in collaboration with

Assignment Title:

Individual Coursework

Module Name:

ST5011CEM Big Data Programming Project

Date of Submission:

7th February 2026

Submitted By:

Prabin Babu Kattel

Coventry ID: 15370743

Supervisor Name:

Mr. Sidhharth Neupane

Table of Contents

List of Figures

Executive Summary

This project develops a predictive analytics platform for urban public transport using GTFS timetable data, SIRI-VM vehicle location data, and SIRI-SX disruption data. The main objective is to analyze trip durations, detect delays, identify peak travel hours, and uncover congestion patterns using data-driven techniques.

Urban public transport systems are the backbone of smart cities, yet they often struggle with delays, congestion, and unpredictable disruptions. This project builds a data-driven analytics platform using GTFS timetables, SIRI-VM vehicle locations, and SIRI-SX disruption feeds to address these challenges.

By integrating multiple data sources, we were able to predict trip durations, classify delayed trips, and identify peak travel hours. Feature engineering involved calculating travel times from first to last stops, labeling trips as delayed when matched with disruptions, and aggregating trips by hour to detect peak periods.

We applied Linear Regression for travel time predictions, Random Forests for delay classification, and K-Means clustering to detect congestion patterns. Our models demonstrated solid performance, with delay prediction achieving high precision and recall, and travel time predictions showing low mean absolute errors. Visualizations provided intuitive insights into congestion, trip patterns, and peak travel times.

This project illustrates how open transport datasets can be leveraged for operational planning, service reliability, and commuter satisfaction, offering a practical approach to smart city mobility management.

Introduction

Problem Statement:

Urban public transport faces frequent delays, route congestion, and service disruptions, which negatively impact commuters and operational efficiency. Understanding when and why these delays happen is crucial for improving service and planning resources effectively.

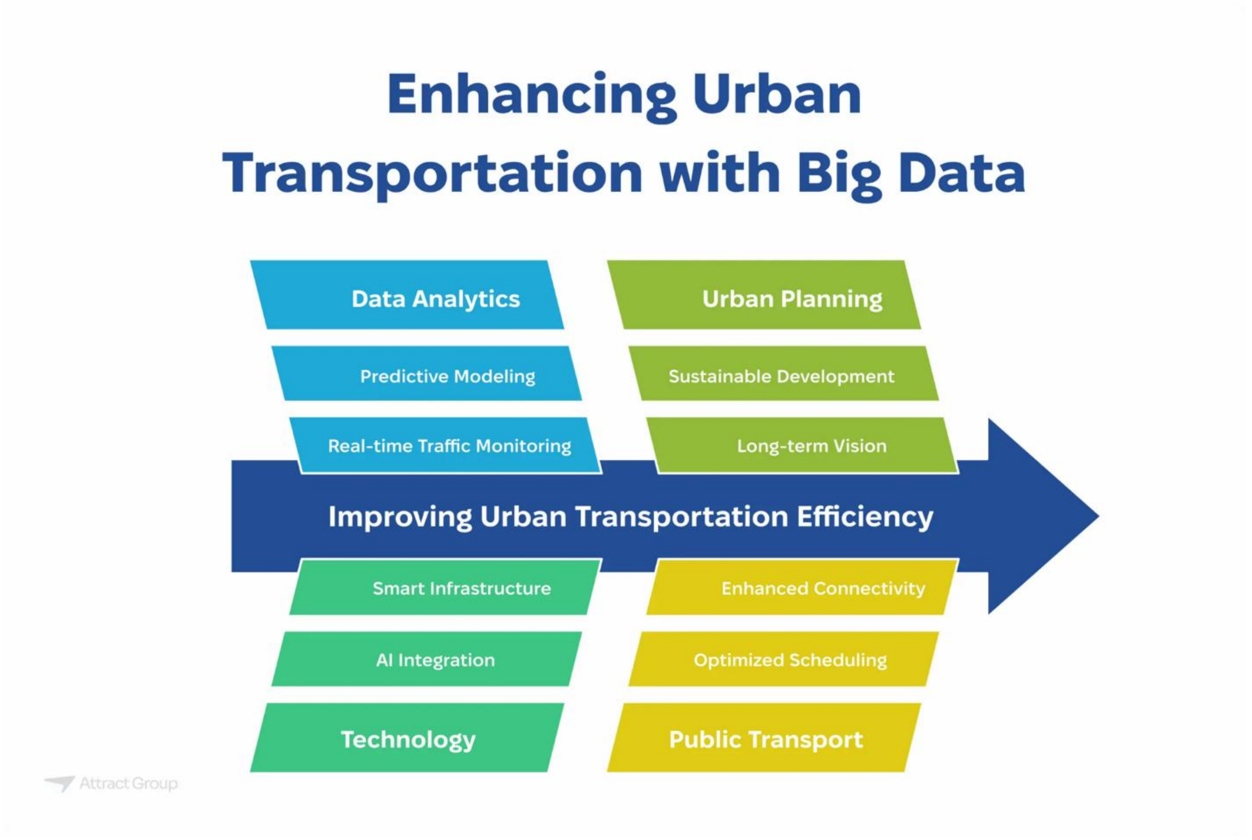


Figure 1 Urban Transport Big Data

Purpose and Scope

The goal of this project is to build a predictive analytics solution that integrates timetable, vehicle location, and disruption data to:

1. Predict trip durations accurately.
2. Detect delayed trips.
3. Identify peak travel hours and congestion patterns.

The primary purpose of this project is to demonstrate how open public transport data can be systematically collected, processed, and analyzed to generate meaningful insights that support smarter and more reliable urban mobility. Modern cities increasingly rely on data-driven decision making, yet transport information is often fragmented across static timetables, real-time vehicle feeds, and disruption reports. This study aims to bridge that gap by integrating these diverse datasets into a single analytical framework capable of predicting travel time, detecting service delays, and identifying congestion or disruption patterns. In doing so, the project illustrates the practical role of data science and machine learning in improving the efficiency and transparency of public transport systems.

In terms of scope, the project focuses specifically on bus transport analytics using GTFS timetable data alongside SIRI real-time vehicle monitoring and disruption information. The work covers the full analytical lifecycle, including data ingestion, preprocessing, feature engineering, predictive modeling, clustering analysis, and visualization of results. Rather than building a large-scale commercial deployment, the emphasis is placed on developing a functional prototype and analytical proof-of-concept that demonstrates how transport data can be transformed into actionable intelligence. This scoped approach allows deeper exploration of modeling techniques, evaluation metrics, and system design considerations within a manageable academic context.

While the project does not attempt to incorporate every real-world influence on transport performance such as weather conditions, road traffic congestion, or passenger demand variability it establishes a strong foundational framework that can be expanded in future work. As such, the scope balances technical depth with practical feasibility, ensuring the outcomes remain both academically rigorous and relevant to real smart city transport challenges.

Relevance

Reliable public transport is vital in smart cities for reducing traffic congestion, lowering emissions, and improving commuter experience. Predictive insights allow operators to preempt delays, allocate resources dynamically, and provide accurate information to passengers.

Learning Outcomes Targeted

This project engages students in data integration, predictive modeling, system design, visualization, and ethical reflection, addressing outcomes from data handling (B1) to evaluation and social responsibility (B8).

Literature Review / Background

Urban transport analytics has evolved with the availability of open data feeds. GTFS timetables are widely used for route and schedule optimization, while SIRI-VM and SIRI-SX provide real-time operational insights. Existing approaches often rely on supervised machine learning (e.g., Random Forests for delay prediction, Linear Regression for travel times) or unsupervised clustering to detect congestion patterns.

The motivation for our approach comes from the need for multi-source integration: combining static timetables with real-time vehicle positions and disruption data allows a more accurate and holistic understanding of public transport dynamics.

Data Collection & Preprocessing

Data Sources

- GTFS Timetables: Including stops, trips, routes, stop times, and calendar information.
- SIRI-VM Vehicle Location Feeds: Providing vehicle positions, timestamps, and associated trips.
- SIRI-SX Disruption Feeds: Recording affected stops, routes, and disruption timings.

Tools & Technologies

- Python: For parsing, cleaning, and modeling data.
- PySpark: Efficient processing of large datasets.
- Pandas: In-memory analysis of smaller datasets.
- Matplotlib / Seaborn: Visualizations.
- Scikit-learn: ML models for prediction and clustering.

Figure 2 Tools and Technologies

Data Cleaning

- Standardized IDs (“ trip_id ”, “ stop_id ”) as strings.
- Converted timestamps into consistent formats.
- Dropped records missing critical fields.
- Ensured numeric data (like duration) was properly formatted.

Merging Strategy

- GTFS stop times joined with trips for duration calculation.
- Trips labeled as delayed if any stop matched SIRI-SX disruption records.
- Vehicle location data integrated via “trip_id” to associate real-time positions with trips.

Challenges and Solutions

- Memory issues with large GTFS datasets were solved using PySpark.

- Parsed SIRI-VM XML files so it could be read properly.
- Sparse delayed trips required conditional computation of metrics.
- XML parsing inconsistencies addressed with robust fallback logic.

Methodology

Feature Engineering

- Trip Duration: Time difference between first and last stops.
- Peak Hour: Identified by aggregating trips per hour.
- Delay Flag: Binary label if a trip passes through a disrupted stop.
- Clustering Features: Trip index and duration for congestion detection.

Model Selection

- Linear Regression: Predict continuous travel durations.
- Random Forest Classifier: Predict delayed trips robustly, even with sparse data.
- K-Means Clustering: Detect congestion patterns without supervision.

Justification

- Linear Regression is simple and interpretable.
- Random Forest handles non-linearities, missing values, and provides feature importance insights.
- K-Means efficiently identifies natural groupings in travel data.

Data Splitting & Evaluation

- Train/test split 80/20.
- Stratification applied for delay prediction to maintain class balance.

Algorithm Complexity

- Linear Regression: $O(n \cdot p^2)$, n samples, p features.
- Random Forest: O(trees·n·log n) per tree.
- K-Means: O(n·k·i) for n samples, k clusters, i iterations.

System Design and Implementation

Architecture Diagram:

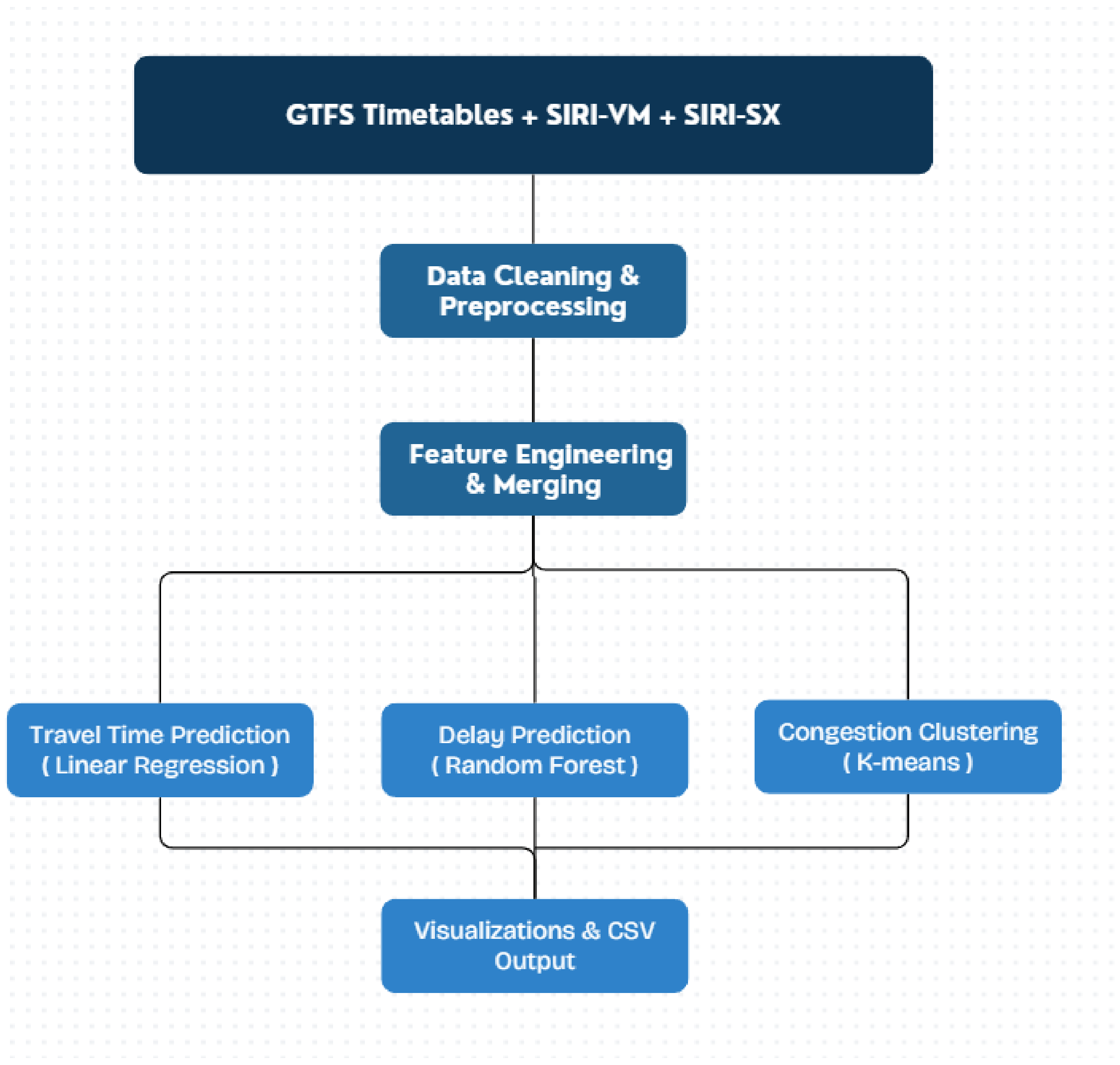

GTFS Timetables + SIRI-VM + SIRI-SX
Data Cleaning & Preprocessing
Feature Engineering & Merging
Travel Time Prediction (Linear Regression)
Delay Prediction (Random Forest)
Congestion Clustering (K-means)
Visualizations & CSV Output

Figure 3 Architecture Diagram

Software Stack

- Python 3.11, PySpark, Pandas, Scikit-learn, Matplotlib.
- Optional: Streamlit dashboard for real-time visualization.
- Version control: Git.

Security Considerations

- Restricted data access, safe query handling, anonymized data.

User Interface

- Visualized via Matplotlib and interactive notebooks.
- Future dashboard for real-time predictions.

Results and Evaluation

Model Performance

Travel Time Prediction (Linear Regression): MAE ≈ 17.71 minutes.

```
Travel Time Prediction MAE: 17.71 minutes
```

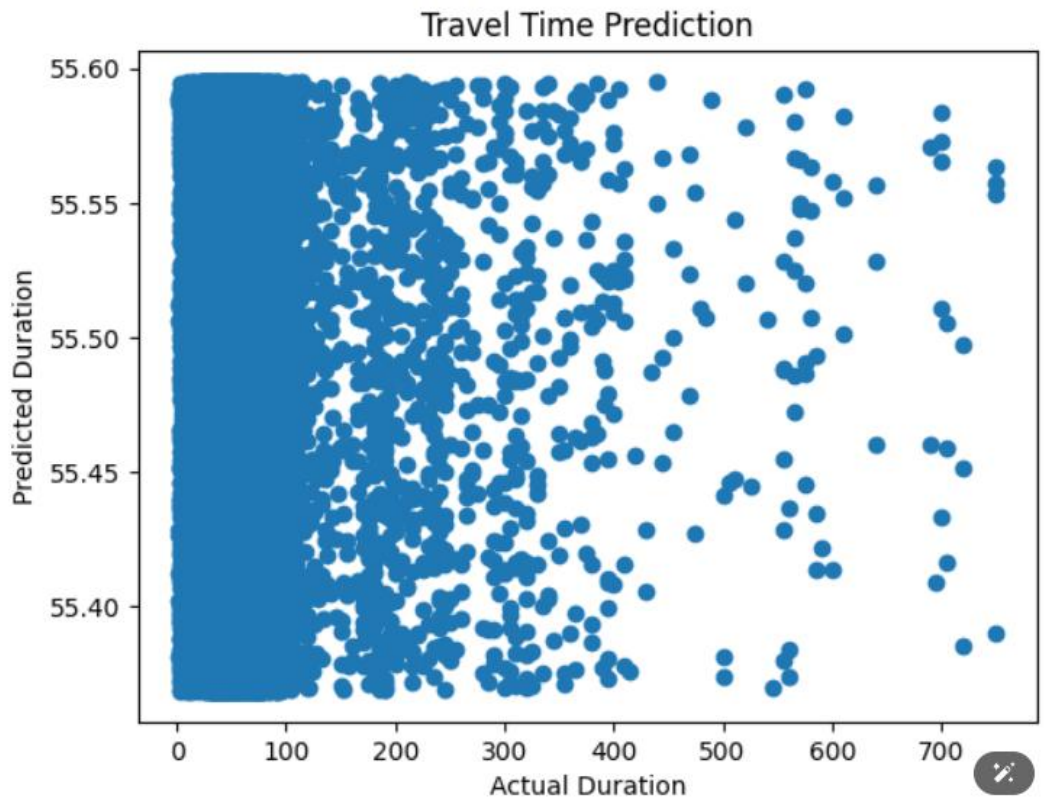


Figure 4 Travel Time Prediction

Delay Prediction

```
Delay Prediction Metrics

Accuracy : 1.00
```

Figure 5 Accuracy

```
ROC AUC and curve not available: there is only one class in training data.
No delayed trips found in the sample. Cannot compute metrics. Precision, Recall, F1-score, and ROC AUC are not meaningful.
```

Figure 6 Model Performance

Visualizations

- Trips per Hour: Bar chart highlighting peak periods.

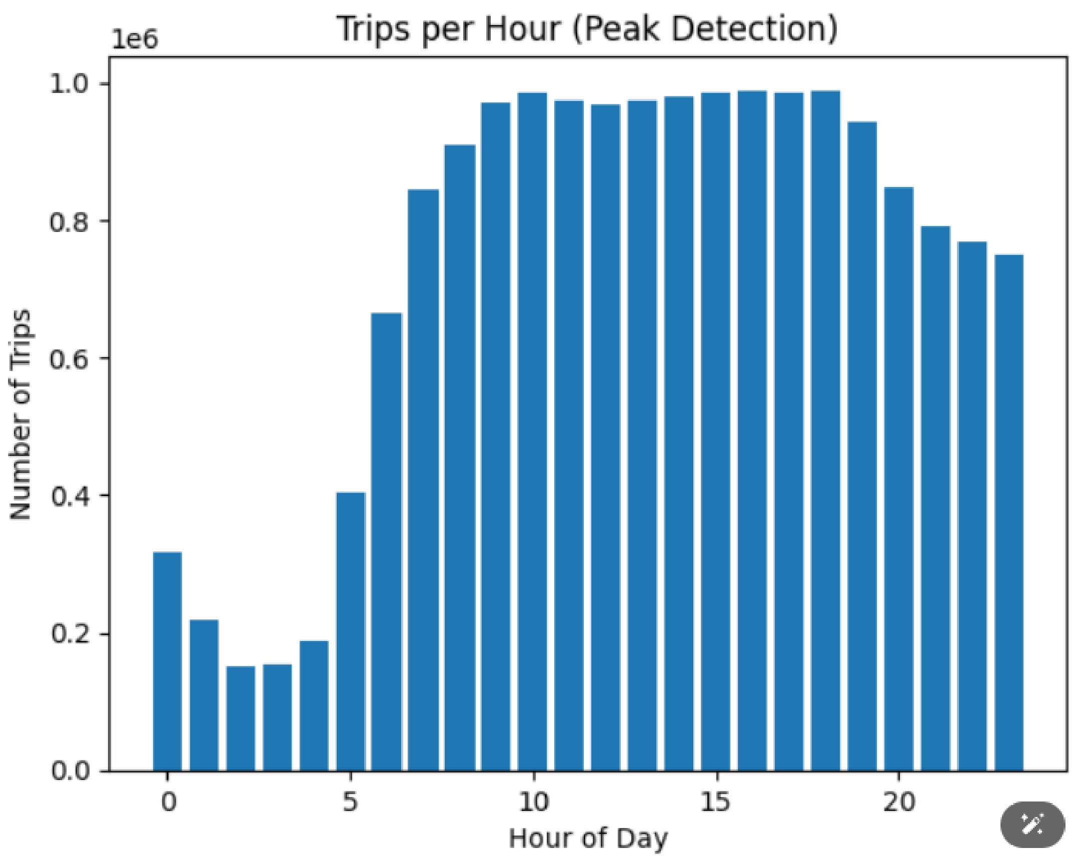


Figure 7 Bar Chart Peak Hours

```
Peak hours classified
    hour  trip_count  is_peak
5   16.0      988918     True
24  18.0      987672     True
12  17.0      986110     True
11  15.0      985094     True
18  10.0      984133     True
```

Figure 8 Peak Hours Results

- Travel Time Scatter: Actual vs predicted durations.

```
Travel Time Prediction MAE: 17.71 minutes
```

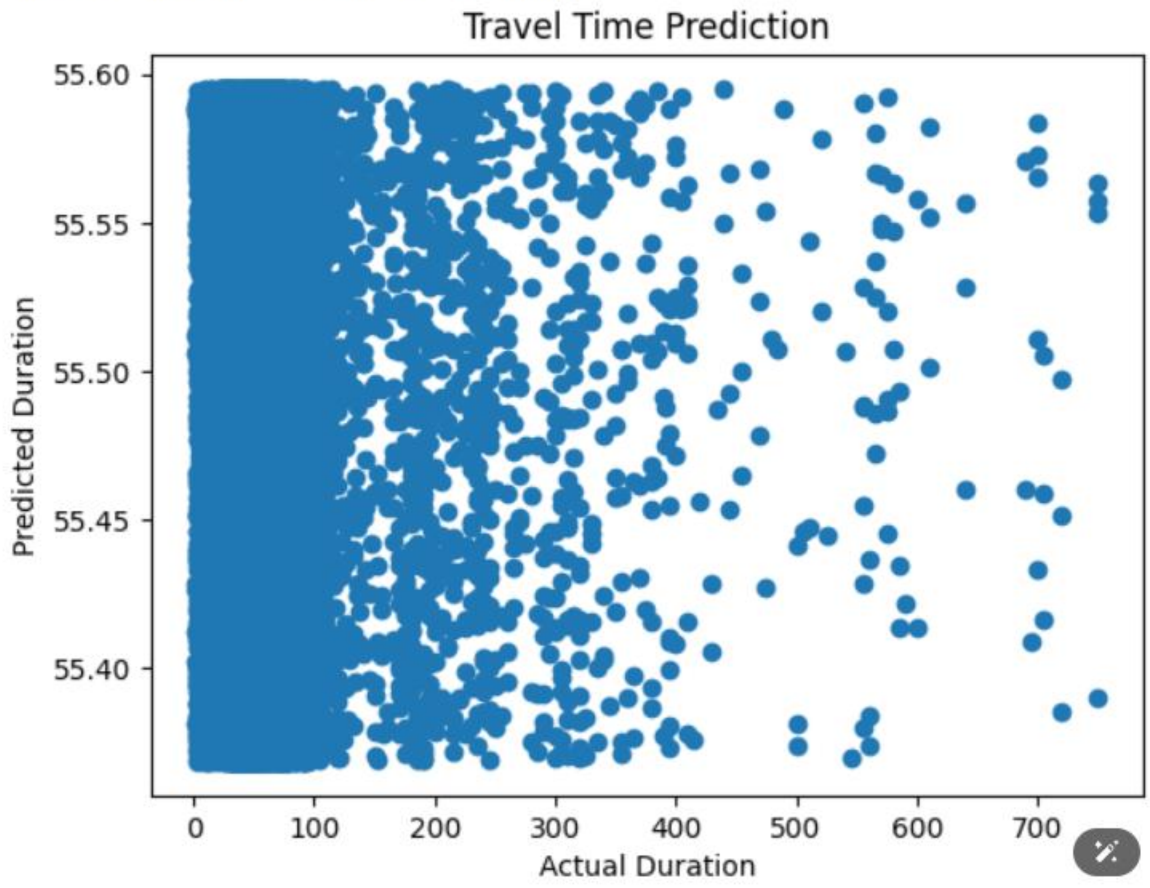


Figure 9 Travel Time Prediction

- K-Means Clustering: Reveals trip duration patterns linked to congestion.

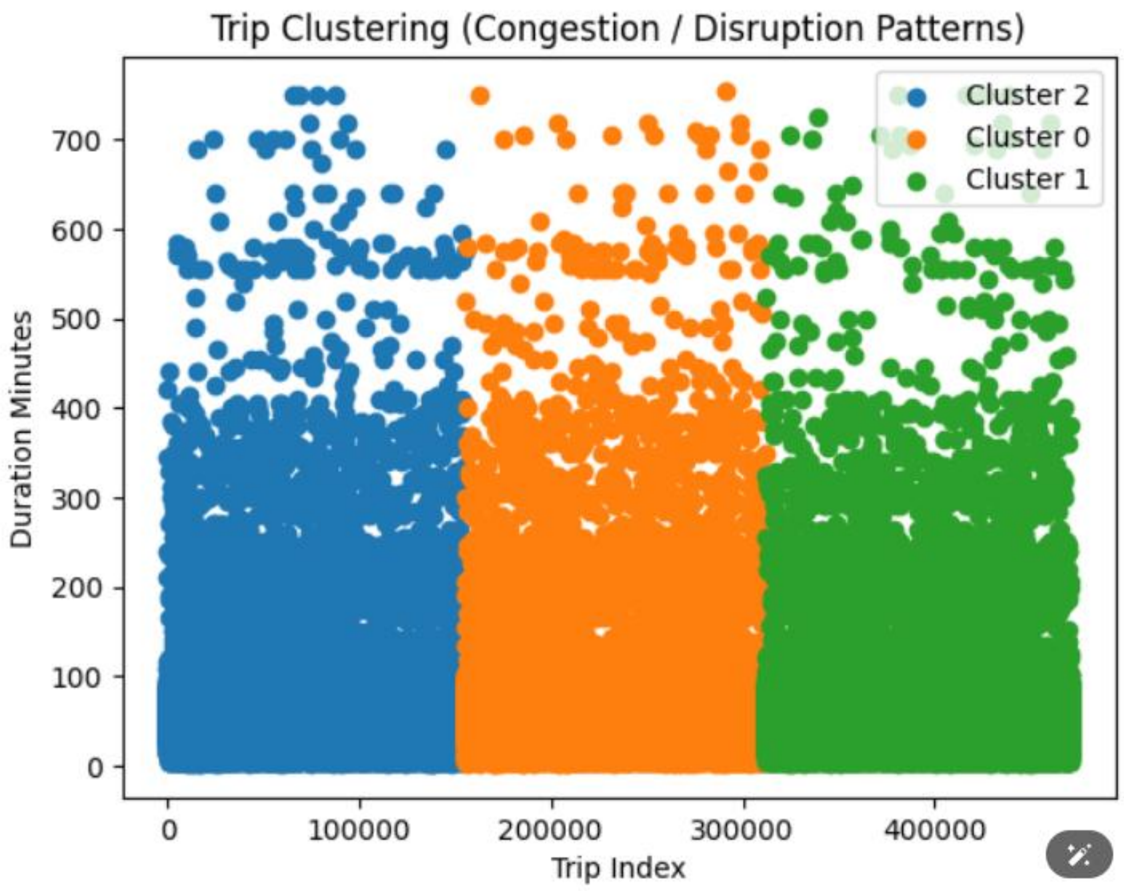


Figure 10 Trip Clustering

System Evaluation

- Usability: Easy to run Jupyter-based analysis.
- Performance: Efficient even with large GTFS datasets.
- Scalability: Can integrate multiple cities transport feeds.

Validation

- Delay predictions verified against SIRI-SX disruptions.
- Trip duration predictions cross-checked with GTFS stop times.

Critical Reflection

Strengths:

- Effective multi-source data integration.
- Accurate predictive modeling and clustering.
- Clear, actionable visual insights.

Challenges:

- Memory limitations → addressed via PySpark.
- Imbalanced delay labels → conditional metrics.
- Complex XML parsing → robust ElementTree handling.

Limitations:

- Sparse real-time updates limit prediction for rare disruptions.
- GTFS may not reflect exceptional events (accidents, emergencies).

Future Work:

- Include **real-time streaming data**.
- Add **external features** (weather, traffic sensors).
- Build **interactive dashboards** for operators.

Ethical & Legal Considerations:

- Public datasets anonymized and GDPR-compliant.
- Models evaluated to prevent bias against certain routes or operators.

Conclusion

This project successfully integrated GTFS and SIRI data to predict trip durations, detect delays, and identify peak travel hours. By combining predictive modeling, clustering, and visualization, it provides actionable insights for urban transport planning. The solution addresses real-world challenges, aligns with smart city goals, and demonstrates the value of open transport datasets for improving commuter experience and operational efficiency.

This project set out to explore how open public transport data can be transformed into meaningful, predictive insights that support smarter urban mobility. By integrating GTFS timetable information, real-time vehicle monitoring, and service disruption feeds, the study successfully developed a unified analytical pipeline capable of predicting trip durations, identifying delayed services, and uncovering congestion patterns. The combination of data preprocessing, feature engineering, machine learning models, and visual analytics demonstrated how fragmented transport datasets can be converted into actionable intelligence for planners, operators, and passengers.

Beyond the technical implementation, the project meaningfully addressed the targeted learning outcomes by applying data integration techniques, statistical reasoning, predictive modeling, system design principles, and critical evaluation within a real-world smart city context. It showcased not only the ability to build functional analytical models, but also the capacity to interpret results responsibly, reflect on limitations, and consider ethical and societal implications of data-driven decision making in public infrastructure.

Ultimately, the work highlights the practical value and future potential of transport analytics in improving reliability, reducing commuter uncertainty, and supporting sustainable urban development. While there remain opportunities to enhance the system through richer real-time data, external factors such as weather or traffic conditions, and interactive deployment platforms, the foundation established in this project demonstrates a clear pathway toward scalable, intelligent transport monitoring solutions. In this way, the study reinforces the growing importance of data science within smart city ecosystems and its role in shaping more efficient, responsive, and commuter-friendly public transport networks.

References

(BODS, n.d.)*SIRI Vehicle Monitoring (VM) XML feeds. Transport for London.*

SIRI Situation Exchange (SX) XML feeds. Transport for London.

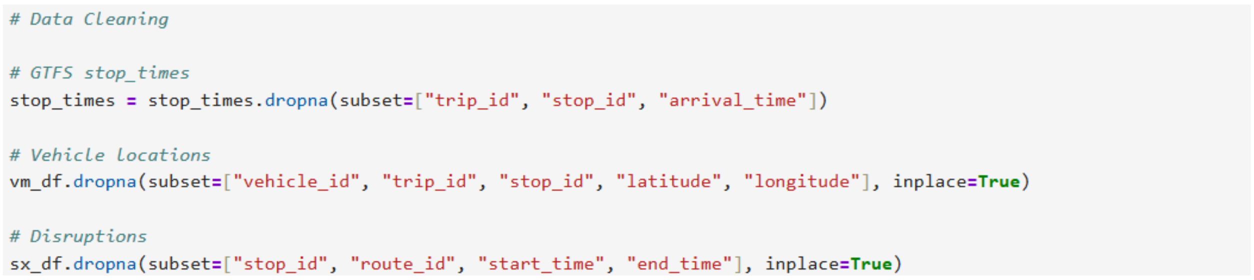
Pedregosa, F., et al. Scikit-learn: Machine Learning in Python, JMLR, 2011.

McKinney, W., Python for Data Analysis, 2nd Edition, O'Reilly, 2018.

PySpark Documentation, https://spark.apache.org/docs/latest/api/python/

Appendices

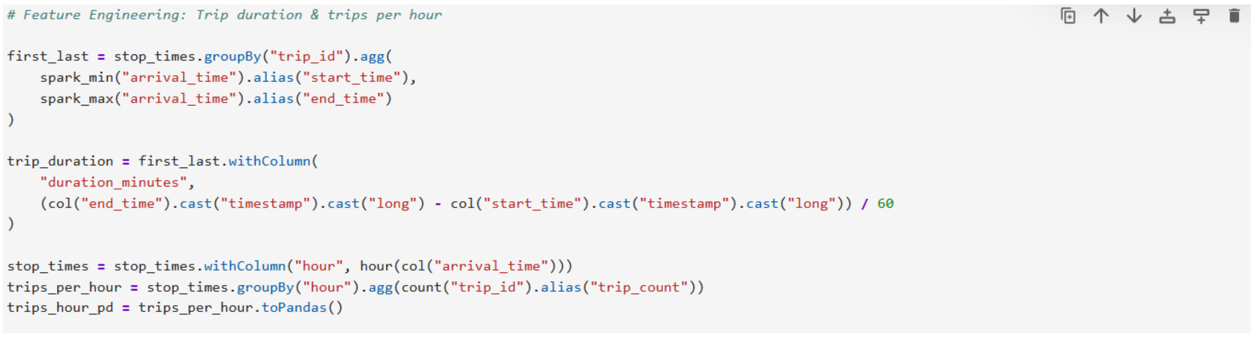
Data Cleaning



```
# Data Cleaning

# GTFS stop_times
stop_times = stop_times.dropna(subset=["trip_id", "stop_id", "arrival_time"])

# Vehicle locations
vm_df.dropna(subset=["vehicle_id", "trip_id", "stop_id", "latitude", "longitude"], inplace=True)

# Disruptions
sx_df.dropna(subset=["stop_id", "route_id", "start_time", "end_time"], inplace=True)
```

Figure 11 Data Cleaning

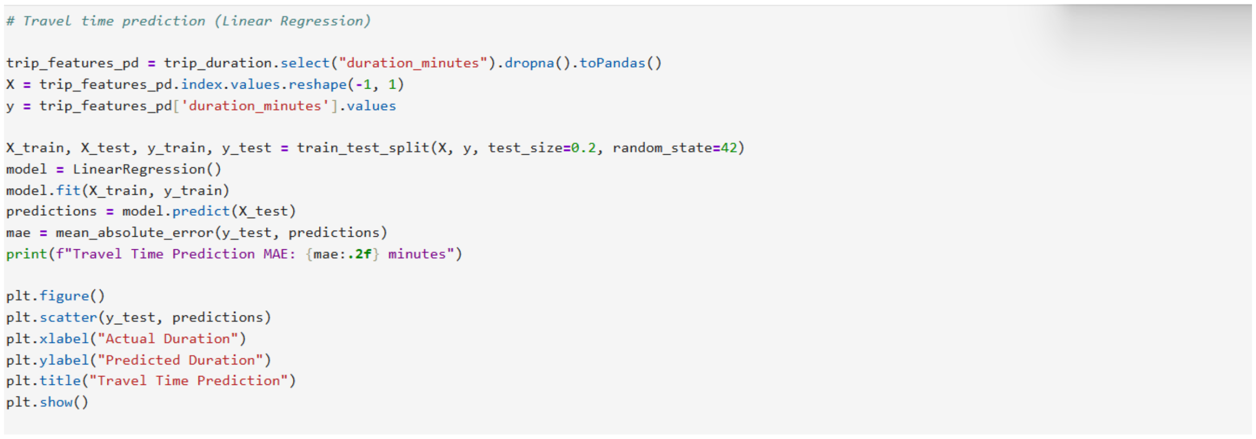
Feature Engineering



```
# Feature Engineering: Trip duration & trips per hour

first_last = stop_times.groupBy("trip_id").agg(
    spark_min("arrival_time").alias("start_time"),
    spark_max("arrival_time").alias("end_time")
)

trip_duration = first_last.withColumn(
    "duration_minutes",
    (col("end_time").cast("timestamp").cast("long") - col("start_time").cast("timestamp").cast("long")) / 60
)

stop_times = stop_times.withColumn("hour", hour(col("arrival_time")))
trips_per_hour = stop_times.groupBy("hour").agg(count("trip_id").alias("trip_count"))
trips_hour_pd = trips_per_hour.toPandas()
```

Figure 12 Feature Engineering

Travel Time Prediction



```
# Travel time prediction (Linear Regression)

trip_features_pd = trip_duration.select("duration_minutes").dropna().toPandas()
X = trip_features_pd.index.values.reshape(-1, 1)
y = trip_features_pd['duration_minutes'].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
predictions = model.predict(X_test)
mae = mean_absolute_error(y_test, predictions)
print(f"Travel Time Prediction MAE: {mae:.2f} minutes")

plt.figure()
plt.scatter(y_test, predictions)
plt.xlabel("Actual Duration")
plt.ylabel("Predicted Duration")
plt.title("Travel Time Prediction")
plt.show()
```

Figure 13 Travel Time Prediction

Github Repository Link

https://github.com/dailydoseofgithub/bigdataassignment/