station with the Citibike Bike Sharing system in NYC. Each group will be assigned a specific station. To complete this project, you must apply the skills you have learned in DSCC 202-402 Data Science at Scale. This sort of forecasting application could be the basis of how an application like Bike Angels runs (**https://citibikenyc.com/bike-angels**) and helps keep the system running smoothly with the help of Data-Intensive Applications and Human participation. This type of automated inventory tracking is a classic augmentation application.

This document gives you and your team helpful background information and the rubric to grade your project. The project is worth 40% of your overall grade in the course. Everyone in your group will receive the same grade.

If you are a graduate student and registered for the DSCC-402 section of the course, you must submit not only your group project but also a written description of how your implementation meets the criteria outlined in the hidden debt page to avoid technical debt. Each graduate student will submit their paper/description. Graduate student work on this paper will be done outside the group and individually.

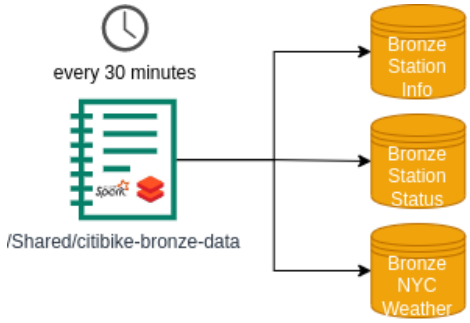
Each group will be assigned a specific station within the NYC Bike Share system to “adopt” for their application development. Your station is defined in the global variable: ***GROUP_STATION_ASSIGNMENT***. First, your group should establish a GitHub repo from your updated fork of the class GitHub repo: https://github.com/lpalum/dscc202-402-spring2023. Next,

there is a Final Project directory that includes a skeleton of your project. Next, in the background subdirectory, a getting started notebook will help acquaint you with the raw data sources for your project. Next, an include directory and notebook are executed in each project to provide the definition of global variables to be used within your project. Finally, your group should attach your repo to the databricks environment using the databricks repo feature: https://docs.databricks.com/repos/index.html. One helpful workflow is establishing a branch of your repo for each group member and then posting pull requests as necessary to merge your complete project on the master branch. This way, each group member can work on different project parts without affecting other members until the merge to master is done and the changes are pulled into their branch.



Your group will submit your project by providing a link to your master branch on or before the May 6th, 2023, due date.

The starting data package for your project is maintained in a raw data directory in DBFS. In addition, a set of delta tables is updated every 30 minutes by an automated job that runs on the platform.

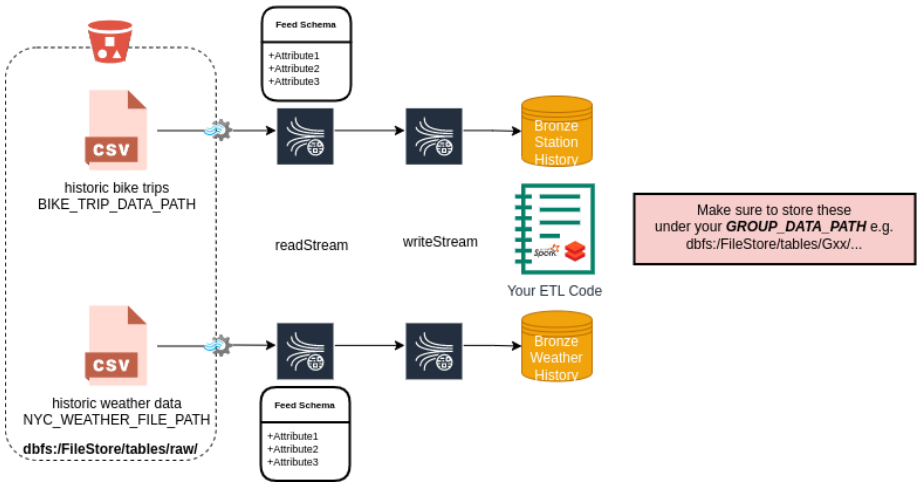


Your group should develop the following notebooks/components:

- Extract Transform Load (ETL) - 10pts

  Follows the medallion format of data development to refine the raw bike and weather information into what your model and application will require at the bronze, silver, and gold level of data quality/refinement.

  - Document the delta tables your project requires at the bronze, silver, and gold levels. In addition, your streaming definition should include the path, schema, and data definitions.
  - Process historical trip and weather data with an ETL pipeline using a spark streaming job to ingest new data as it comes into the sources. Your pipeline should explicitly define schemas and their evolution. Use Partitioning and Z-ordering to optimize delta tables for your application.



  - Your ETL pipeline should create Bronze Data from the raw sources.
  - Your ETL pipeline should create Silver Data for your required training features and run-time inference requirements.

- Your ETL pipeline should define the Gold application and monitoring data even if created and maintained in the application notebook.
- Your ETL pipeline should be immutable. No side effects to running it multiple times with the same input data.
- ***YOU MUST STORE YOUR DATA IN YOUR GROUP DIRECTORY IN DBFS. SEE GROUP_DATA_PATH GLOBAL VARIABLE.***

- Exploratory Data Analysis (EDA) - 10pts

  uses the raw weather and bike trip data to answer the following questions (at a minimum):
  - What are the monthly trip trends for your assigned station?
  - What are the daily trip trends for your given station?
  - How does a holiday affect the daily (non-holiday) system use trend?
  - How does weather affect the daily/hourly trend of system use?
  - ***NOTE IF YOU CREATE ANY SQL TABLES OR VIEWS, BE SURE TO USE YOUR GROUP'S UNIQUE DB NAME. SEE GROUP_DB_NAME GLOBAL VARIABLE.***
- Modeling and ML Ops - 10pts
  - Considering historical data, build a forecasting model that infers net bike change at your station by the hour.
  - Register your model at the **staging** and **production** level within the Databricks model registry.
  - Store artifacts in each MLflow experiment run to be used by the application to retrieve the staging and production models.
  - Tune the model hyperparameters using the MLflow experiments and hyperparameter scaling (spark trials, hyper-opts).
  - ***YOU MUST NAME YOUR GROUP MODEL IN THE MLFLOW REGISTRY WITH YOUR GROUP'S UNIQUE NAME. SEE GROUP_MODEL_NAME GLOBAL VARIABLE.***
- Application - 10pts
  - A gold data table should store inference and monitoring data.
  - Monitoring your staging vs. production model actual vs. predicted real-time display. You will need more recent data than the end of your training set. This

data is available in the BRONZE_STATION_STATUS_PATH delta table and is updated every 30 minutes by a regular job running on the platform.

- Include code that archives the current production model and promotes the staging model when you specify that it should occur via testing of the staging model.
- Each run of the notebook will update/display the following:
  - Current timestamp when the notebook is run (now)
  - Production Model version
  - Staging Model version
  - Station name and a map location (marker)
  - Current weather (temp and precip)
  - Total docks at this station
  - Total bikes available at this station
  - Forecast the available bikes for the next 4 hours.
  - Highlight any stock out or full station conditions over the predicted period.
  - Monitor the performance of your staging and production models using an appropriate residual plot that illustrates the error in your forecasts.

Steps to get started

- Fork the class repo into your group GitHub repo/account. The class repo will provide the project template and getting started information.
- Create branches for each group member to work on their project parts.
- Establish the connection to your repo in databricks
- Develop a plan to develop your application and coordinate your group. This project is a group software/data development project. Ask questions about the best way to do this if you need clarification, or ask for a review of your plan. The earlier, the better.
- Pull, develop, test, commit, merge to master, and repeat till complete.

References

- https://citibikenyc.com/
- https://citibikenyc.com/bike-angels
- https://github.com/lpalum/dscc202-402-spring2023
- https://docs.databricks.com/repos/index.html
- https://citibikenyc.com/system-data

- https://openweathermap.org/api/one-call-api#hist_parameter