STIB-MIVB: Hack My Ride

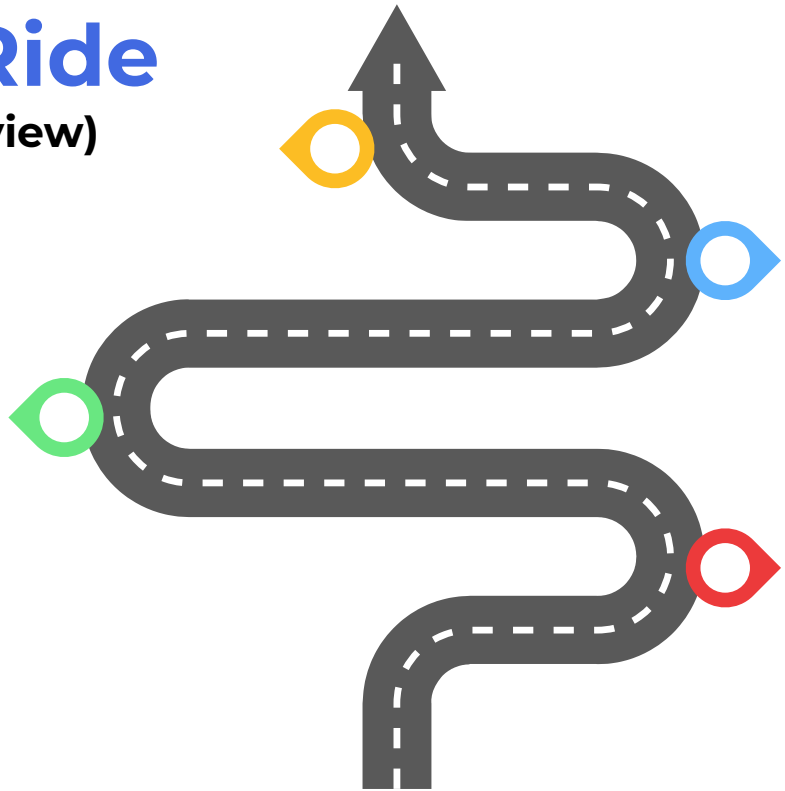
Data Mining & QoS Analysis (Pipeline Overview)

Prepared by:
Ahmad (000559204), Alaa (000559472), Chidiebere (000559941), Zyad (000566436)

Supervised by:
Prof. Dr Mahmoud Sakr

Course:
INFO-H423

[GitHub Repository]



AGENDA

1. **SHAPEFILE**
   Data Preparation & Management
2. **GTFS SCHEDULE**
   Data Preparation & Management
3. **Vehicle Position**
   Data Preparation & Management
4. **Time Groups**
   Clustering & Transformations
5. **Punctuality QoS**
   Methodology & Analysis
6. **Regularity QoS**
   Methodology & Analysis
7. **Bottlenecks**
   Analysis of intersecting lines
8. **Prediction Model**
   Exploring feasibility

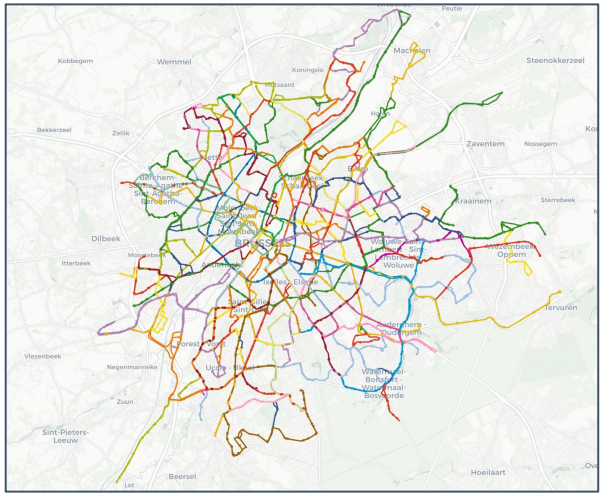
1 **Shapefile:** Data Prep & Management

Preprocessing

- Check for null values.
- Update column names.
- Map vehicle types to BUS, TRAM, METRO.
- Check for distinct lines & stops.
- Ensure that vehicle, line, and direction are in sync between stops and lines shp.

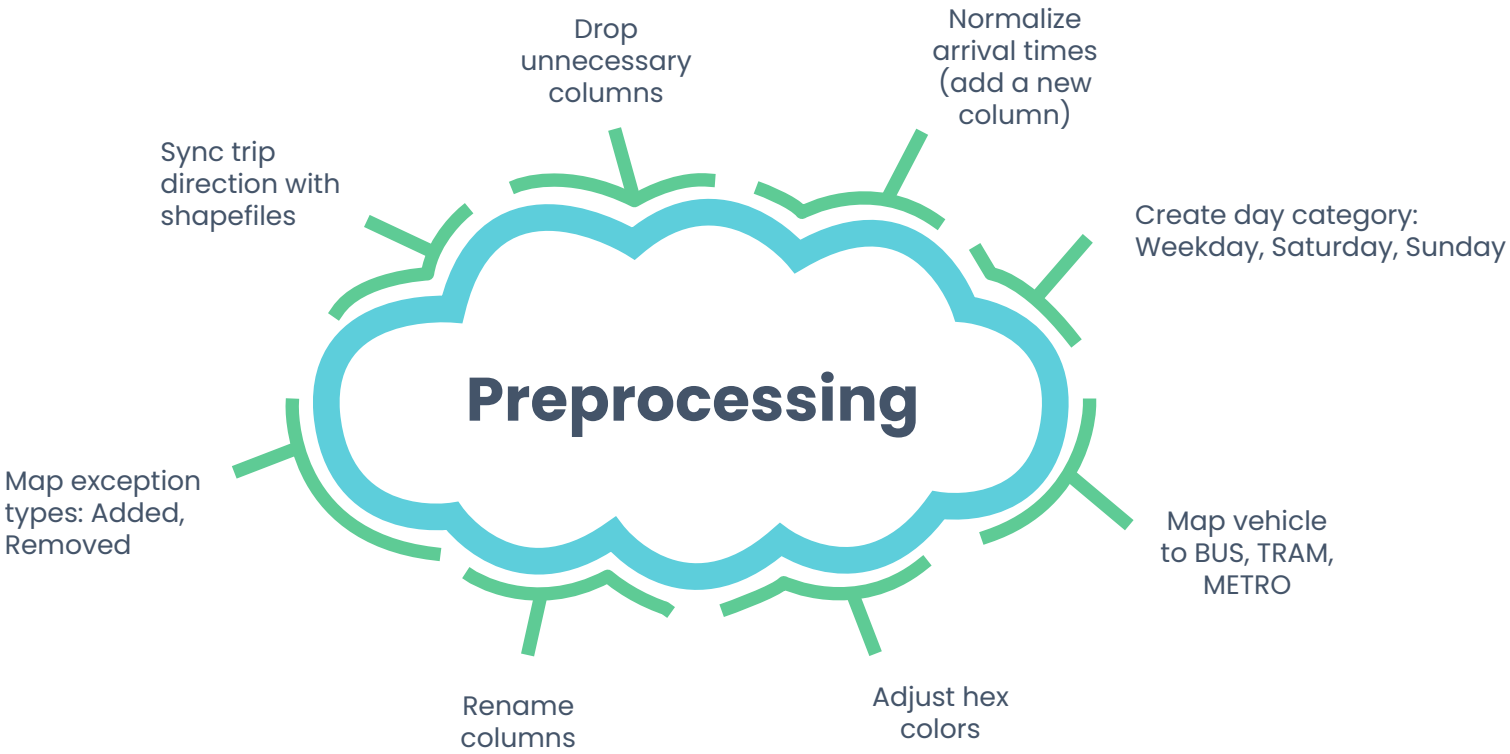
Visualization

- Visualize entire STIB network across Brussels, including all stops and lines.
- Observe individual lines, direction and their respective stops.
- Prepare map setup for usage in the upcoming QoS analysis.



Source Code: stib_network_viz.ipynb

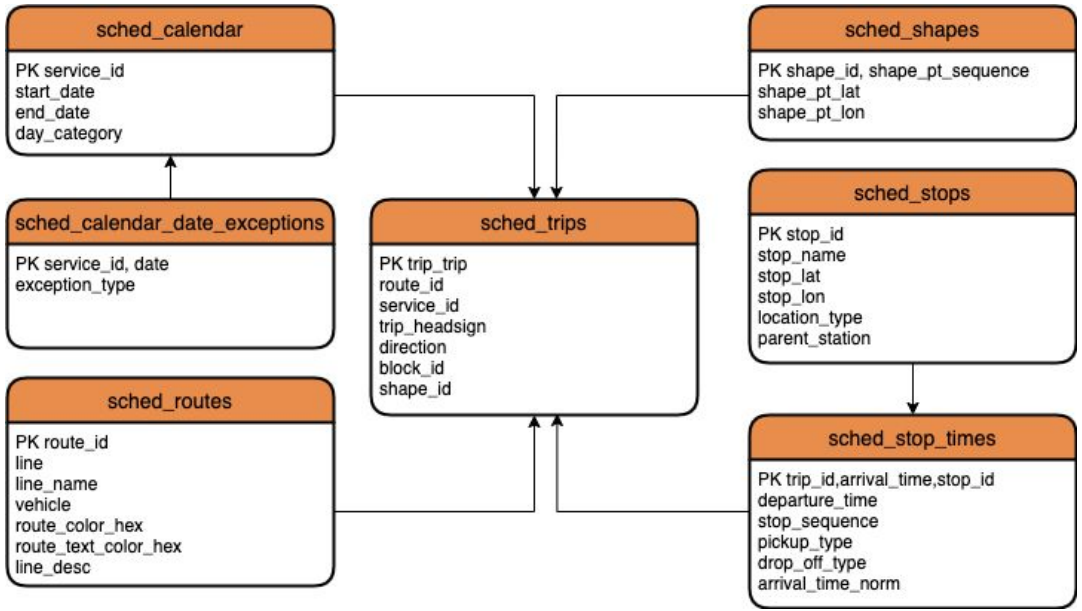
2 GTFS Schedule: Data Prep & Management



Source Code: stib_schedule_preprocess_db_load.ipynb

2 GTFS Schedule: Data Prep & Management

ER Diagram of preprocessed GTFS Schedule data loaded to Postgres



Source Code: stib_schedule_preprocess_db_load.ipynb

3 Vehicle Position: Data Prep & Management

Part 1: Additional Preprocessing of Shapefiles

- Stripping off letters from stop_id
- Remapping name of night bus lines to sync with GTFS
- Deal with a special case where stop_name of last stop and the end_stop_name has a different name

	vehicle	line	direction	stop_seq	stop_id	stop_name	end_stop
0	BUS	12	1	9	1780	TRONE	BRUSSELS CITY
48	BUS	53	1	36	3001	HOP. MILITAIRE	HOPITAL MILITAIRE
54	BUS	57	1	21	3001	HOP. MILITAIRE	HOPITAL MILITAIRE
59	BUS	59	2	29	3125	HOP. ETT.-IXELLES	HOPITAL ETTERBEEK-IXELLES
64	BUS	63	1	22	3401	CIM. DE BRUXELLES	CIMETIERE DE BRUXELLES
81	BUS	74	2	25	2417	UCCLE-STALLE	UCCLE STALLE
108	BUS	N04	1	31	3401	CIM. DE BRUXELLES	CIMETIERE DE BRUXELLES
122	BUS	N12	1	29	2477	STALLE (P)	STALLE P
162	TRAM	62	1	25	6502	CIM. DE JETTE	CIMETIERE DE JETTE
164	TRAM	81	1	35	1042	ICHEC	MONTGOMERY

Source Code: stib_vh_pos_preprocess_db_load.ipynb

3 **Vehicle Position:** Data Prep & Management

Part 2: Combine Preprocessed Shapefile and GTFS together

- Motive: To deal with the case where there are some stop_id that are not present in shapefile
- Solution: Make the best of both worlds by combining both
- Merge using vehicle, line, direction and stop_name
- Ensure only unique cases for each stop_id

Part 3: Preprocess JSON Vehicle Position data

- Flatten JSON data
- Re-align timezone of JSON data with GTFS data
  - UTC+0 —> UTC+02 (Brussels timezone with daylight savings)
- Add day category (‘Weekday’, ‘Saturday’, ‘Sunday’) and have additional columns from timestamp by splitting into date and time as separate features.

Source Code: stib_vh_pos_preprocess_db_load.ipynb

3 Vehicle Position: Data Prep & Management

Part 4: Two Layer Mapping of Vehicle Position with Combined Shapefile + GTFS

- Motive: To identify direction, names of stops (both end and previous) and also stop sequence

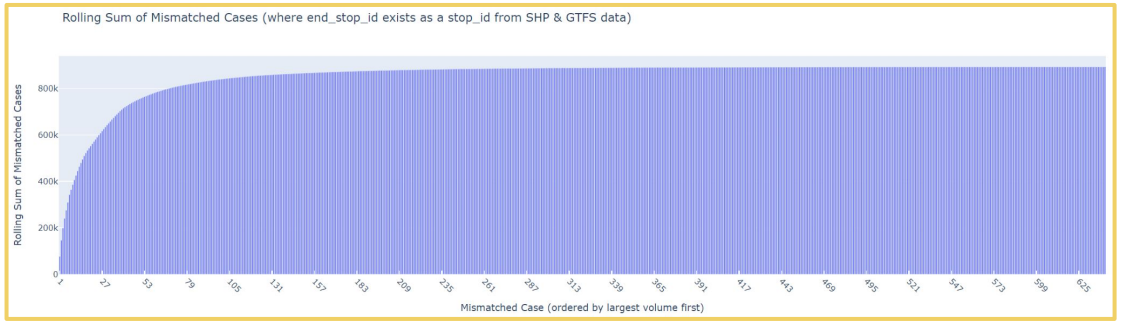
First layer mapping resulted in matching 15mil tuples out of 19 mil

Drop mismatched cases where the stop_id does not exist in either Shapefile or GTFS

Group each case by line with end_stop, and calculate cumulative rolling sum to identify cases to target

Perform second layer mapping to deal with cases where end stop belongs to an intermediate stop

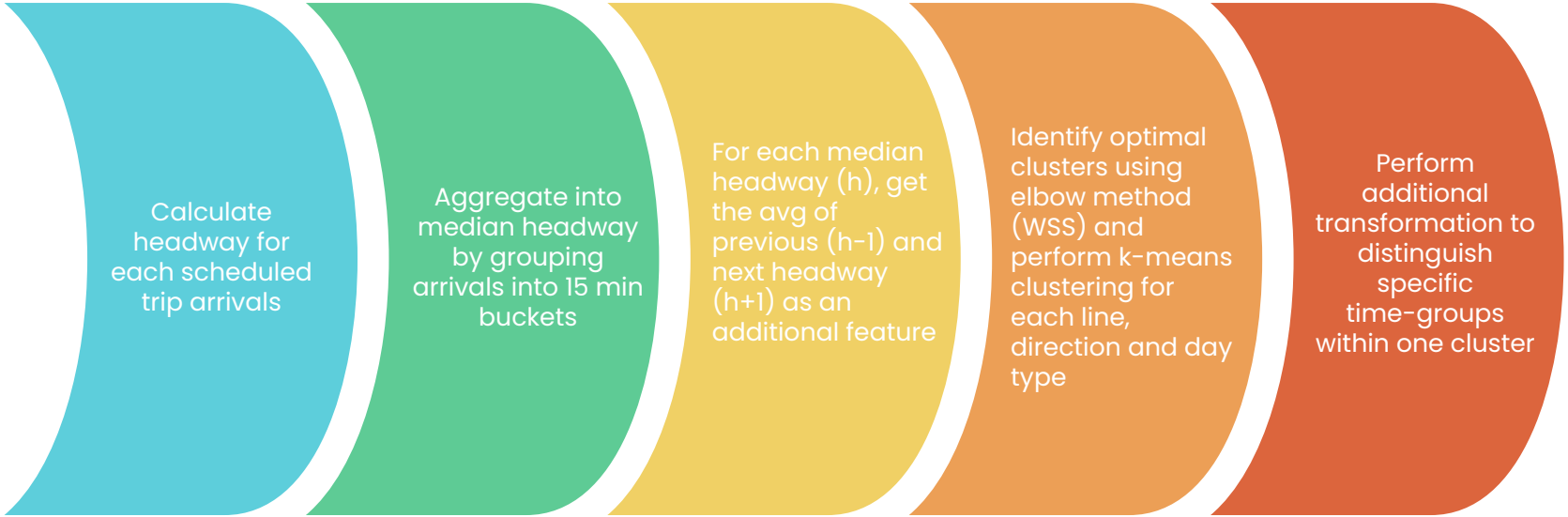
Remaining cases were not possible to be remapped



Final tally of successfully matched tuples: 17 mil

Source Code: stib_vh_pos_preprocess_db_load.ipynb

4 **Time Groups:** Clustering & Transformation



Note: Clustering results and visualization can be found in the dashboard.

Source Code: stib_schedule_eda.ipynb

5 **Punctuality QoS:** Methodology & Analysis

Objective:

- Analyze the delay experienced for each scheduled vehicle per stop, line, direction for the time-groups, lines, stops that belong to the Punctuality QoS.

Method:

- Map actual arrival times for vehicle (json) to scheduled arrival times (gtfs).

How:

- Focus on vehicle positions with "distance from previous stop" as 0.
- Defined plsql function: find_closest_match()*
- Get Delay: ABS(Actual Arrival - Scheduled Arrival).
- Generalize over the lines, vehicles, time of day

Note: Analysis visualizations can be found in the dashboard.

*Challenges are discussed in the next slide.

Source Code: stib_vh_pos_ewt_punctuality.ipynb

5 **Punctuality QoS:** Challenges

Challenges: find_closest_match()

1: Records with start/end_date = 19/09/2021 in calendar in GTFS schedule was mapped to Saturday.

Impact: Inability to map to the scheduled vehicle time.

Solution: Manual update.

2: Records with start_date as 04/09/2021 or 11/09/2021 but end_date = 19/09/2021 was mapped to Saturday

Impact: Inability to map to the scheduled vehicle time.

Solution: Cater for these cases in psql function.

3: GTFS schedule 1 files don't include 20/09/2021 or 21/09/2021 while json files have date up to 21/09/2021.

Impact: Inability to map to the scheduled vehicle time.

Solution: Cater for these cases in psql function.

Source Code: stib_vh_pos_ewt_punctuality.ipynb

6 **Regularity QoS:** Methodology & Analysis

Objective:

- Analyze the Excess Waiting Time (EWT) experienced for each scheduled vehicle per stop, line, direction for the time-groups, lines, stops that belong to the Regularity QoS.

Method:

$$EWT = AWT - SWT$$

$$SWT = \frac{\sum scheduled\_headways^2}{2(\sum scheduled\_headways)}$$

$$AWT = \frac{\sum actual\_headways^2}{2(\sum actual\_headways)}$$

Note: Analysis visualizations can be found in the dashboard.

*Challenges are discussed in the next slide.

Source code: ewt_table_setup.sql

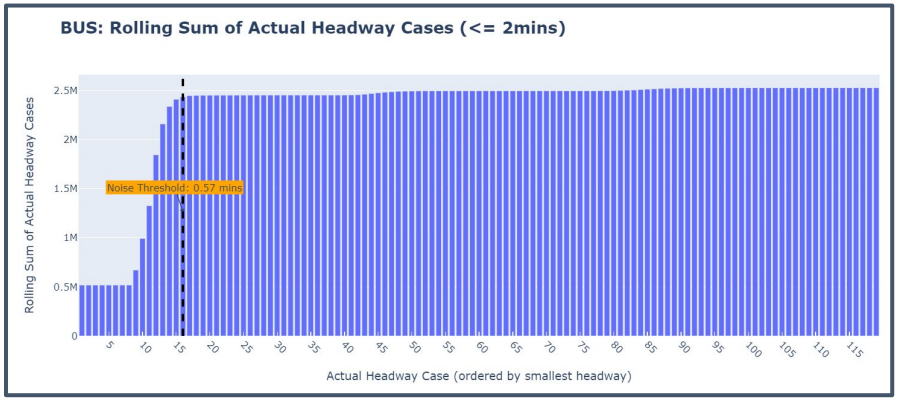
6 Regularity QoS: Challenges (1/2)

Issue:
When identifying cases where vehicle is at a stop (distance from prev_stop = 0), there were cases where two rows has very similar timestamp creating the ambiguity of whether:

- **Hypothesis 1:** It is a case of the same vehicle waiting for extra duration at a stop
- **Hypothesis 2:** An additional vehicle arrived to the stop almost directly after the previous one

Resolution:

- Check whether this is a common occurrence across all lines & stops for each vehicle type
- From the chart (right), it can be seen that up to 0.57mins, there were 2.4 mil cases (knee method) that are very common, confirming that it aligns with hypothesis 1
- These 0.57mins threshold was used to drop rows with these cases
- Actual headways were then recalculated



Source code: stib_vh_pos_ewt_eda.ipynb

6 **Regularity QoS:** Challenges (2/2)

Issue:

- There were cases where some stops for specific line would have all headways with unreasonably large numbers
- This indicates possible missing data of vehicle arrival in between those headways
- As it's quite impossible to constantly have a headway of 30+ mins for a regularity time-group which is scheduled for 8 mins (as an example)

Resolution:

- Given these cases were impractical, and would add unnecessary bias to our analysis
  - The respectives rows were dropped for those specific stops, timegroup etc.
- We would prefer to work with a smaller subset of data that we are confident with, rather than a larger subset of data that has question marks all over it

Source code: stib_vh_pos_eda.ipynb

7 **Bottlenecks:** Analysis of intersecting lines

Objective:

- Further investigate lines that have poor QoS in terms of regularity
- More specifically explore the impact of intersecting lines with a subset sequence of at least 3 stops or more

Method:

- Identify subset of stops with at least an average of 50% or higher cases with 3mins + EWT across a line
- Ensure it has at least a minimum sequence of 3 stops
- Identify additional lines that intersect with this line, to fully understand the impact of the delayed headways that could propagate across the multiple lines
- Visualize the unique cases on a map, with weights on the stops to indicate the level of QoS degradation (displayed in dashboard)

Source code: stib_vh_pos_ewt_regularity.ipynb

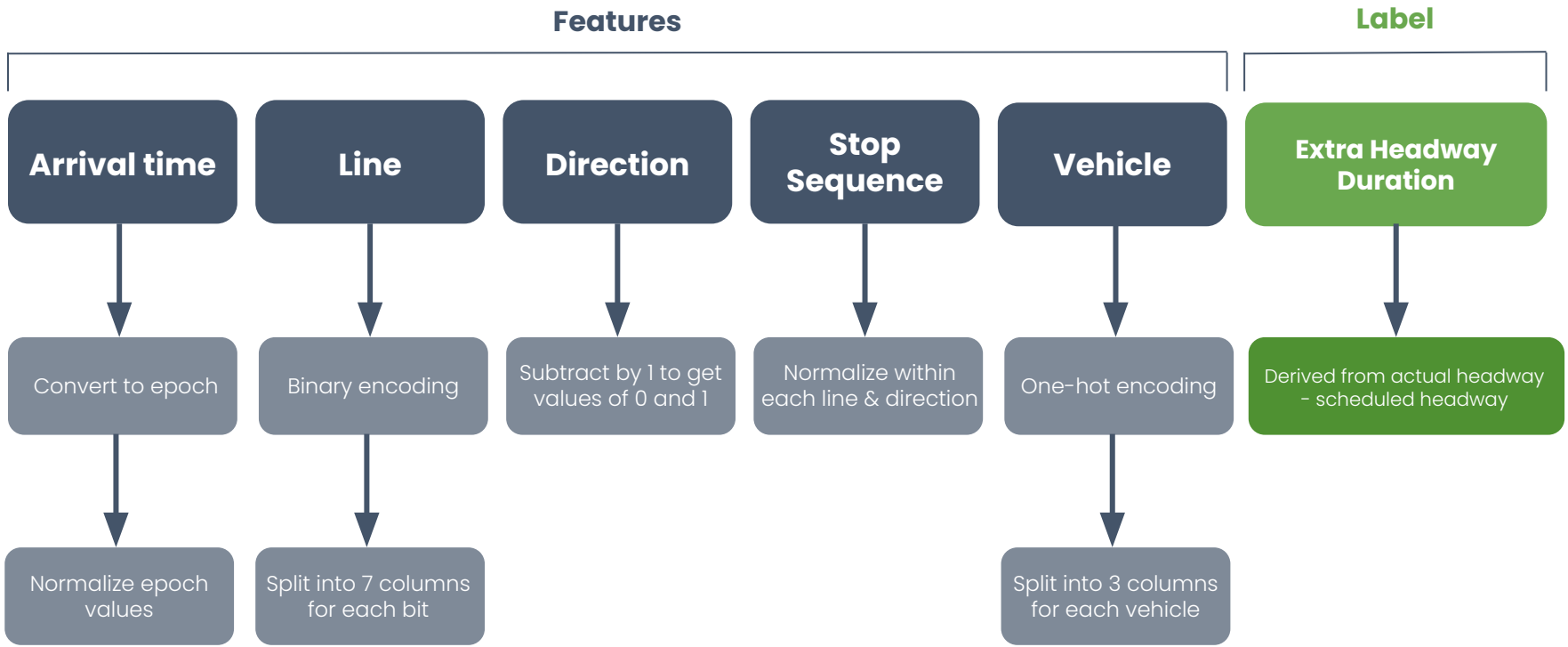
8 **Prediction Model:** Exploring Feasibility

OBJECTIVE

Predicting the extra headway time a customer will wait for a vehicle at a specific station at a specific time of the day.

Source code: stib_vh_pos_extra_headway_duration_prediction.ipynb

8 Prediction Model: Features & Target Label



Source code: stib_vh_pos_extra_headway_duration_prediction.ipynb

8 Prediction Model: Model Evaluation

Dataset:
The total number of data records used is 663,519 records.

Test-Train Split:
80% for training and 20% for testing.

Models Tested:

Model	RMSE
Polynomial Regression	6.67 mins
Random Forest Regressor	7.04 mins
XGBoost Regressor	6.8 mins

Note:
- SVR suffered alot due to high dimensionality
- ARD Regressor did not perform well due to the absence of correlation among features.

Additional Methods:
- Applied hyperparameter tuning using GridSearchCV for Random Forest to see if there were any significant improvements, but there were not
- Implemented PCA to reduce dimensions (14 to 6) with maintaining 80% variance , but it further degraded results

Source code: stib_vh_pos_extra_headway_duration_prediction.ipynb

9 Final Thoughts

Recommendations

- Identify possible ways to shift more of the service to metros, given that vehicles that drive on road, typically have a lower QoS in terms of regularity
    - Begin with city center coverage as that is the hot zone for poor QoS as identified from the heatmap
- Introduce additional vehicles for buses & trams during weekdays and especially peak hours, as that is when the quality of service takes the largest hit
- Address important lines that suffer from regular degradation of QoS and shares a subset of paths with multiple lines
    - Kill two or more birds with one stone ;)

Terminus.