# MBTA System Analysis: Identifying and Addressing Transit System Bottlenecks Through Data Analytics

## Introduction

The Massachusetts Bay Transportation Authority (MBTA) operates one of the oldest and most extensive public transit systems in the United States. Serving the Greater Boston area, the MBTA's subway system (commonly known as "the T") is a critical infrastructure component that facilitates over 1.2 million passenger trips on an average weekday. However, the system faces significant operational challenges that impact its reliability and efficiency, particularly during peak hours and adverse weather conditions. These challenges, combined with aging infrastructure and increasing ridership demands, have created a pressing need for innovative solutions to improve service reliability and passenger experience.

### a. Problem Identification

The MBTA system's challenges stem from multiple interconnected factors that significantly impact its daily operations. At the infrastructure level, aging tracks have led to numerous speed restrictions throughout the network, while single-track sections create substantial bidirectional traffic constraints. Platform capacity limitations at key stations further compound these issues, and the existing signal system places additional constraints on train spacing and frequency. These infrastructure limitations create bottlenecks that ripple throughout the entire system, affecting service reliability and passenger satisfaction.

Operational inefficiencies present another significant challenge. The current scheduling system struggles to optimize train frequency during peak hours, and the response to real-time disruptions often lacks the agility needed in a modern transit system. The ability to predict and prevent cascading delays remains limited, and communication of service changes to passengers frequently falls short of expectations. These operational challenges are particularly evident during rush hours when the system operates at maximum capacity.

Weather impact presents a third major challenge for the MBTA system. Severe weather events regularly cause system-wide delays, and the infrastructure shows particular vulnerability to extreme temperatures. The current weather-related contingency planning has proven insufficient to maintain reliable service during adverse conditions, and the system's seasonal adaptation capabilities require significant enhancement. These weather-related challenges affect not only daily operations but also long-term infrastructure maintenance and reliability.

The integration of various data sources poses yet another significant challenge. The MBTA's operational data comes from numerous sources, each with different update frequencies and format requirements. The lack of real-time integration between these systems hampers effective decision-making, while limited historical data analysis capabilities prevent the development of more sophisticated predictive models. The inconsistent quality and format of data across different systems further complicate efforts to implement comprehensive solutions.

### b. Project Scope and Objectives

Our project addresses these challenges through a comprehensive data analytics approach that encompasses several key objectives. The primary goal is to develop a sophisticated real-time analysis system that will transform how the MBTA monitors and manages its operations. This system will integrate multiple data streams, including real-time MBTA operational data, weather information, and historical performance metrics, to provide actionable insights for system operators and managers.

The technical implementation of our solution leverages modern data architecture to create a robust and scalable system. Our approach begins with comprehensive data collection and processing, incorporating real-time data from the MBTA V3 API, historical information from the MBTA Open Data Portal, and weather data from the OpenWeather API. This data undergoes automated cleaning and standardization processes to ensure consistency and reliability in our analyses.

The analytics infrastructure we have designed utilizes cloud-based processing architecture to handle the large volume of data generated by the transit system. Our machine learning pipeline implements sophisticated predictive models, while real-time analytics processing ensures that insights are available to decision-makers when they need them most. The scalable storage solutions we have implemented ensure that both historical and real-time data remain accessible for analysis and reporting.

### c. Expected Outcomes

The implementation of our system is expected to yield improvements across multiple aspects of MBTA operations. In terms of operational efficiency, we anticipate aiding a reduction in average delay times through better prediction and prevention of service disruptions through providing more accurate arrival time predictions and enabling more effective resource allocation during service disruptions. These improvements will be particularly noticeable in schedule adjustments during peak hours and weather events.

## Methodology

### a. API Usage and Data Collection

We obtained the required data for this project from three sources: OpenWeatherMap (OneCall, HistoryBulk), MBTA Blue Book Open Data Portal, and the MBTA V3 API.

- OpenWeatherMap
- MBTA Blue Book Open Data Portal
- MBTA V3 API

In addition to the data sources listed above, we also made use of the General Feed Transit Specification (GTFS) file provided to the public by the MBTA to make better use of its data. This GTFS file contains information regarding all major aspects of the system, including stops, routes, schedules, information about the live data feed (V3 API), and more. This data was

uploaded to our storage bucket under the directory mbta_gtfs and is referenced when we transitioned the historical MBTA data from the silver layer to the gold layer.

**b. Data Processing**

The HistoryBulk download provided us with a .csv file containing weather data from 2023-01-01 to 2023-12-31 at a resolution of 1-hour. This allowed us to insert the data directly into the Bronze layer, where it was then fed into our historical_weather_preprocessing dataflow, which performs the following operations in order:

1. Import batch data
2. Filter out invalid rows for key columns such as temperature, humidity, and dt (datetime/timestamp)
3. Derive new columns for Fahrenheit-based temperature variables, convert dt from UNIX time to a proper UTC timestamp
4. Drop unnecessary columns
5. Send to silver layer as a parquet file to persist changes to data types

The historical transit data obtained from the MBTA Blue Book Open Data Portal was similarly preprocessed using the alt_historical_mbta_data_preprocessing dataflow with the following operations:

1. Import historical data (unrolling was not necessary since the API response had no nested fields)
2. Assign new data types to several columns including service_date, stop_sequence, and e event_time
3. Filter out rows with null values on the three aforementioned columns
4. Select a final batch of columns, excluding those that have been changed or overwritten
5. Send to silver layer as a parquet file to persist changes to data types

After each set of historical data had landed in the silver layer, we proceeded to create two notebooks within our Databricks workspace. These notebooks, each with a “batch-processing” suffix, would serve as areas where we could gain a better understanding of our data and process it into the gold layer, creating several aggregated datasets in the process. In addition, we utilized Databricks notebooks to perform exploratory data analysis on our data, this will be covered at the beginning of the following subsection.

For our historical MBTA data, the first operation was to match each entry’s stop_id value with a human readable stop_name value, which we retrieved from the stops.txt file in our mbta_gtfs directory. This allowed us to provide a name which riders might recognize (‘Boston University East’) instead of a series of numbers denoting a particular stop’s id within the GTFS file (70146/70147). Following this, we created a timestamp column derived from the event_time_sec column. Finally we calculated dwell time (dwell_time_seconds), which represents the period of time between when the same train is listed as arriving (“ARR”) and departing (“DEP”) a given station; time between arrival messages at the same station was used to calculate headway (headway_seconds). After we derived the necessary features, we

proceeded to create a series of aggregations: by day, by hour (to later match our weather data), by route, and by stop. These aggregations, along with the original dataset with modified features, were then saved to our gold layer.

For our historical weather data, a limited amount of batch processing was required. We derived a date_time_est column, which converted the UTC time that is standard with OpenWeatherMap downloads to EDT/EST. This would allow us to accurately match Boston's weather to the MBTA transit events. We then created the following weather aggregations: daily, wind by condition, and heavy rain. However, since the resolution of our weather data was already at 1-hour intervals, we felt further aggregation was not quite necessary and would detract from our ability to predict headway and dwell time with any granularity.

### c. Data analysis

#### i. Exploratory Data Analysis (Databricks)

Our analysis began with a simple exploratory data analysis of our silver layer data for both OpenWeatherMap and the MBTA in a databricks notebook.

The dataset comprises 23,987,198 records, containing detailed information about service dates, routes, trips, stops, and various event types. The data schema includes temporal attributes (service_date, event_time), spatial components (route_id, stop_id, stop_sequence), and operational parameters (vehicle_id, event_type, direction_id).

The distribution of event types (Figure 1) shows that the majority of events are either arrivals (ARR) or departures (DEP), with approximately 12.1 million and 11.6 million events respectively. Predicted arrivals (PRA) and predicted departures (PRD) constitute a smaller portion of the dataset, with about 216,000 and 119,000 events respectively. This distribution suggests that the system primarily records actual transit events rather than predictions.

Route utilization analysis (Figure 2) indicates that the Green Line is the most heavily utilized, with its four branches (E, D, B, and C) accounting for the highest number of events. The Green-E branch leads with approximately 4 million events, followed closely by Green-D and Green-B with about 3.7 million events each. The Red and Orange lines show similar usage patterns with roughly 2.9 million events each, while the Blue Line and Mattapan Trolley record fewer events, at 2.3 million and 1.1 million respectively.

The temporal analysis of events across stop sequences (Figure 3) shows patterns in service timing. The average event time fluctuates between 52,000 and 56,000 seconds (approximately 14.4 to 15.5 hours), with notable variations across different stop sequences. Early sequences show relatively stable timing, but there are periodic spikes and dips throughout the route progression, possibly indicating scheduled holding points or typical delay patterns.

Daily operational patterns (Figure 4) demonstrate peak service hours between 9:00 and 16:00, with event counts consistently around 2.5-3 million during these hours. The system shows lower activity during early morning (8:00) and late afternoon (17:00) hours, suggesting reduced service frequency during these periods.

The distribution of event timing (Figure 5) shows the spread of event occurrences throughout the day. The box plot indicates that most events occur between approximately 40,000 and 70,000 seconds (11.1 to 19.4 hours) from the start of service, with some outliers extending beyond these bounds. This distribution helps understand the typical service window and identify potential anomalies in operation timing.

The correlation analysis between stop sequence and event timing (Figure 6) shows a weak positive correlation (0.018), suggesting that event times are largely independent of a stop's position along the route. This independence indicates that the MBTA system maintains relatively consistent service patterns regardless of a stop's location in the sequence.

An examination of event distribution by route and time of day (Figure 7) shows distinct operational patterns across different lines. The heatmap visualization shows higher event concentrations during mid-day hours (9:00-16:00) across all routes, with the Green Line branches exhibiting particularly intense activity. The Blue and Mattapan lines show notably lighter activity patterns, consistent with their lower overall event counts.

Weather conditions play a significant role in transit operations. The temperature distribution (Figure 8) shows a roughly normal distribution centered around 40-60°F, with tails extending into both extreme cold and heat conditions. This temperature range represents typical operating conditions for the system.

Analysis of weather patterns by condition type (Figure 9) shows interesting relationships between different meteorological parameters. Thunderstorm conditions are associated with both high temperatures (70.7°F) and high humidity (88.2%), while snow events correlate with the lowest average temperatures (31.6°F). Clear conditions show moderate temperature ranges (53.1°F) with lower humidity levels (64.9%), suggesting optimal operating conditions.

The examination of extreme weather events throughout the year (Figure 10) shows seasonal challenges for transit operations. Winter months (December through February) show significant low-temperature days, with February experiencing the highest count of extreme cold events. Extreme rain events cluster in summer months (July-September), while high-temperature days are relatively rare, suggesting that the system operates primarily in moderate temperature conditions.

### ii. Gold Layer Analysis

Our analysis of the Gold Layer

Note, that although stop_id provides us with the ability to distinguish between specific platforms, stop_name does not do so. Therefore the aggregation by stop_name will reflect the average headway of each train that passes through the entire station (for example both Green and Blue line trains at Government Center), instead of any specific route (Green Line-B or Green Line-C). This is particularly relevant for all Green Line stations further “inbound” than Kenmore, as multiple lines converge in the central subway, our calculations of headway do not distinguish between Green Line trains of different routes. However, given that the struggles of the central

subway are well documented, and subterranean stations being relatively isolated from inclement weather issues, this aggregation method is not of particular concern.

#### iii. Gold Layer Modeling

### d. Final Architecture

Our final architecture consisted of the following steps:

1. Data Ingestion using Azure Data Factory Pipelines
2. Data Cleaning and Initial Preprocessing using Azure Data Factory Dataflows
3. Initial Data Exploration using Azure Databricks notebooks
4. Additional Preprocessing Performed using Azure Databricks notebooks
5. Model Training (initially attempted in Azure ML Studio) using Azure Databricks notebooks
6. Visualization of Data Exploration and Analysis using PowerBI

## Results and Findings

### a. Machine Learning Results

For reasons which are covered in our conclusion under the Challenges Faced subsection, we were unable to run automated ML jobs using the Azure ML studio. As a result, we chose to migrate our ML processes onto Databricks. With the more limited ML environment of Databricks, we choose to run 4 models: LinearRegression, DecisionTreeRegressor, RandomForestRegressor, and GBTRegressor (Gradient Boosted Tree). Out of the 4 models, the GBTRegressor consistently outperformed the remaining models that we tested. We tested all four models against the following combinations of target variables and datasets:

- Headway for all routes

GBTRegressor achieved an RMSE of 83.51 and an $R^2$ of 0.6710, outperforming all other models. The DecisionTreeRegressor came closest, with an RMSE of 98.09 and an R2R^2R2 of 0.5462. The GBTRegressor's execution time of 28.12s is higher than others, but its accuracy trade-off justifies the computational cost.

- Headway for each route: Green-B, Red, Orange, Blue

Green Line-B: GBTRegressor achieved an RMSE of 79.36 and an $R^2$ of 0.3697, outperforming other models, though the $R^2$ values suggest room for improvement in predictive accuracy for this route.

Red Line: The GBTRegressor had an RMSE of 90.95 and an $R^2$ of 0.6143, a marked improvement over LinearRegression and RandomForestRegressor.

Blue Line: GBTRegressor stood out with an RMSE of 75.79 and $R^2$ of 0.7734, indicating strong performance and accurate predictions.

Orange Line: With an RMSE of 80.83 and $R^2$ of 0.6240, GBTRegressor again provided the best performance among all models.

- Dwell Time for each route (excluding Mattapan due to ridership differences)

The GBTRegressor excelled with an RMSE of 3.18 and an $R^2$ of 0.8082, indicating excellent predictive accuracy. This result confirms its capability to model more granular target variables effectively.

The GBTRegressor consistently provided better results across all dataset and target variables. However, it is a significantly more complex model that requires more computational resources to achieve its accuracy.

### b. Weather Impact

The weather features used to predict headway and dwell time did not appear to have significant predictive power when attempting to estimate across all routes. However, when we narrow the available data down by route_id, we can see patterns of weather variable impacts start to emerge.
Figures 15-18 detail the most predictive features for headway for the Green Line B-Branch, Red Line, Orange Line, and Red Line, respectively. In each figure, we can notice the presence of pressure, humidity, temp_F (or max or min), and wind_speed. The consistent presence of weather variables among the top 10 most important predictors for headway highlights their significant influence on transit operations. These factors can directly and indirectly impact headway by affecting vehicle performance, passenger boarding and alighting times, and overall traffic conditions. For example, pressure and humidity can influence precipitation, which may slow down transit vehicles or disrupt schedules. Temperature (temp_F) can affect rider behavior, such as increased ridership during extreme weather, potentially leading to delays. Meanwhile, wind_speed can impact vehicle stability and travel speeds, especially in exposed areas.

## Conclusion

### a. Challenges Faced

#### i. Issues with Sourcing Data

Initially, we had chosen to use OpenWeatherMap data similar to that used as part of this course. However, it was later discovered that student accounts do not have the subscriptions necessary to query weather data that was farther back by more than a year. In search of a solution, we contacted OpenWeatherMap to determine the right product which would suit our needs. Unfortunately, this process took longer than expected. We had two products to choose from: the OneCall API and the HistoryBulk download. The OneCall API had limited free use of 1,000 calls/day while the HistoryBulk was a one-time purchase of $10. In order to source our data for free, we would require approximately 8 days to source the data we needed from the OneCall API. Additionally, this timeline does not include any free calls spent on development and debugging our pipelines. As such, after consulting with the Teaching Assistants regarding our situation late in the semester, it was decided that the safest option would be to purchase the

HistoryBulk for $10. This allowed us to guarantee meeting the deadline for deliverables and avoid the possibility of being charged for the OneCall API and not receive the necessary data.

### ii. Issues with Azure ML and Permissions

During the implementation of our big data workflow, we encountered several significant challenges that impacted our project timeline and required adaptive solutions. The primary obstacles emerged from our interaction with Azure Machine Learning services, which presented several critical technical hurdles throughout the development process.

One of the most persistent challenges involved environment stability issues within the Azure ML platform. Our team experienced multiple instances of notebook losses during development, which significantly disrupted our workflow and resulted in lost progress. These interruptions were compounded by persistent difficulties in maintaining consistent access to compute resources, leading to unexpected platform behavior that frequently impeded our development progress.

Permission and access control complications created another layer of complexity in our project implementation. Initially working under the professor's subscription, our team faced recurring access limitations that hindered our ability to work efficiently. When attempting to create within the Azure ML Workspace, we encountered server-side errors that prevented the creation of new .ipynb files. The recurring error message "File 'Notebook.ipynb' was not created to user directory due to server-side error' became a significant obstacle in our development process.

To address these challenges, we attempted to migrate to a personal Azure account and establish new compute resources. While we initially succeeded in creating these resources, subsequent attempts to activate them proved unsuccessful. This created a significant bottleneck in our development pipeline and required us to develop alternative approaches to our workflow.

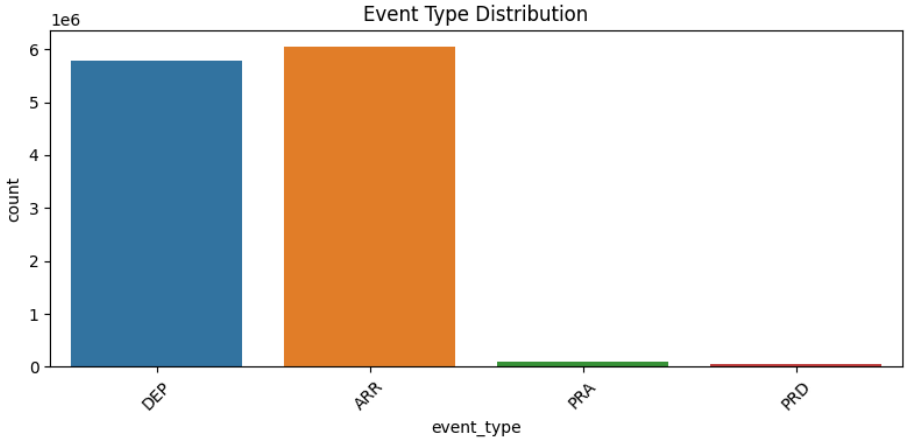
These challenges provided some valuable lessons for future big data implementations. We learned the importance of establishing redundant development environments early in the project lifecycle and maintaining comprehensive documentation of environment configurations. Furthermore, this experience showed the necessity of building in additional time buffers when working with cloud services to account for potential infrastructure challenges.

## b. Summation of Findings and Final Recommendations

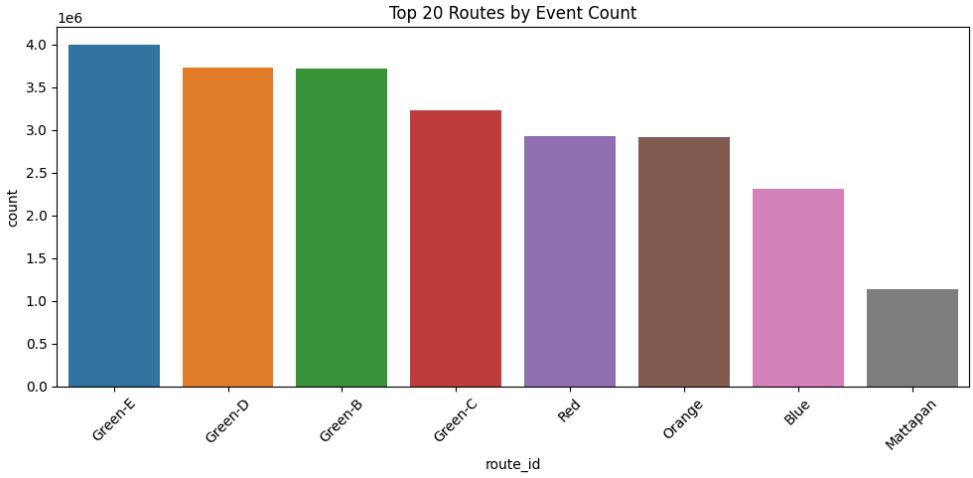
Although headway and dwell time largely do not vary greatly from the predetermined times set by the schedule, operators, and dispatchers of the MBTA, we can note that our models find predictive power within the provided weather features. In particular, humidity and pressure, factors that influence precipitation, are particularly important. Additionally, when drilling down to route level, temperature's importance increases, reflecting changes in ridership or system functionality that correlate with temperature to a certain level. These findings indicate that there is a measurable impact of weather on the system's operation, and potential for positive improvement if real-time weather conditions could be incorporated into scheduling or predictive functions.

# Appendix

## Figure 1




## Figure 2




## Figure 3

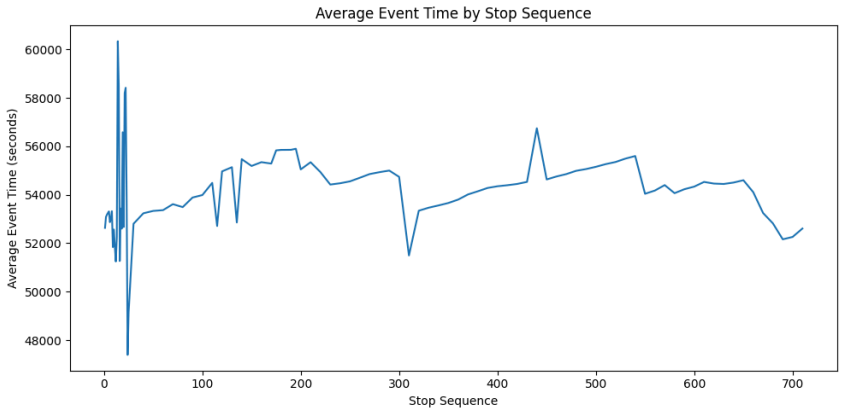



**Figure 4**

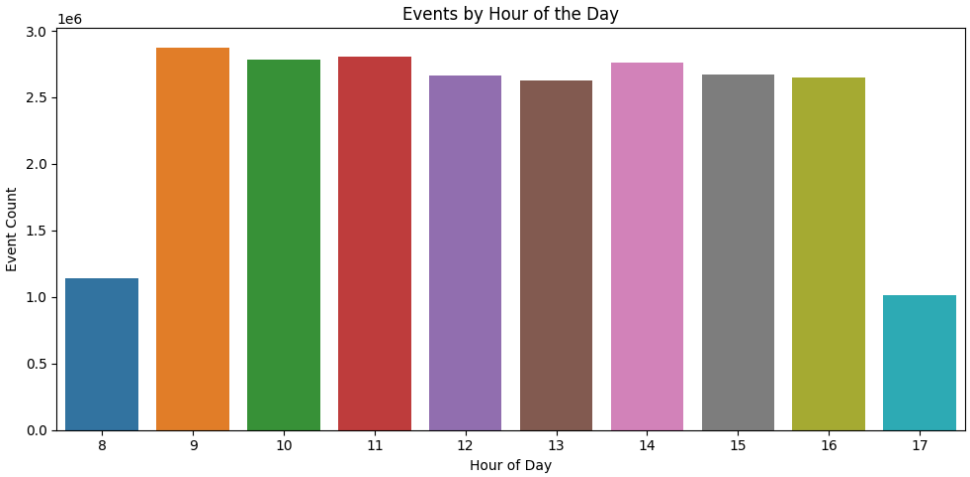



**Figure 5**

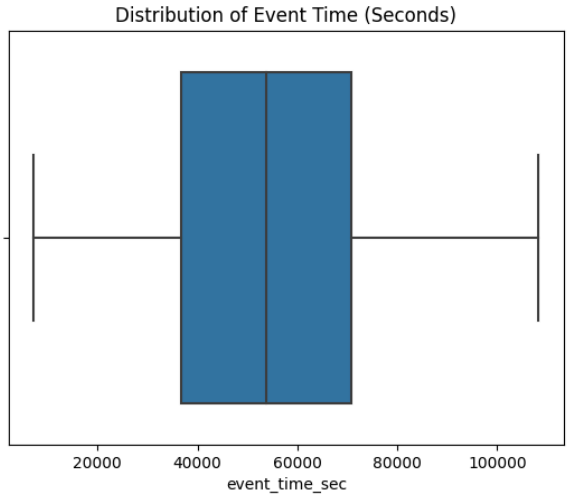


**Figure 6**

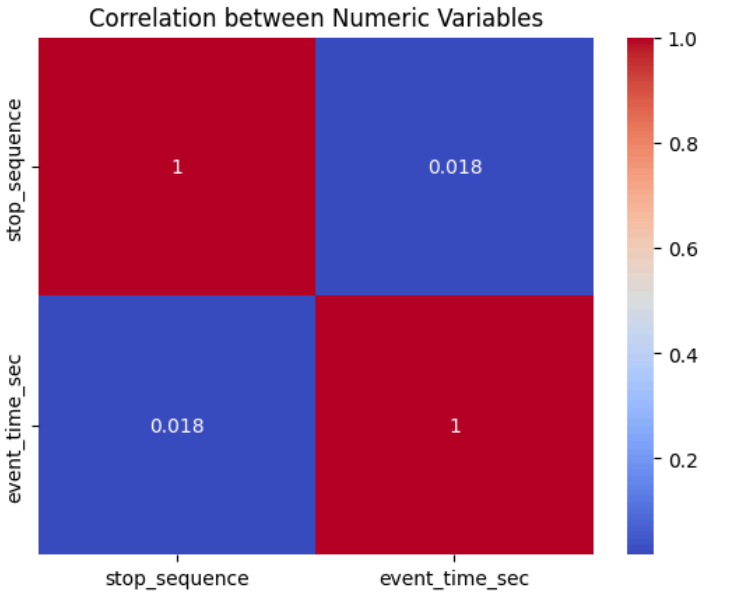



**Figure 7**

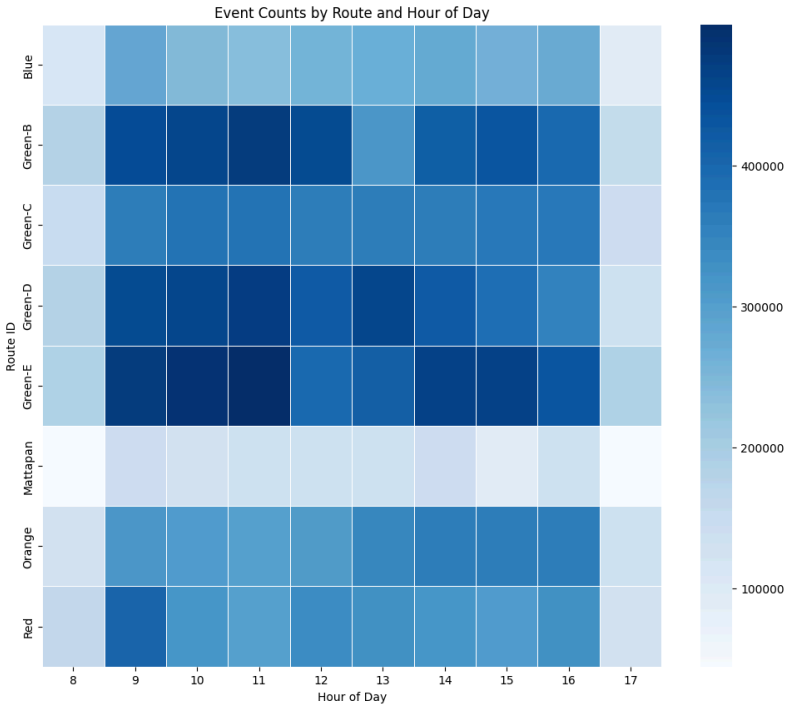



**Figure 8**

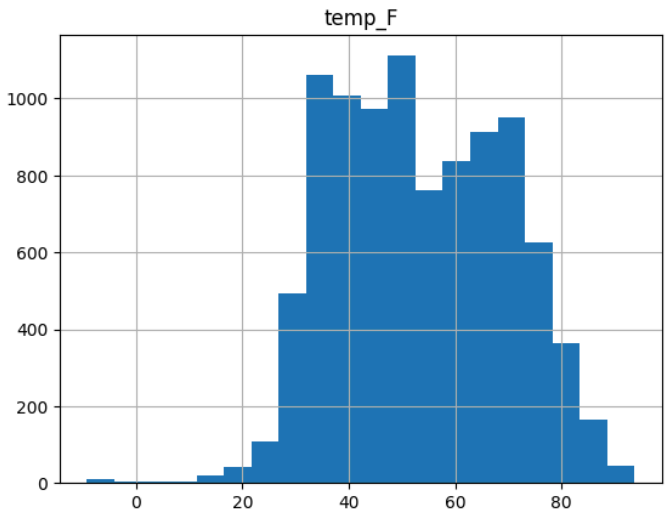



**Figure 9**

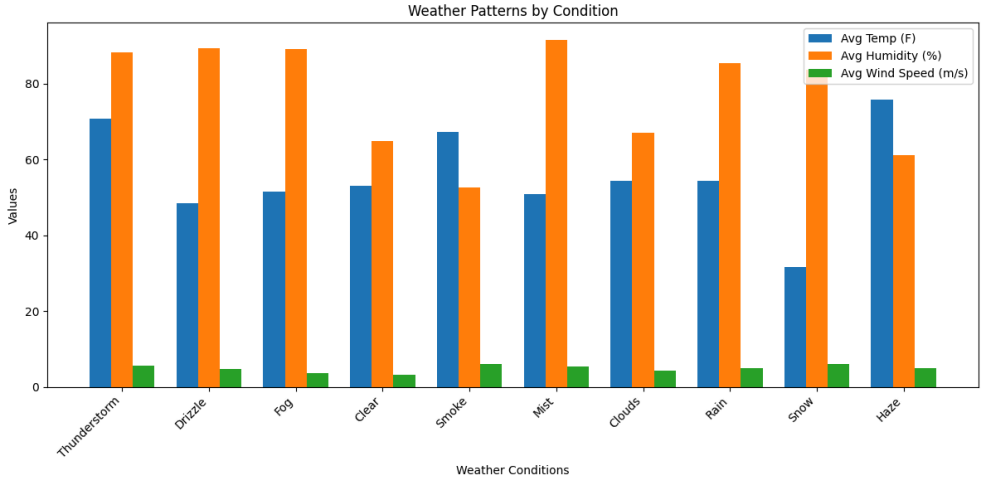
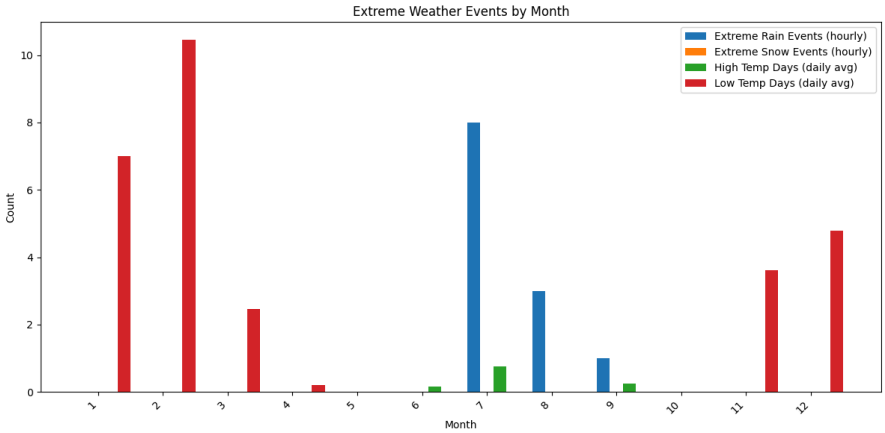



**Figure 10**




## Headway

```
=== LinearRegression ===
RMSE: 114.4110
R²: 0.3826
MAE: 79.6492
MSE: 13089.8760
Execution Time: 6.29s

=== DecisionTreeRegressor ===
RMSE: 98.0917
R²: 0.5462
MAE: 68.3385
MSE: 9621.9850
Execution Time: 7.40s

=== RandomForestRegressor ===
RMSE: 106.7664
R²: 0.4623
MAE: 75.9735
MSE: 11399.0551
Execution Time: 9.79s

=== GBTRegressor ===
RMSE: 83.5139
R²: 0.6710
MAE: 57.6826
MSE: 6974.5728
Execution Time: 28.12s
```

**Figure 11**

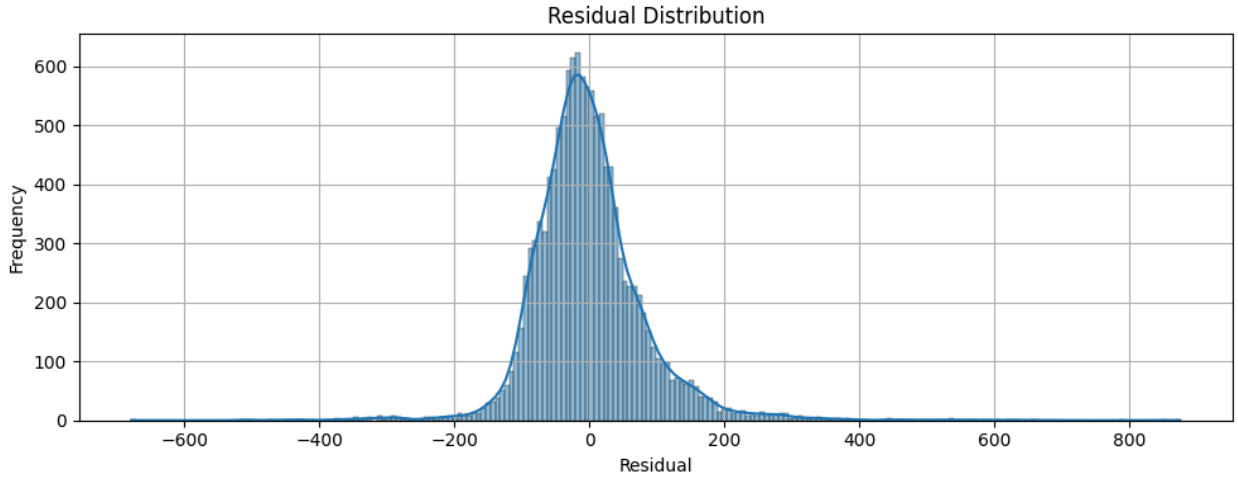
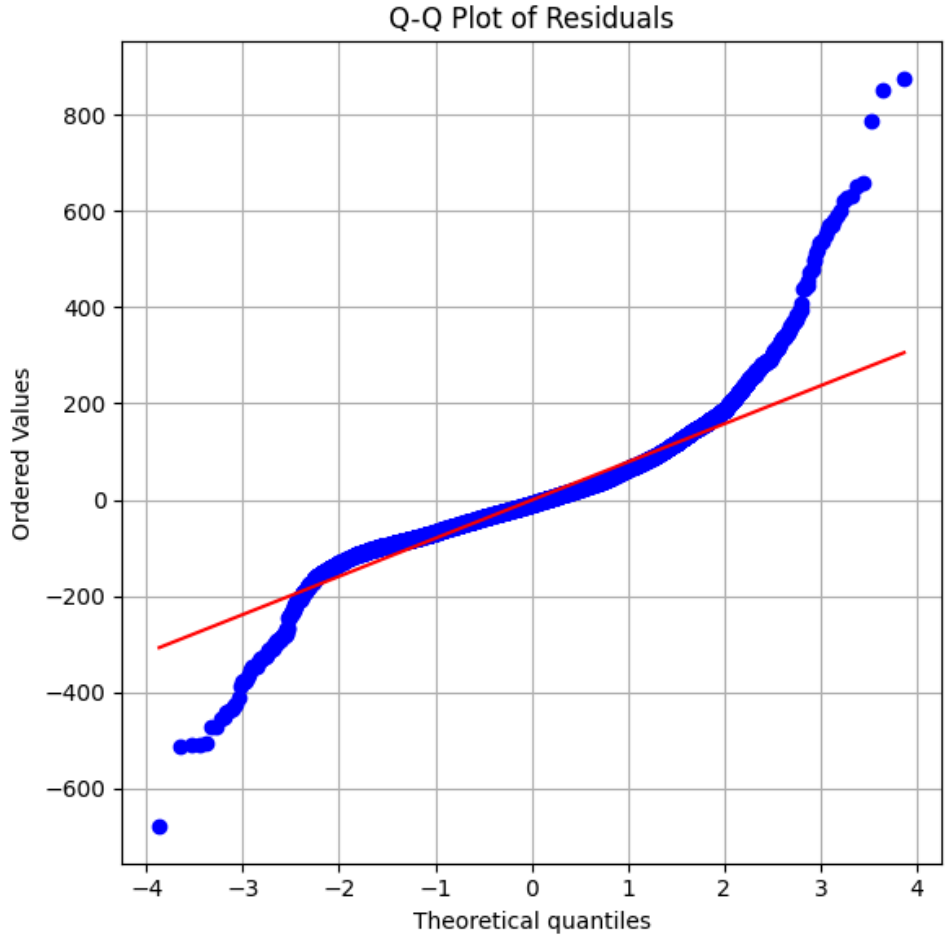



**Figure 12**




**Dwell Time**

```
=== LinearRegression ===
RMSE: 4.7245
```

R²: 0.5764
MAE: 3.3827
MSE: 22.3206
Execution Time: 5.60s

=== DecisionTreeRegressor ===
RMSE: 4.0912
R²: 0.6823
MAE: 2.9376
MSE: 16.7382
Execution Time: 7.11s

=== RandomForestRegressor ===
RMSE: 4.3132
R²: 0.6469
MAE: 3.0567
MSE: 18.6033
Execution Time: 9.29s

=== GBTRegressor ===
RMSE: 3.1795
R²: 0.8082
MAE: 2.3843
MSE: 10.1089
Execution Time: 27.65s

**Figure 13**

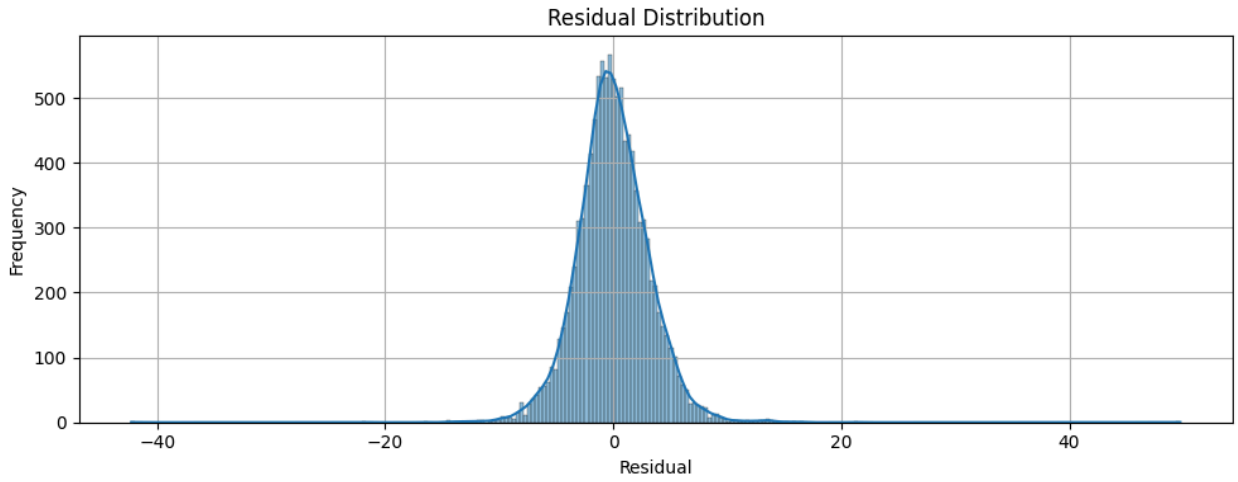
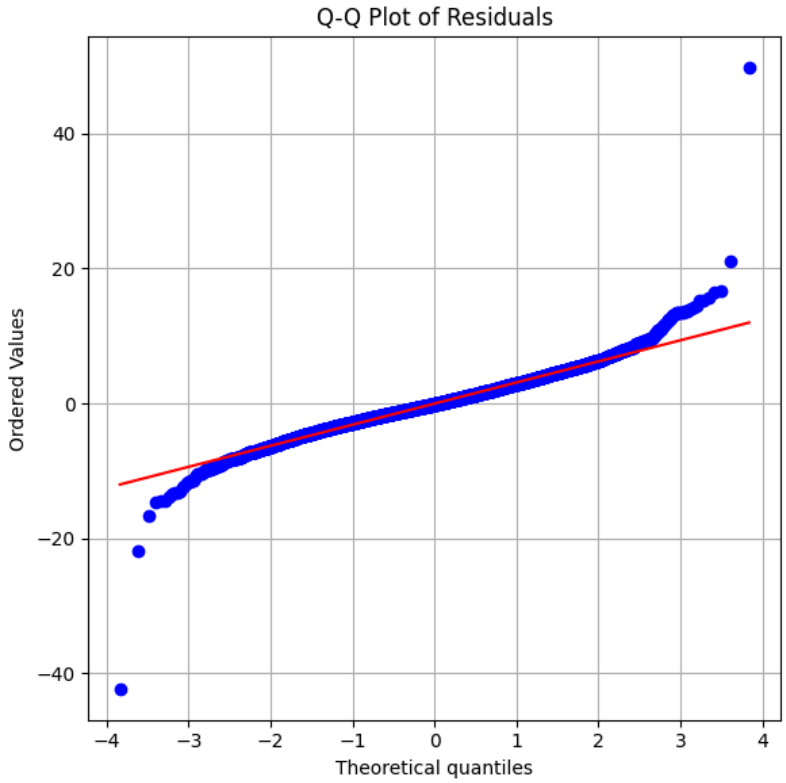


**Figure 14**

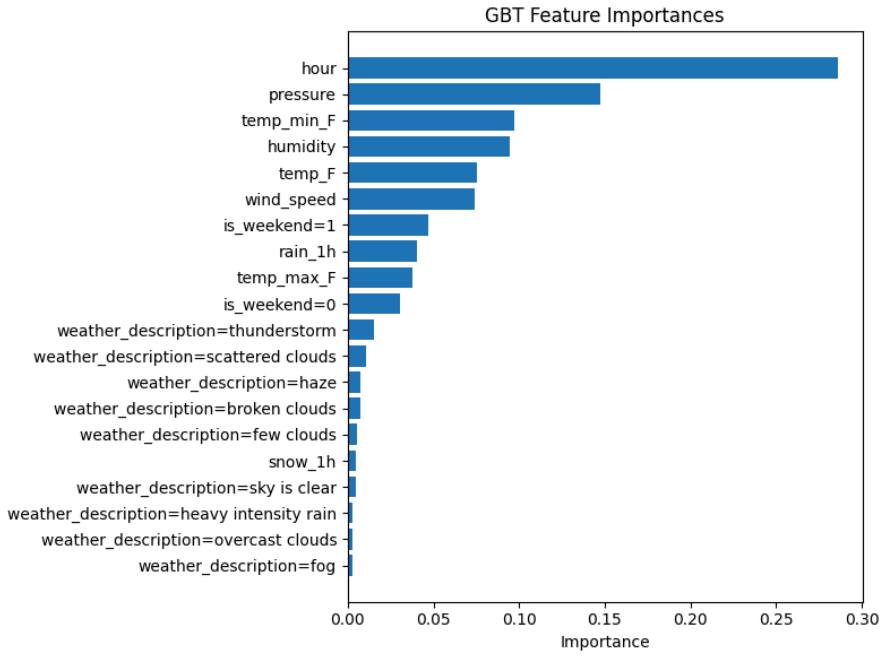



**Green Line-B**

```
=== LinearRegression ===
RMSE: 95.5671
R²: 0.0860
MAE: 61.8482
MSE: 9133.0669
Execution Time: 3.99s

=== DecisionTreeRegressor ===
RMSE: 83.5634
R²: 0.3012
MAE: 53.9265
MSE: 6982.8464
Execution Time: 4.09s

=== RandomForestRegressor ===
RMSE: 84.7987
R²: 0.2803
```

```
MAE: 55.8754
MSE: 7190.8197
Execution Time: 5.56s

=== GBTRegressor ===
RMSE: 79.3622
R²: 0.3697
MAE: 50.6941
MSE: 6298.3556
Execution Time: 22.02s
```

**Figure 15**




**Red Line Predictions**

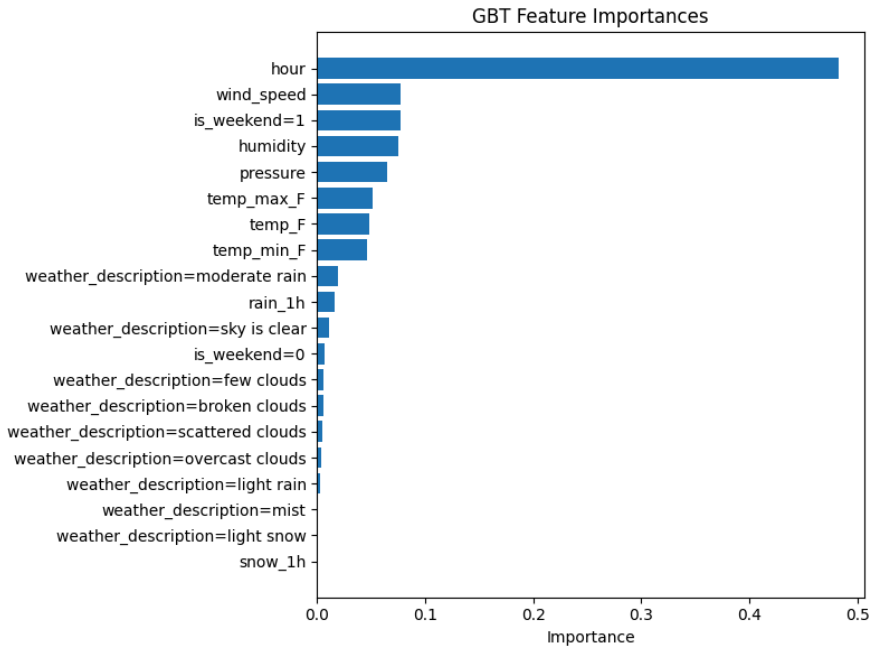
```
=== LinearRegression ===
RMSE: 137.8151
R²: 0.1145
MAE: 81.8501
MSE: 18992.9921
Execution Time: 11.48s

=== DecisionTreeRegressor ===
RMSE: 93.9119
```

R²: 0.5888
MAE: 58.4058
MSE: 8819.4366
Execution Time: 5.57s

=== RandomForestRegressor ===
RMSE: 108.0580
R²: 0.4556
MAE: 66.3349
MSE: 11676.5323
Execution Time: 6.97s

=== GBTRegressor ===
RMSE: 90.9492
R²: 0.6143
MAE: 56.0657
MSE: 8271.7530
Execution Time: 22.85s

**Figure 16**

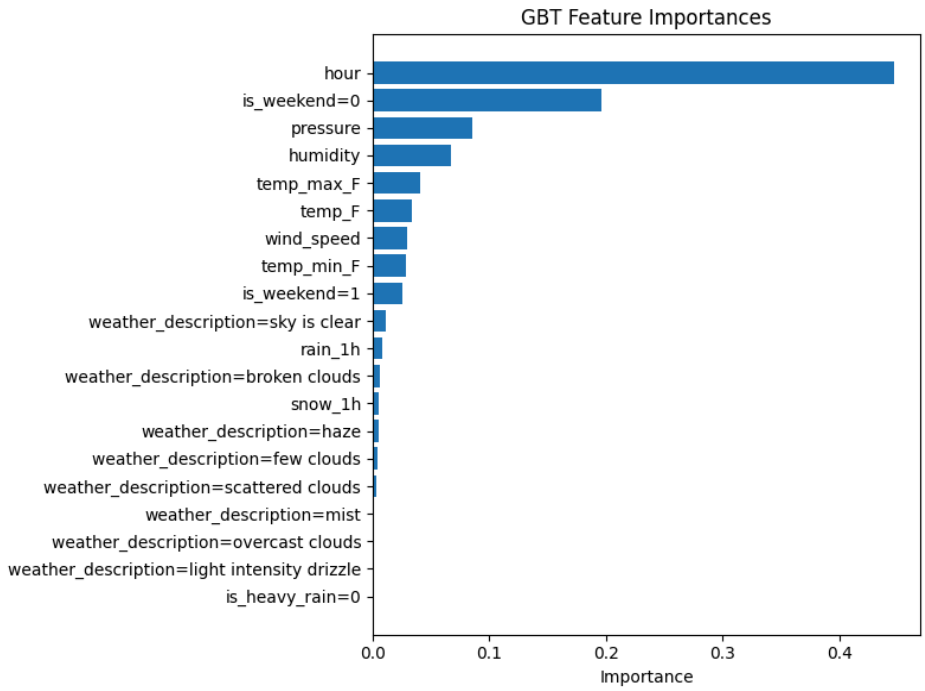



**Blue Line**

=== LinearRegression ===

```
RMSE: 142.2549
R²: 0.2017
MAE: 112.4966
MSE: 20236.4543
Execution Time: 4.43s

=== DecisionTreeRegressor ===
RMSE: 82.6716
R²: 0.7304
MAE: 56.9087
MSE: 6834.5921
Execution Time: 4.09s

=== RandomForestRegressor ===
RMSE: 101.8804
R²: 0.5905
MAE: 76.1340
MSE: 10379.6153
Execution Time: 5.17s

=== GBTRegressor ===
RMSE: 75.7906
R²: 0.7734
MAE: 52.5114
MSE: 5744.2199
Execution Time: 22.12s
```

**Figure 17**




**Orange Line**

```
=== LinearRegression ===
RMSE: 122.3618
R²: 0.1384
MAE: 82.4941
MSE: 14972.4140
Execution Time: 3.94s

=== DecisionTreeRegressor ===
RMSE: 86.5423
R²: 0.5690
MAE: 67.4522
MSE: 7489.5781
Execution Time: 4.44s

=== RandomForestRegressor ===
RMSE: 96.1074
R²: 0.4685
MAE: 73.5578
MSE: 9236.6375
Execution Time: 5.65s
```

```
=== GBTRegressor ===
RMSE: 80.8317
R²: 0.6240
MAE: 60.8381
MSE: 6533.7703
Execution Time: 22.38s
```

**Figure 18**

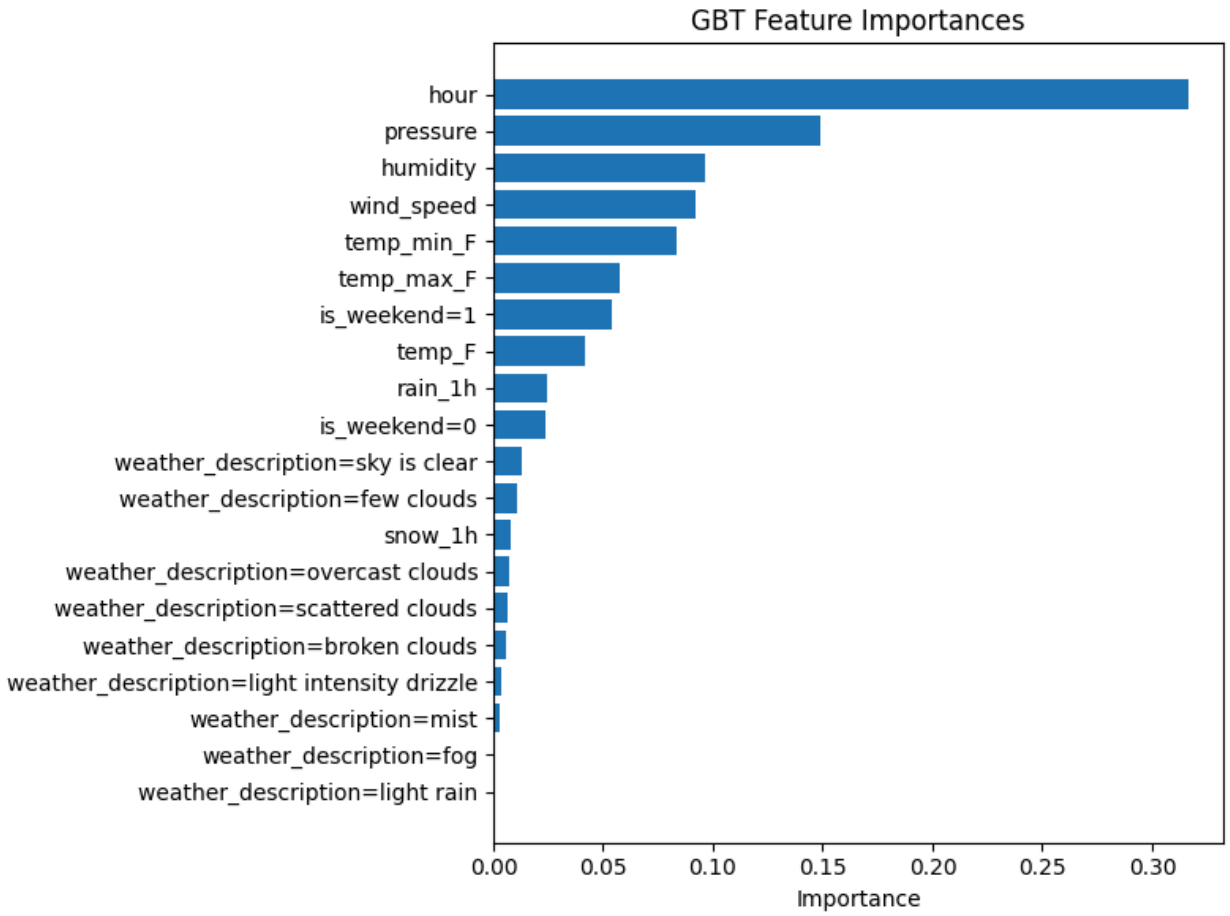


**Figure 19**

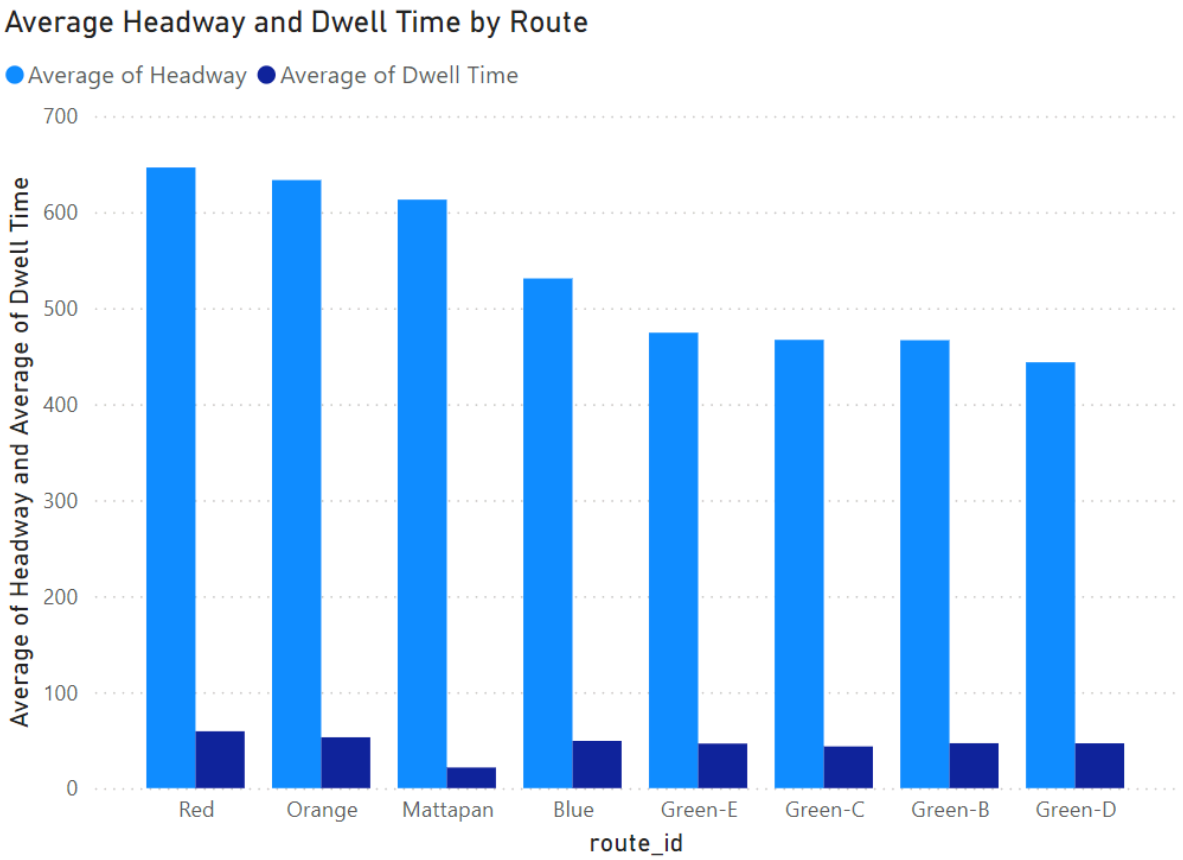



**Figure 20**

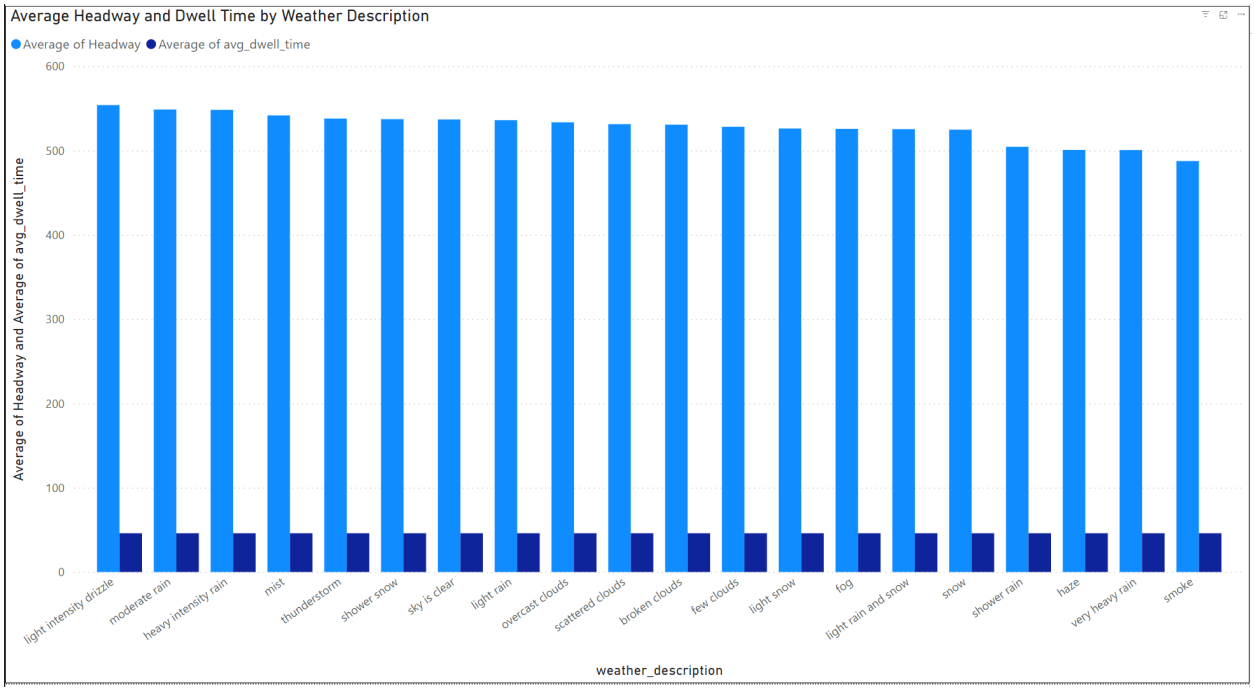


**Table 1: Correlation Matrix of Weather Parameters**

| Parameter | Temperature | Humidity | Pressure | Wind Speed |
|---|---|---|---|---|
| **Temperature** | 1.000 | -0.048 | -0.207 | -0.175 |
| **Humidity** | -0.048 | 1.000 | -0.271 | -0.134 |
| **Pressure** | -0.207 | -0.271 | 1.000 | -0.245 |
| **Wind Speed** | -0.175 | -0.134 | -0.245 | 1.000 |

**Note:** Values represent Pearson correlation coefficients between pairs of weather parameters. All correlations are relatively weak ($|r| < 0.3$), indicating limited linear relationships between these weather variables.