Seminar report

Modeling Concept Drift in New York City's Subway System: Online and Offline Detection Approaches

Sanish Gurung

July 28, 2025

Faculty of Science
University of Helsinki

Contact information

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki,Finland

Email address: info@cs.helsinki.fi
URL: http://www.cs.helsinki.fi/

Use of Artificial Intelligence (AI): I have used AI tools in the following ways:

1. I checked spelling errors and english fluency check like "something ..., something" to see if they sounded better or to correct my spelling. Moreover, I used keywords or expressions to check for grammatical correctness.

Note: I want to complete the course ‘Concept Drift in Data Science’ as normal course (project + exam) and not as seminar. I had a problem replacing ‘Seminar Report’ with ‘Project Report’ in this report (maybe Computer Science template does not allow it).

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Study programme
Faculty of Science	

Tekijä — Författare — Author

Sanish Gurung

Työn nimi — Arbetets titel — Title

Modeling Concept Drift in New York City's Subway System: Online and Offline Detection Approaches

Ohjaajat — Handledare — Supervisors

Prof. Indre Zliobaite

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Seminar report	July 28, 2025	17 pages

Tiivistelmä — Referat — Abstract

This project investigates concept drift in New York City subway traffic data from 2017 to 2021. It aims to uncover seasonal trends and detect sudden shifts in mobility behavior caused by the COVID-19 pandemic. The study proposes both online learning and offline analysis approaches for drift detection.

This project uses a publicly available dataset from Kaggle, containing turnstile entry and exit counts recorded at 4-hour intervals across multiple boroughs and neighborhoods. High-level features such as 'Total Traffic' and 'Netflow' were derived to capture key movement patterns. Temporal aggregation enabled analysis at daily level, revealing significant disruptions during 2020 and 2021. Spatial filtering was applied to focus on regions of particular interest and assess model performance.

Detection methods include regression-based models using adaptive full retraining and fixed-length rolling windows. Additionally, streaming windowing techniques such as ADWIN and KSWIN were implemented. Results indicate that daily granularity is more effective in detecting abrupt changes. A comparative analysis of various models further highlights their strengths and weaknesses.

ACM Computing Classification System (CCS)
Computing Methodologies → Machine Learning → Learning Paradigms → Unsupervised Learning
Computing Methodologies → Machine Learning → Learning Settings → Online Learning Settings

Avainsanat — Nyckelord — Keywords

concept drift, time series analysis, online learning, adaptive models, urban mobility, COVID-19

Säilytyspaikka — Förvaringsställe — Where deposited

Helsinki University Library

Muita tietoja — övriga uppgifter — Additional information

Concept Drift in Data Science

Contents

1 Introduction

In today's data-driven world, many governments, industries, and organizations continuously generate massive volumes of streaming data. This surge in data availability has increased the demand for robust analytics and machine learning methods which can handle dynamic environments, and deliver accurate and real-time predictions. However, a challenge that arises in these non-stationary environments is concept drift, a phenomenon where the statistical properties of the target variable change over time in unexpected ways, degrading the performance of previously trained models. [4]

Concept drift poses a significant challenge to the reliability of predictive models in dynamic environments, where the relationship between input data and outcomes can evolve over time. This issue is prevalent in real-world applications such as medical diagnostics and models of future climate, where changing behaviors or system properties render historical patterns obsolete. If not properly addressed, concept drift can lead to degraded model performance and flawed decision-making. In data-driven systems, continuously adapting to these shifts is critical to maintain relevance and accuracy over time [7].

In urban transportation systems, continuous data streams are generated, making them ideal candidates for time series and adaptive learning methods. These data streams contain different observations, such as energy emission of the vehicle, and turnstile-based entries and exits. When external factors remain stable over time, data points tend to follow repetitive, seasonal patterns, allowing us to understand recurring traffic behaviors. However, unprecedented events such as the COVID-19 pandemic can greatly alter human behavior, such as changing means of transportation [3]. Therefore, analyzing human traffic behavior and exploring adaptive and predictive modeling techniques offers an excellent research opportunity to investigate how well these methods respond to dynamic and evolving real-world conditions.

The project report is structured as follows: In Chapter 2, the dataset used in this project is introduced and the rationale behind selecting key features for analysis and modeling purposes is explained. In Chapter 3, we explore human traffic behavior using spatial and temporal components. Chapter 4, examines which temporal granularity is most suitable for our analysis and modeling. In Chapter 5, various drift detection methods are explored in greater detail. Lastly, Chapter 6 highlights general conclusions of this project report.

2 Dataset and Feature Overview

This chapter introduces the dataset and explains the rationale behind selecting the most relevant features for time series analysis and machine learning modeling.

For this project, I used the 'NYC Subway Traffic 2017–2021' dataset, available at https://www.kaggle.com/datasets/eddeng/nyc-subway-traffic-data-20172021?select=NYC_subway_traffic_2017-2021.csv. The dataset contains turnstile-based counts of entries and exits recorded at 4-hour intervals in NYC subway.

2.1 Pre-existing Features

The 'NYC Subway Traffic 2017–2021' dataset contains 17 features. Table 2.1 and 2.2 describe each feature and show its corresponding number of unique values.

Features	Description	Unique Values
Unique ID	A custom identifier key corresponding to a unique combination of stop name, remote unit, line and connecting lines	469
Datetime	Timestamp *	9911
Stop Name	Station Name	345
Remote Unit	Higher-level hierarchical identifier	439
Line	The line that 'Stop Name' belongs to	45
Connecting Lines	Lines that pass through this station	103
Daytime Routes	Lines that pass through this station during daytime	71

Table 2.1: Description of Features (Part 1)

*'Datetime' is the 4 hour window timestamp in which data is collected between 2 hours before and 2 hours after the marked timestamp

†Dataset excludes Staten Island (one of the Borough) in New York City, leading to only 4 Boroughs

‡Line incorporates 'Remote Unit', 'Connecting Lines', 'Daytime Routes', and 'North/South Direction Labels'

Features	Description	Unique Values
North Direction Label	Destination of north-going lines	45
South Direction Label	Destination of south-going lines	42
Division	Line belonging to unique subway system divisions	3
Structure	Structure of the station	5
Borough	Administrative divisions of New York City†	4
Neighborhood	Neighborhood the stop is in	51
Latitude	Geographic coordinate specifying north-south or vertical position on Earth	440
Longitude	Geographic coordinate specifying east-west or horizontal position on Earth	440
Entries	Total number of entries recorded by turnstile device in 4 hour window	21752
Exits	Total number of exits recorded by turnstile device in 4 hour window	19151

Table 2.2: Description of Features (Part 2)

2.2 Derived Features

To facilitate high-level temporal analysis and concept drift detection, two additional features were derived from the original data: 'Total Traffic' and 'Netflow'.

The `Total Traffic` feature represents the total number of passengers either entering or exiting a station:

$$\texttt{Total Traffic} = \texttt{Entries} + \texttt{Exits} \tag{2.1}$$

This captures overall foot traffic at a given location within a specified time window.

The `Netflow` feature is defined as the net change in occupancy at a station:

$$\texttt{Netflow} = \texttt{Entries} - \texttt{Exits} \tag{2.2}$$

A positive net flow suggests crowd buildup, whereas a negative net flow may indicate clearing out of the station.

2.3 Feature Selection

For both analysis and modeling, the following features were selected due to their clarity, interpretability, and relevance to temporal and spatial patterns:

- `Unique ID`
- `Stop Name`
- `Datetime`
- `Borough`
- `Neighborhood`
- `Entries`
- `Exits`
- `Total Traffic`
- `Netflow`

Other features were excluded due to limited documentation, ambiguous semantics, or minimal relevance to the goals of concept drift analysis and time series modeling.

In the next chapter, we conduct exploratory data analysis (EDA) to investigate subway traffic distribution across boroughs and neighborhoods, analyze temporal trends over multiple years, and examine daily ridership patterns at key stations.

3 Exploratory Data Analysis

This chapter focuses on understanding human presence across neighborhoods and boroughs. Furthermore, it examines population density trends in selected neighborhoods over five years and explores general human traffic behavior on specific days.

3.1 Preliminary Observations

Fig. 3.1 illustrates the distribution of subway traffic across boroughs. It is evident that Manhattan and Brooklyn are the busiest, while Queens and the Bronx experience approximately half the traffic of those major boroughs.

Fig. 3.2 shows the 10 busiest (in cyan) and 10 least busy (in red) neighborhoods in the dataset. Most of the busiest neighborhoods are located in Manhattan, whereas the least busy ones (shown in red) are primarily in the other boroughs.

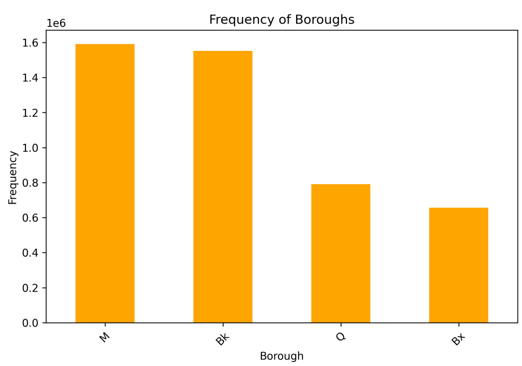

Figure 3.1: Frequency of Borough

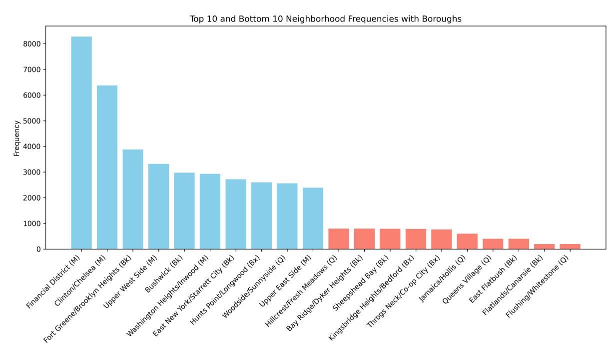

Figure 3.2: Frequency of Neighborhoods

3.2 Trends

Fig. 3.3 shows the distribution of total traffic in the five busiest neighborhoods over five years for the month of July. The figure indicates an increasing number of riders from 2017 to 2019 due to improvement in the infrastructures. However, from 2020, ridership dropped by more than half compared to 2019, with a continuing downward trend observed into 2021, reflecting the onset and impact of the COVID-19 pandemic [2].

Fig. 3.4 shows the distribution of entries, exits, total traffic, and netflow at the busiest station, 'Grand Central - 42 St', from 2019 to 2021 on May 1st. The years 2017 and 2018 are omitted because their patterns were nearly identical to 2019. Over the three-year span, the general trends remain similar in shape, but overall ridership decreased significantly from 2019 to 2020, followed by a slight increase in 2021.

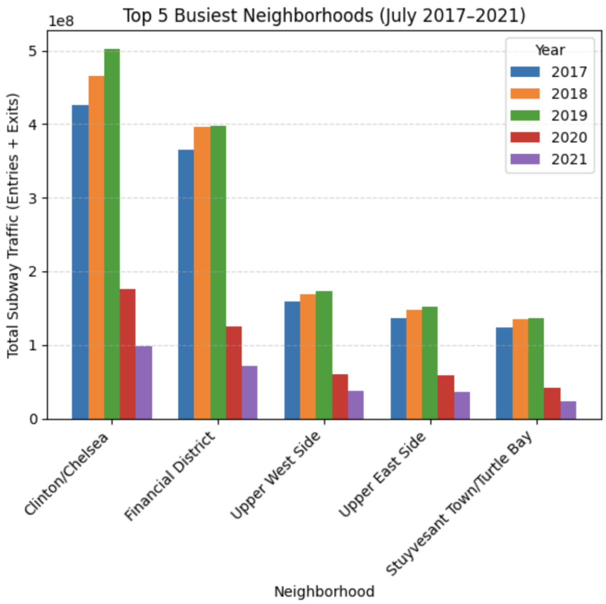

Figure 3.3: Top 5 busiest neighborhoods

Figure 3.4: Ridership count throughout the day

Overall, these exploratory analyses reveal clear spatial and temporal patterns in subway usage that will can inform subsequent modeling and concept drift detection.

In the next chapter, we will select suitable temporal granularity i.e. either daily or monthly for the detection processes.

4 Temporal Granularity Analysis

Before applying drift detection techniques, it is crucial to determine the most suitable temporal resolution to model changes in subway traffic behavior. This chapter explores time-series patterns at both the borough and neighborhood levels and evaluates whether daily or monthly granularity better captures fluctuations that may signal drift.

4.1 Borough-Level Analysis

To evaluate suitable time granularity, we plot traffic data at both daily and monthly levels at Manhattan and Bronx boroughs. Figs. 4.1 and 4.3 present the total traffic ('Entries' + 'Exits') over time, while Figs. 4.2 and 4.4 show the netflow ('Entries' - 'Exits').

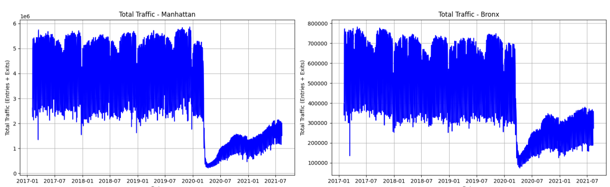

Figure 4.1: Daily Total (Sum) Traffic (Entries + Exits) per Borough

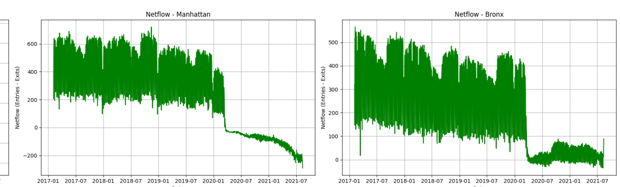

Figure 4.2: Daily Averages of Net Flow (Entries - Exits) per Borough

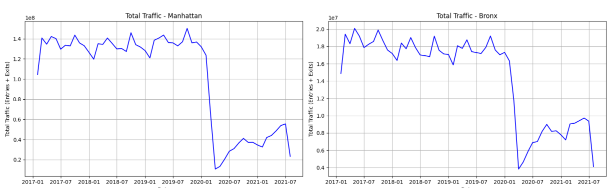

Figure 4.3: Monthly Total (Sum) Traffic (Entries + Exits) per Borough

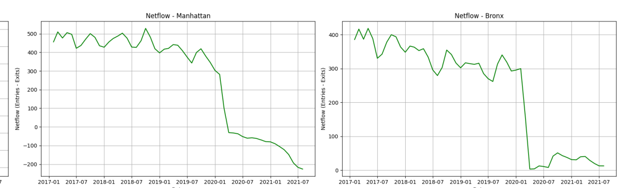

Figure 4.4: Monthly Averages of Net Flow (Entries - Exits) per Borough

When comparing daily and monthly views, we observe that monthly aggregation significantly smooths the signal. Notably, in Figs. 4.3 and 4.4, traffic values remain consistently high before March 2020, masking short-term fluctuations and sudden changes. This smoothing effect may obscure meaningful deviations, especially relevant for concept drift detection.

In contrast, daily-level analysis preserves finer details and reveals abrupt changes that could indicate drift events, such as those caused by the COVID-19 pandemic. Therefore, for all subsequent modeling and drift detection, we choose 'daily temporal resolution'.

4.2 Neighborhood-Level Analysis

A similar comparison is performed at the neighborhood level. Figs. 4.5 and 4.7 illustrate the total traffic trends, while Figs. 4.6 and 4.8 show netflow trends.

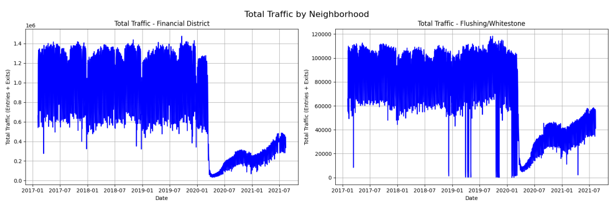

Figure 4.5: Daily Total (Sum) Traffic (Entries + Exits) per Neighborhood

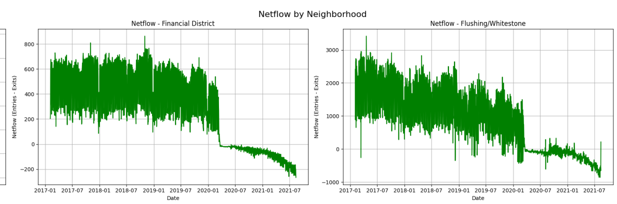

Figure 4.6: Daily Averages of Net Flow (Entries - Exits) per Neighborhood

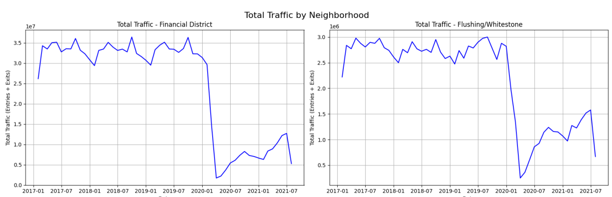

Figure 4.7: Monthly Total (Sum) Traffic (Entries + Exits) per Neighborhood

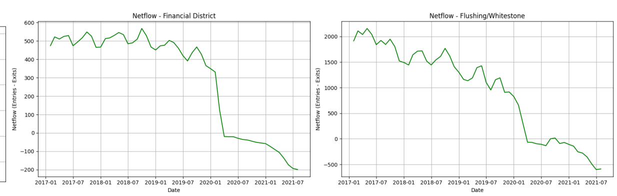

Figure 4.8: Monthly Averages of Net Flow (Entries - Exits) per Neighborhood

Consistent with borough-level observations, daily data at the neighborhood level reveals more granular and variable patterns, whereas monthly aggregation tends to flatten changes and obscure local irregularities.

In a nutshell, daily temporal granularity offers a clearer and more nuanced view of fluctuations in subway traffic compared to monthly aggregation. Therefore, all modeling and drift detection analyses in the subsequent chapters will be conducted using daily-level time-series data to ensure higher sensitivity to behavioral changes.

In the following chapter, the methodologies used to detect concept drift (outlining both offline and online approaches) and evaluating their applicability to human mobility data in the NYC subway system are described.

5 Drift Detection Methods

This chapter covers various detection methods to analyze the concept drift in the fluctuations of human behaviors in the usage of NYC traffic. The detection techniques involves detecting the changes in data points over time, and moreover, help in fitting a model which can predict values in the future.

5.1 Online Detection

This section focuses on the concept of online detection in the context of evolving data streams. Online detection refers to identifying changes in real-time as new data points arrive (streaming data).

In this work, online detection is used to monitor variations in subway traffic patterns in New York City. By tracking features such as ‘Total Traffic’ and ‘Netflow’, the system aims to capture abrupt or gradual changes in human traffic behavior over time, which can be influenced by external events, such as, the COVID-19 pandemic, policy changes, or seasonal shifts.

5.1.1 ADWIN

ADWIN (ADaptive WINdowing) is a popular algorithm designed for online concept drift detection because it operates without the need to store the entire data history which makes it feasible and suitable for real-time applications.

ADWIN maintains a variable-length sliding window over the most recent data points. The central idea is to detect statistically significant changes in the average of data over time. The window is dynamically adjusted over time i.e., it grows when the data appears stable and shrinks rapidly when a change is detected.

In practice, the mechanism works by splitting the window into two sub-windows and comparing their averages using a statistical test. If the difference between the means exceeds a certain confidence bound (adjusted by the `delta` parameter i.e. 0.0001 in our case), ADWIN identifies as a drift.

Parameter	Value
delta	0.0001
clock	4
max _*buckets*	20
min_window_length	3
grace_period	2

Table 5.1: Parameter used for ADWIN

Also, if the drop/spike is very short-lived (probably 1-3 data points) or followed immediately by a rebound, ADWIN tends to miss detection because the sensitivity of ADWIN can be mainly defined with `delta` and `min_window_length` parameters. A lower `delta` makes ADWIN more conservative,meaning that it is hesitant to detect small changes, leading to fewer false positives while `min_window_length` prevents short-term point changes due to not having larger window size for comparison.

Table 5.1 shows the parameters used for ADWIN and more information about each parameter can be found on [5].

Fig. 5.1 and 5.2 show the drift detection by ADWIN over the entire time period of the dataset for every borough. Notably, we can observe that sudden drop at March 2020 is being detected (drawn with red horizontal line). Fig. 5.1 shows that with ADWIN, it has succeeded in detecting the statistical changes, especially after COVID-19 pandemic time (after March 2020). However, repetitive trends before COVID-19 pandemic has not been able to capture. Similar conclusion can also be drawn from Fig. 5.2, but ADWIN appears to perform even worst when ‘Total Traffic’ is used as a feature.

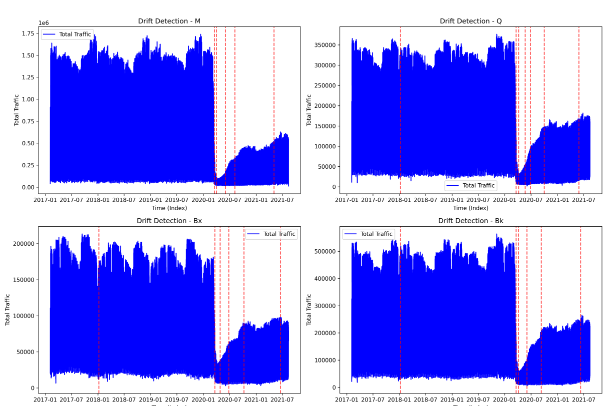

Figure 5.1: Daily Drift Detection for Total Traffic for Every Borough (ADWIN)

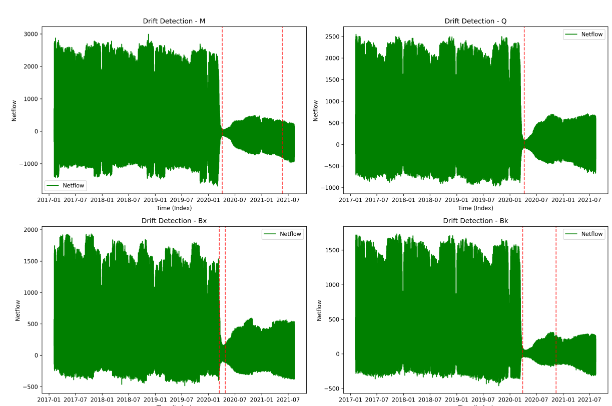

Figure 5.2: Daily Drift Detection for Netflow for Every Borough (ADWIN)

Parameter	Value
alpha	0.01
window_size	4
stat_size	100

Table 5.2: Parameter used for KSWIN

Fig. 5.3 and 5.4 illustrate the daily drift detection for ‘Total Traffic’ and ‘Netflow’ respectively. To avoid the clutter, I have plotted for one busies (‘Financial District’) and one least busies (‘Flushing/Whitestone’) neighborhoods on the left and right hand side of each figure, respectively.

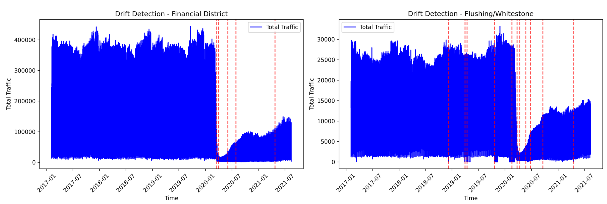

Figure 5.3: Daily Drift Detection for Total Traffic for Neighborhood (ADWIN)

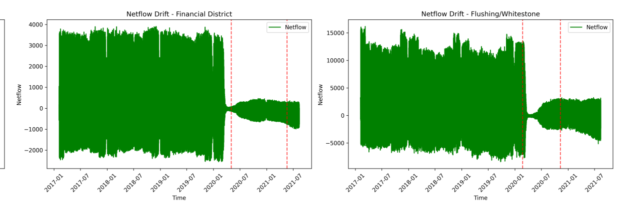

Figure 5.4: Daily Drift Detection for Netflow for Neighborhood (ADWIN)

5.1.2 KSWIN

KSWIN (Kolmogorov-Smirnov Windowing) is another online and unsupervised method for detecting concept drift by comparing data distributions over time. It uses the Kolmogorov-Smirnov (KS) test to compare two sliding windows of recent data i.e. a reference window with past data and a current window with the latest data.

If the KS test detects a significant difference between these two windows’ distributions (based on a set significance level), it flags as a drift. The advantage over the traditional detection method is that it tries to detect changes in the overall data distribution, and not just mean or variance.

Table 5.2 shows the parameters used for KSWIN, and more information about each parameter can be found on [6].

Fig. 5.5 and 5.6 show the drift detection for ‘Total Traffic’ and ‘Netflow’, by KSWIN over for every borough. Notably, we can observe many change points (red horizontal lines), indicating that the model was able to detect even minor change points for both cases.

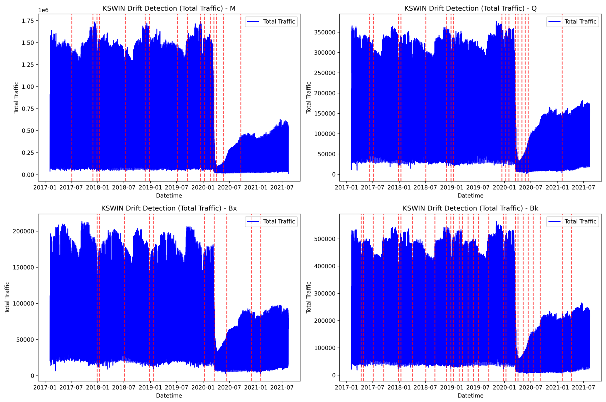

Figure 5.5: Daily Drift Detection for Total Traffic for Every Borough (KSWIN)

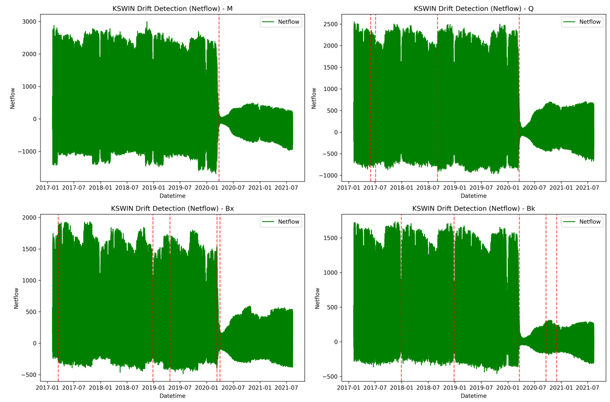

Figure 5.6: Daily Drift Detection for Netflow for Every Borough (KSWIN)

Fig.5.7 and Fig.5.8 illustrate daily drift detection for Total Traffic' and Netflow', respectively, within the Financial District' and Flushing/Whitestone' neighborhoods. Using the same KSWIN parameters, these neighborhood-level plots reveal more change points compared to the borough-level results shown in Fig.5.5 and Fig.5.6. This suggests that the effectiveness and sensitivity of the method depend significantly on the characteristics of the data being analyzed.

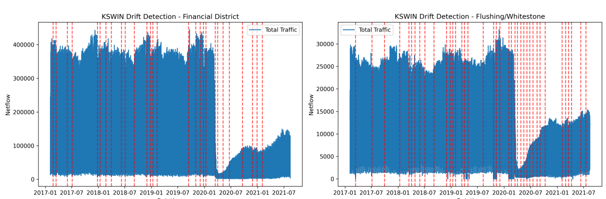

Figure 5.7: Daily Drift Detection for Total Traffic for Neighborhood (KSWIN)

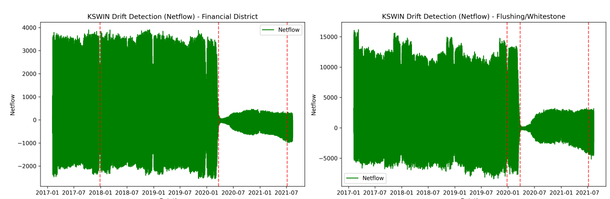

Figure 5.8: Daily Drift Detection for Netflow for Neighborhood (KSWIN)

5.2 Offline Analysis

In this section, we will examine how traditional machine learning models like support vector regressor, linear regressor, prophet and random forest regressor perform on the concept drift data. At a high level, two training strategies are implemented: adaptive full retraining and a fixed-length (7-day) rolling window approach.

5.2.1 Adaptive Full Retraining

Adaptive Full Retraining refers to the process of continuously retraining models using all past data points and testing them on newly arriving data as time progresses. In this approach, I first trained the model using the first week of data, and from that point onward, I began predicting each subsequent day.

Fig. 5.9, 5.10, 5.11 and 5.12 show the prediction of ‘Total Traffic’ of Manhattan using linear regressor (LR), support vector regressor (SVR), prophet and random forest regressor (RFR), by integrating them in an adaptive full retraining manner.

In Fig. 5.9, it is evident that linear regressor model fails to predict ‘Total Traffic’ because model is simple in the sense that it assumes linear relationship between input and target variables. Thus, the model is highly unreliable and unpredictable when there is extreme fluctuations in the data points.

In Fig. 5.10, predicted values are usually high and fail to predict lower actual values due to epsilon-insensitive loss, meaning that if the data point flunctuation is immediate and lower, SVR tends to ignore such small deviation and furthermore, smoothens the predictions.

Comparing to SVR and LR, prophet gives reliable prediction as shown in Fig. 5.11. However, prophet is often late in adapting or predicting the actual values when there is a sudden drop or gradual increase in data points. This is due to prophet waiting for enough data after an actual change to confirm an actual change has occurred. The evidence can be seen from around March 2020 where the predictions begin to diverge significantly from the actual values.

Among models, random forest regressor predicts better as the forest is more denser over time (due to the increase in training size), increasing the capability to capture data trends.

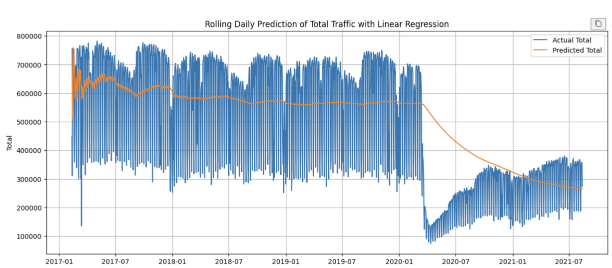

Figure 5.9: Linear Regressor (Adaptive Full Retraining)

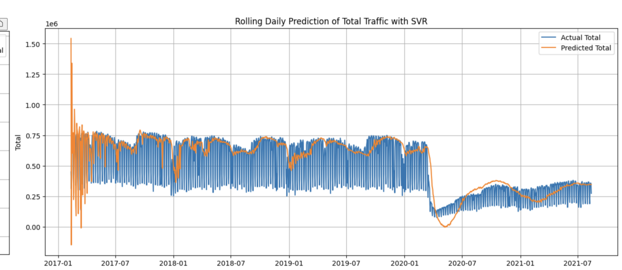

Figure 5.10: Support Vector Regressor (Adaptive Full Retraining)

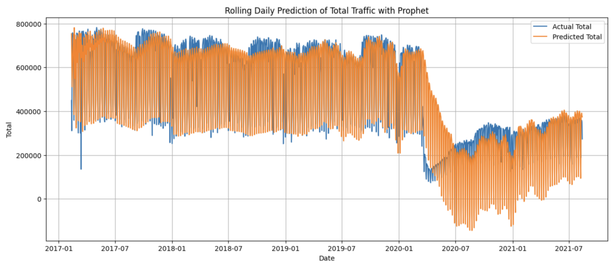

Figure 5.11: Prophet (Adaptive Full Retraining)

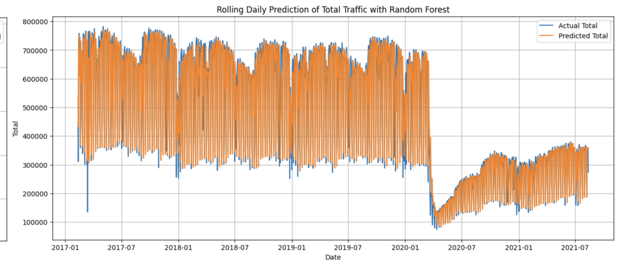

Figure 5.12: Random Forest Regressor (Adaptive Full Retraining)

5.2.2 7-Day Rolling Window

This subsection introduces another offline detection approach known as the 7-day rolling window. In this method, a model is initially trained on the first week's data and used to predict the Total Traffic for the following day. As time progresses, the model is retrained on a new 7-day window each time, continuously shifting forward by one day to predict the next.

Fig. 5.13, 5.14, 5.15 and 5.16 show the prediction with the respective models mentioned in the caption, integrating with the 7-day rolling window techniques.

At a high level, we observe that all models tend to predict fairly well when there is a sudden drop in Total Traffic. This may be because the models are trained on a week's worth of data, allowing them to slightly adapt to recent trends. However, each model fails to predict 'Total Traffic' in different ways, as shown in Fig. 5.13, 5.14, 5.15, and 5.16. This can be attributed to the fact that the models are trained on a small amount of data (only one week's worth), making them too simplistic and unable to capture complex patterns, which can be a sign of underfitting.

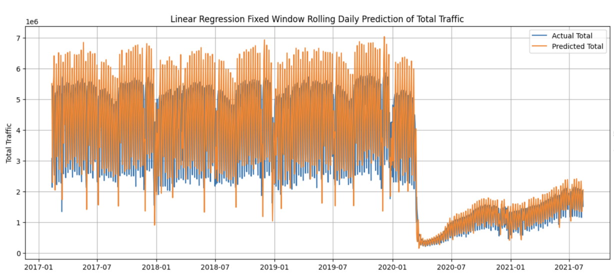

Figure 5.13: Linear Regressor (7-days Rolling Window)

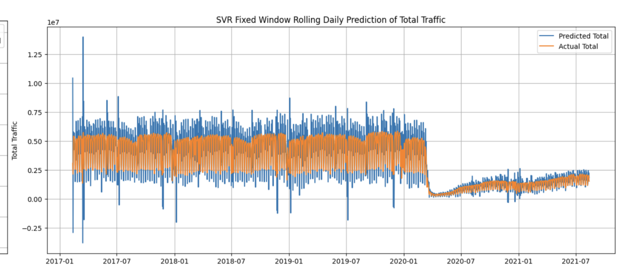

Figure 5.14: Support Vector Regressor (7-days Rolling Window)

Finally, in the next chapter, we conclude our findings and provide a comprehensive analysis

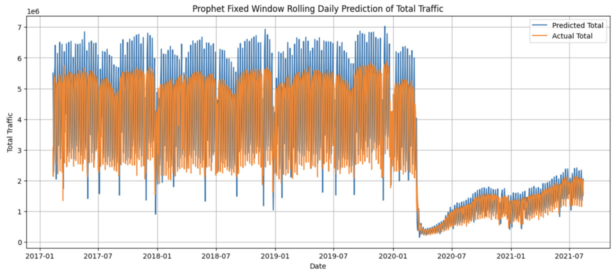

Figure 5.15: Prophet (7-days Rolling Window)

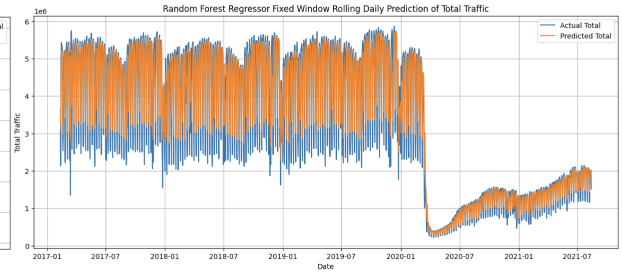

Figure 5.16: Random Forest Regressor (7-days Rolling Window)

of the detection techniques.

6 Conclusions

In this project, we used the ‘NYC Subway Traffic 2017–2021’ dataset to understand human traffic behavior. The process involved a thoughtful approach to selecting useful features, especially important when working with datasets that are preprocessed by third parties (such as those from Kaggle), which often lack scientific validation. Despite this, we were able to gain a clear overview of trends at both the borough and neighborhood levels, which greatly helped guide our decisions on which temporal granularity to focus on for analysis and modeling.

For the detection process, we applied online detection techniques using ADWIN and KSWIN. While we successfully identified several change points, a major challenge lies in hyperparameter tuning and the need for domain knowledge. Hyperparameter tuning requires a good understanding of statistics and probability theory to grasp the underlying mechanisms of these algorithms. Alternatively, having domain knowledge can significantly help in choosing effective parameters, such as tuning them to detect change points around the COVID-19 pandemic era.

On the other hand, we observed important trade-offs in integrating various regression models using adaptive full retraining and a 7-day rolling window. Among them, adaptive full retraining with the random forest regressor performed best in predicting ‘Total Traffic’ values. However, this approach requires training the model at each time step using all past data, making it slow and computationally expensive. While the 7-day rolling window approach addresses these issues, all regressor models trained with just one week of data tend to be too simple and suffer badly to accurately predict future values due to underfitting.

In a nutshell, since predicting human traffic behavior can be done on a daily or monthly basis and does not require real-time analysis, it may be more practical to omit the online detection processes. To better capture data patterns and build more robust models, increasing the rolling window size to two weeks or even using monthly data may help reduce underfitting and improve prediction performance.

Finally, developing predictive models helps us understand traffic dynamics and moreover, provides essential support for urban planning and development, particularly in infrastructure decisions and mobility management strategies [1].

References

[1] M. A. Çolak and O. Ü. Bayrak. "Predictive Modeling of Urban Travel Demand Using Neural Networks and Regression Analysis". In: *Urban Science* 9.6 (2025). ISSN: 2413-8851. DOI: 10.3390/urbansci9060195. URL: https://www.mdpi.com/2413-8851/9/6/195.

[2] A. Kerr. *A Historical Timeline of COVID-19 in New York City*. Accessed: 2025-05-30. 2021. URL: https://www.investopedia.com/historical-timeline-of-covid-19-in-new-york-city-5071986.

[3] A. Li, P. Zhao, H. Haitao, A. Mansourian, and K. W. Axhausen. "How did micro-mobility change in response to COVID-19 pandemic? A case study based on spatial-temporal-semantic analytics". In: *Computers, Environment and Urban Systems* 90 (2021), p. 101703. ISSN: 0198-9715. DOI: https://doi.org/10.1016/j.compenvurbsys.2021.101703. URL: https://www.sciencedirect.com/science/article/pii/S0198971521001101.

[4] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. "Learning under Concept Drift: A Review". In: *IEEE Transactions on Knowledge and Data Engineering* (2018), pp. 1–1. ISSN: 2326-3865. DOI: 10.1109/tkde.2018.2876857. URL: http://dx.doi.org/10.1109/TKDE.2018.2876857.

[5] River Contributors. *ADWIN - Drift Detection*. Accessed: 2025-05-30. n.d. URL: https://riverml.xyz/dev/api/drift/ADWIN/.

[6] River Contributors. *KSWIN - Drift Detection*. Accessed: 2025-05-30. n.d. URL: https://riverml.xyz/dev/api/drift/KSWIN/.

[7] I. Žliobaitė, A. Bifet, B. Pfahringer, and G. Holmes. "Active Learning With Drifting Streaming Data". In: *IEEE Transactions on Neural Networks and Learning Systems* 25.1 (2014), pp. 27–39. DOI: 10.1109/TNNLS.2012.2236570.