ST CLAIR COLLEGE,

DOWNTOWN CAMPUS, WINDSOR

Data Analytics for Business

4th Semester

CAPSTONE PROJECT - INTERIM REPORT

<u>TTC BUS DELAY ANALYSIS – RACE AGAINST TIME</u>

Submitted By

Group 7

Khyati Surolia - 0856344

Udanie Subasinghe - 0860626

Sankeerthana Karem - 0862177

Navya Senakkayala - 0856219

Manisatya Alapati - 0857335

Supervised By

Prof. Manjari Maheshwari

Submission Date: 01/03/2025

Introduction

Public transportation efficiency is crucial for urban mobility, and delays in transit services can significantly impact commuters. The Toronto Transit Commission (TTC), one of North America's largest transit systems, frequently experiences delay due to various factors such as weather conditions, operational issues, and peak-hour congestion. Understanding these delays through data-driven analysis can help improve transit reliability and passenger experience.

In the first phase of this capstone project, we conducted Exploratory Data Analysis (EDA) to identify patterns and key contributors to TTC delays. This included analyzing historical data, visualizing trends, and identifying correlations between variables. In the current phase, Part 2, we focus on modeling these delays using machine learning techniques and building Azure cloud pipelines. The goal is to develop predictive models that can forecast delays based on relevant factors and provide actionable insights for transit optimization.

Methodologies

The goals of the current project phase are stated below:

- To create an automated **data ingestion pipeline** in **Azure Data Factory (ADF)** for ingesting bus delay data using two methods:
  - **Azure Blob Storage** container into an **Azure SQL Database**.
  - **Azure Data Lake Storage,** i.e. ADLS to another **Azure Data Lake Storage**
- To predict delay and probability using various modelling approaches like:
  - ARIMA
  - SARIMA
  - RF Regressor/Classifier
  - XGBoost regressor
  - LSTM model
  - GRU deep learning
  - MLP
  - Monte Carlo simulations

Here is a high-level summary of the pipeline creation process:

Data Ingestion Pipeline Steps:

- **Data Source Configuration:** Configured a source-linked service to Azure Blob Storage. The Excel file was uploaded into a storage account container using MS Azure Storage Explorer.

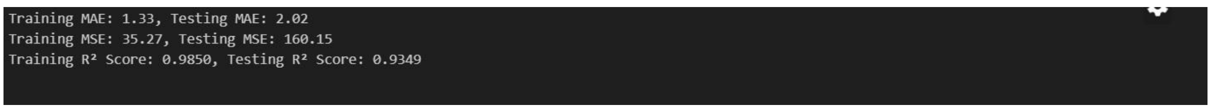
- **Data Transformation & Loading**: The **Copy Data** activity was used to ingest data from Blob Storage into Azure SQL Database.

- **Destination Configuration**: Configured Azure SQL Database as the pipeline destination. Ensured the target table (dbo.Bus_Delay_Data) was created in the database.
- **Scheduling**: The pipeline was scheduled to run **daily at 2 AM** for automatic data ingestion.
- **Trigger Setup**: A trigger was configured to execute the pipeline on a scheduled basis.

Key Steps in Modelling Approaches

- Data Preparation & Analysis from Phase 1
    - Cleaned data from the **previous phase** was further analyzed for modelling.
    - Exploratory analysis was again reviewed using **Python libraries** such as pandas, numpy, and seaborn to understand data distributions and relationships.
- **Proof of Concept (PoC) for 2023 Data**
    - Initial modelling experiments, such as ARIMA, LSTM, XGBoost, etc., were conducted using **one year of data (2023)** to validate approach feasibility.
    - **Machine learning models** were implemented using sklearn, while deep learning models leveraged **GRU (Gated Recurrent Units)** with tensorflow.
    - **Monte Carlo simulation** was applied to the **dataset** to model uncertainty and assess a range of possible bus delays. The simulation provided **probabilistic delay predictions**, helping to quantify variability and improve risk assessment for transit operations.
- **Feature Engineering & Model Optimization in the entire dataset**
    - Applied **feature engineering techniques** to enhance model performance and capture key predictive patterns.
    - Conducted **hyperparameter tuning** to optimize model efficiency and accuracy for the entire dataset.

Outputs for RF Regressor, XGBoost Regressor and RF Classifier



```
Training MAE: 1.33, Testing MAE: 2.02
Training MSE: 35.27, Testing MSE: 160.15
Training R² Score: 0.9850, Testing R² Score: 0.9349
```

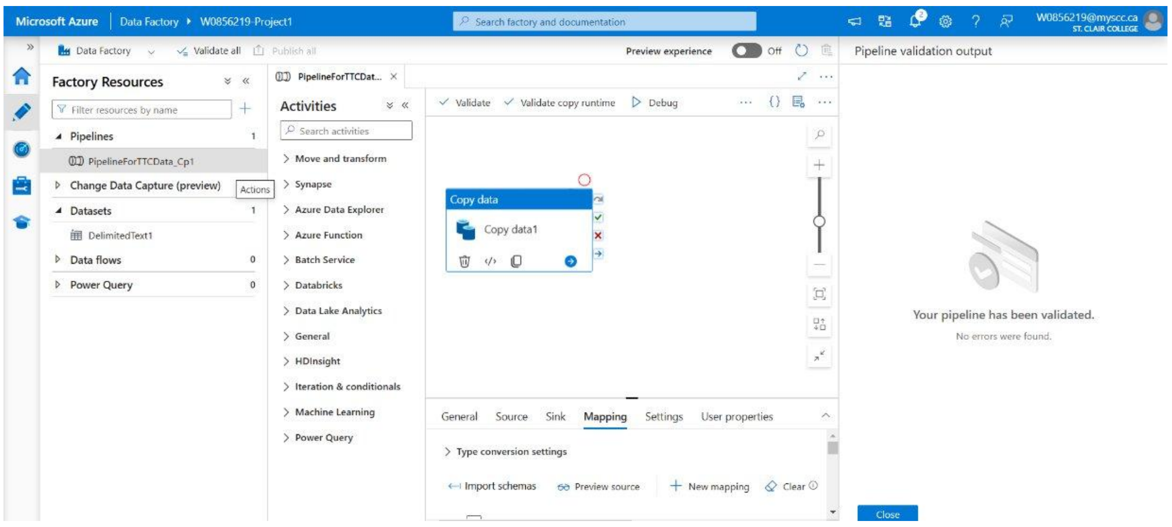
```
Best Parameters: {'colsample_bytree': 0.8, 'learning_rate': 0.05, 'max_depth': 6, 'n_estimators': 200, 'subsample': 0.8}
Training MAE: 1.42, Testing MAE: 1.70
Training MSE: 100.67, Testing MSE: 202.46
Training R² Score: 0.9572, Testing R² Score: 0.9177
```

```
Classification Report:
                       precision    recall  f1-score   support

  delay_category_Long       0.98      0.96      0.97     14764
delay_category_Medium       0.97      0.97      0.97     27418
 delay_category_Short       0.91      0.94      0.93      3827

             accuracy                           0.97     46009
            macro avg       0.95      0.96      0.95     46009
         weighted avg       0.97      0.97      0.97     46009
```

Insights

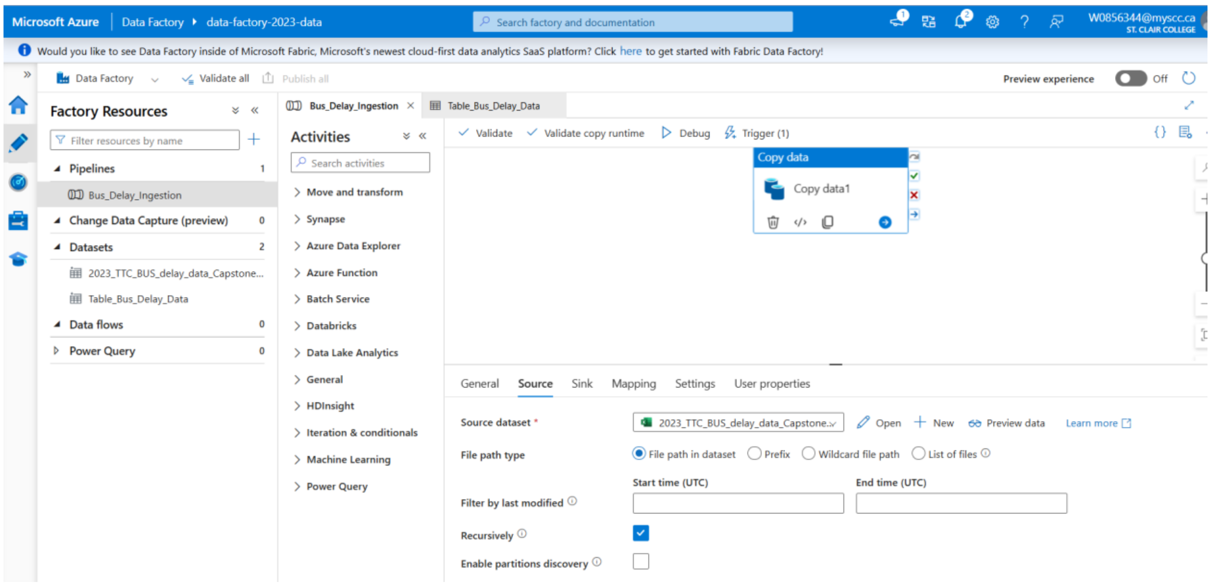
Based on the pipeline setup and execution process, the following insights were gathered:

- **Efficient Data Pipeline Creation**: Azure Data Factory allowed easy integration between Blob Storage and Azure SQL Database, streamlining the data ingestion process.
- **Scheduled Automation**: Automating the pipeline to run daily ensures that the data is updated without manual intervention. This will save time and resources.



Your pipeline has been validated.
No errors were found.

Figure 1: Pipeline creation using ADLS



Would you like to see Data Factory inside of Microsoft Fabric, Microsoft's newest cloud-first data analytics SaaS platform? Click here to get started with Fabric Data Factory!

Pipelines	1
Bus_Delay_Ingestion	
Change Data Capture (preview)	0
Datasets	2
2023_TTC_BUS_delay_data_Capstone...	
Table_Bus_Delay_Data	
Data flows	0
Power Query	0

Source dataset * 2023_TTC_BUS_delay_data_Capstone...

File path type [✓ File path in dataset] Prefix Wildcard file path List of files

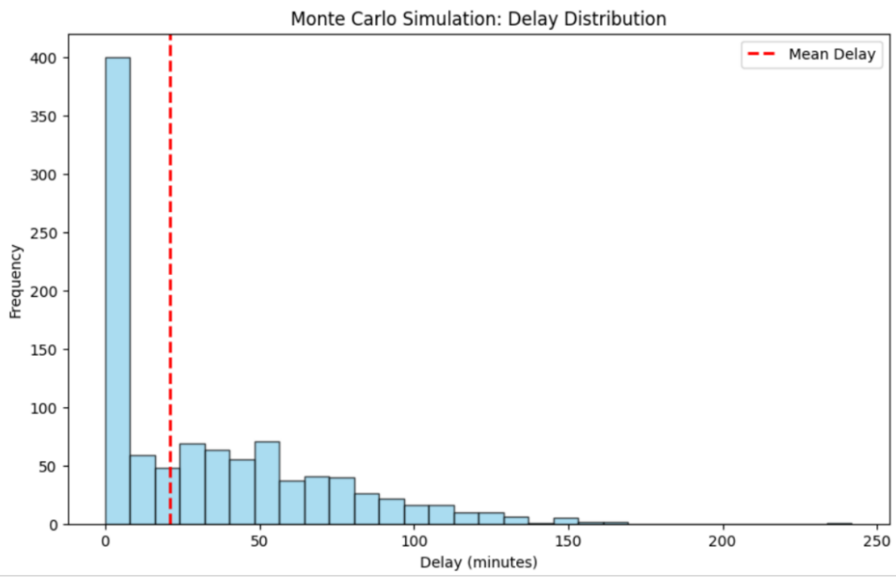
Filter by last modified — Start time (UTC): End time (UTC):

Recursively [✓]

Enable partitions discovery ☐

Figure 2: Pipeline creation using Azure Blob and SQL Server.

Model	Strengths	Weaknesses
RF Regressor/Classifier	Handles non-linearity, robust to missing data, interpretable feature importance	May overfit without tuning, less precise on extreme delays
XGBoost Regressor	High accuracy, efficient feature selection, handles complex patterns	Requires careful tuning, computationally intensive
Monte Carlo Simulation	Provides delay uncertainty estimates, proper for risk assessment	It needs precise probability distributions, high computational cost



Project Progress

- The pipeline successfully connects to both methods Azure Blob Storage and Azure SQL Database plus ADLS, with data being transferred seamlessly between the two.
- The **data schema** for the source and destination is aligned, with proper mappings defined for the data to be inserted into the SQL table.
- **XGBoost** performed best in **prediction accuracy**, making it the recommended model for delay estimation.
- **Random Forest** remains useful for initial analysis and interpretability of feature importance.

Challenges and Issues Encountered

- Initially faced issues with **Blob Storage container visibility** and **large file uploads** due to size limitations. This was resolved by adjusting container permissions and using Azure Storage Explorer.
- Encountered **connection issues to Azure SQL Database** that were resolved by configuring the correct firewall rules and ensuring that the integration runtime had access.
- The connection to the server was not established initially while setting up ADLS. The linked service on Azure Data Factory was fixed by adjusting the networking connection on Azure Data Studio.
- XGBoost requires careful tuning of hyperparameters to prevent overfitting.

Next Steps

Based on the pipeline setup and initial testing, the following next steps are recommended:

- **Monitoring and Validating Pipeline**
  - Pipeline Monitoring: Use the ADF Monitor tab to track the status of pipeline runs.
  - Run History Logs: Review logs for errors and validate the success of pipeline runs.
  - SQL Database Verification: Validate that data was successfully ingested into the Azure SQL database by running SQL queries on the target table.

- **Validate Larger Datasets**:
  - Perform tests with entire datasets to ensure the pipeline scales effectively.
- **Explore Azure Machine Learning**:
  - Implement different model approaches executed as POC i.e. proof of concepts in VS code/JupyterLite on Azure ML services and validate their effectiveness,
- **Documentation & Knowledge Sharing**:
  - Document the entire pipeline creation and monitoring process for future reference and sharing with the team.
  - Share the learnings and insights from this process with team members to ensure the team is well aligned in project progress.
- **Deploy the best-performing model in an automated prediction pipeline.**
- **Validate predictions against real-time bus delay data.**
- **Test Monte Carlo simulations efficiency for the entire dataset.**

Conclusion

This phase implemented an automated cloud pipeline using Azure Data Factory, Blob Storage, SQL Database, and ADLS for seamless TTC bus delay data ingestion. XGBoost emerged as the best-performing model so far in analysis for delay prediction, while Random Forest provided key feature insights.

Challenges like storage visibility, extensive file handling, and SQL connectivity were resolved, ensuring a scalable and automated workflow. The next steps include deploying models in an automated pipeline, testing models for the entire dataset , validating predictions with real-time data, and refining performance to enhance transit reliability and commuter experience.

References

- **https://learn.microsoft.com/en-us/training/modules/intro-to-azure-data-factory/1-introduction**
- **https://scikit-learn.org/stable/modules/ensemble.html**
- **https://www.tensorflow.org/tutorials**