FHV
Vorarlberg University
of Applied Sciences

Classification of GPS Track Data Using Machine Learning Methods

A Case Study of Waste Collection Vehicles

Bachelor thesis
for obtaining the academic degree

Bachelor of Science in Engineering (BSc)

Vorarlberg University of Applied Sciences
Computer Science - Software and Information Engineering

Supervised by
Dipl.-Ing. Dr. techn. Ralph Hoch

Submitted by
Matthias Hefel
Dornbirn, May 2025

Dedication

Dedicated to my younger self, who found a fascination in writing software from a young age and decided to follow his dreams!

And to my parents, who wholeheartedly supported me throughout this journey.

Thank you.

Kurzreferat

Klassifizierung von GPS-Spurdaten mit Machine Learning Methoden am Beispiel von Abfallsammelfahrzeugen

In der Abfallwirtschaft ist die strategische Tourenplanung ein zentraler Prozess, bei dem durch eine effiziente Gebietsaufteilung eine optimale Auslastung der Fahrzeugflotte bei gleichzeitiger Minimierung der Kosten erreicht werden soll. Gerade bei der Planung von Einsätzen in neuen, unbekannten Gebieten sind Entsorgungsunternehmen jedoch häufig mit Unsicherheiten konfrontiert, da Erfahrungswerte fehlen und viele planungsrelevante Annahmen geschätzt werden müssen.

In dieser Arbeit wird untersucht, wie vorhandene GPS-Tracking-Daten von Abfallsammelfahrzeugen genutzt werden können, um die strukturellen Merkmale von Gebieten automatisch zu klassifizieren. Ziel ist es, aus bekannten Gebieten Muster zu extrahieren, die Rückschlüsse auf die räumliche Struktur (z.B. städtisch vs. ländlich) zulassen und somit als Grundlage für Vergleiche mit neuen Gebieten dienen können.

Zu diesem Zweck wurde ein auf maschinellem Lernen basierendes System entwickelt, das GPS-Wegpunktdaten verarbeitet, Merkmale extrahiert, die Daten mit einem semi-supervised clustering Verfahren strukturiert und anschließend ein Klassifikationsmodell trainiert, das neue Datensätze automatisch klassifiziert. Die Klassifizierung von wird in vier Strukturklassen durchgeführt: `URBAN`, `SUBURBAN`, `TOWN` und `RURAL`.

Das Klassifikationsmodell wurde in eine API eingebettet, die eine einfache Anbindung an bestehende Softwarelösungen ermöglicht. Eine automatisierte Klassifizierung kann durch die Angabe einer Tracking-ID durchgeführt werden.

Langfristig bildet diese Arbeit somit die Grundlage für eine ausbaufähige Lösung, bei der auch geografische Strukturdaten (z.B. aus OpenStreetMap) in in die Analyse einbezogen werden, um robuste Vergleichsmöglichkeiten zwischen auf bekannten und neuen Gebieten zu schaffen. Die vollständige Integration dieser zusätzlichen Datenquellen ist jedoch nicht Teil dieser Bachelorarbeit , sondern bleibt zukünftigen Arbeiten vorbehalten.

GPS-Datenklassifizierung, Abfallwirtschaft, Künstliche Intelligenz, Geografische Datenanalyse, Maschinelles Lernen, Automatisierung

Abstract

Classification of GPS Track Data Using Machine Learningn Methods: A Case Study of Waste Collection Vehicles

In waste management, strategic route planning is a central process in which an efficient allocation of areas is intended to achieve optimal utilization of the vehicle fleet while minimizing costs. However, waste management companies are often faced with uncertainties, especially when planning operations in new, unknown areas, as experience is lacking and many planning-relevant assumptions have to be estimated.

This work investigates how existing GPS tracking data from waste collection vehicles can be used to automatically classify the structural characteristics of areas. The aim is to extract patterns from known areas that allow conclusions to be drawn about the spatial structure (e.g. urban vs. rural) and can thus serve as a basis for comparisons with new areas.

To this end, a machine learning-based system was developed that processes GPS waypoint data, extracts features, structures the data using a semi-supervised clustering method and then trains a classification model that automatically classifies new data sets. The classification is carried out in four structure classes: `URBAN`, `SUBURBAN`, `TOWN` and `RURAL`.

The classification model was embedded in an API that enables a simple connection to existing software solutions. Automated classification can be carried out by specifying a tracking ID.

In the long term, this work thus forms the basis for an expandable solution in which geographical structural data (e.g. from OpenStreetMap) is also included in the analysis in order to create robust comparison options between areas known to and new areas. However, the complete integration of these additional data sources is not part of this bachelor project, but is reserved for future work.

GPS Data Classification, Waste Management, Artificial Intelligence, Geographic Data Analysis, Machine Learning, Automation

Contents

List of Figures

List of Tables

List of Abbreviations

AI Artificial Intelligence

API Application Programming Interface

CSV Comma-Separated Values

PKL Python Pickle File

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DACH Germany (D), Austria (A), and Switzerland (CH)

F1-score Harmonic Mean of Precision and Recall

GIS Geographic Information System

GPS Global Positioning System

JSON JavaScript Object Notation

ML Machine Learning

OSM OpenStreetMap

PCA Principal Component Analysis

RF Random Forest

SHAP SHapley Additive exPlanations

URL Uniform Resource Locator

WGS84 World Geodetic System 1984

1. Introduction

"The world's most valuable resource is no longer oil, but data" [1]

In today's digital age, where electronic devices are a part of everyone's daily lives, increasing amounts of data are being generated every day, and this trend shows no signs of slowing down. [2] With this increase in data, businesses ranging across all industries recognize the importance of leveraging it for decision-making and operational efficiency. This has lead to a growing demand for technologies that can gather insights from data and integrate seamlessly into strategic processes.

One industry in which data-driven decision-making is becoming increasingly important is the waste management industry.

1.1. Motivation

> *"The Europe Waste Management Market size was valued at USD 116.21 billion in 2023, and is predicted to reach USD 169.37 billion by 2030, at a CAGR of 4.5% from 2024 to 2030."[3]*

Such growth reflects the increasing need for scalabe, efficient and sustainable waste management practices, in response to rising waste volumes across urban, suburban and rural areas [4].

The waste management sector is considered critical infrastructure and is under growing pressure to adapt to personnel shortages, blackouts, and rising costs. In many cases, the only viable solution is to optimize and automate operational processes. Digital transformation plays a key role in making day-to-day operations significantly more robust and efficient.[5] In this context, the application of machine learning and data analytics promises to enhance strategic planning and reduce operational uncertainty.

This thesis is conducted in collaboration with *infeo GmbH*, a software company specialized in digital transformation in the waste management sector. Their mission emphasizes the need to improve existing infrastructure with innovative tools:

> *"Die Abfallwirtschaft ist ein systemkritischer Wirtschaftsbereich und zählt zur kritischen Infrastruktur. Die Branche muss sich vor Personalausfällen, Blackouts und Kostensteigerungen schützen. Das gelingt oft nur durch Automatisierung und Optimierung von Prozessen, die das Tagesgeschäft maßgeblich robuster und effektiver machen. "[5]*

1.2. Problem Statement

Companies operating in the waste collection business have trouble calculating accurate cost estimates for new service areas when expanding their field of business into areas with no prior operational data. They often have to make assumptions and rough estimates on several paramters concerning the operation cost in new service areas. A data driven estimation can help create more accurate and less risky assessments for unknown collection locations. This can help reduce uncertainties and improve the accuracy of bid calculations. Since GPS tracking data is already collected in existing service areas, there is potential to extract meaningful patterns from this data to support more accurate decision-making in new contexts.

1.3. Solution Approach and Goal

The primary goal of this thesis is to explore how GPS tracking data from waste collection vehicles can be analysed to gain meaningful insight and automatically classified to support better planning decisions.

To achieve this, a machine learning pipeline was developed that involves extracting meaningful features from GPS waypoint data, applying semi-supervised clustering in order to group similar trackings, assigning interpretable labels (eg., `URBAN`, `SUBUBRBAN`, `TOWN` and `RURAL`) based on structural characteristics and training a supervised classification model to automate label prediction for new data. The resulting classifier was built into an API, allowing classification of any given tracking via an API enpoint given its tracking ID.

While the long-term vision proposed by *infeo GmbH* involves a service for full automation of planning parameter suggestions for new geofences, based on OpenStreetMap and internal tracking data, this thesis focuses on building a foundational component: the GPS-based route classification system.

2. Background and Related Work

This chapter provides an overview of the technical and scientific basis relevant for this thesis. Relevant methods include feature extraction to gain meaningful information from GPS data, clustering algorithms for unsupervised learning and related work showing approaches to similar problems. The presented work builds upon already existing methods by combining clustering and classification to identify structural properties of waste collection routes from GPS trackings.

2.1. Technical Background

In order to develop a system that can analyse and classify structural patterns in GPS data, it is necessary to understand how geographic data can be transformed into meaningful features and how to leverage them by applying machine learning techniques. This section outlines the key technical components that support this thesis.

2.1.1. Feature Engineering for Geographic data

Geospatial data, composed of sequential latitude and longitude points, is not inherently suitable for machine learning and requires transformation into numerical features to be suitable for machine learning. Key features commonly used in literature include: speed, acceleration, bearing and distance between consecutive points [13]. These extracted features serve as a representation of the underlying movement behavior of the GPS route. In this thesis, a combination of geometric (e.g., bounding box area, point density) and dynamic (e.g., heading change) features were used to represent each waste collection tracking.

2.1.2. Clustering Algorithms

> "Clustering is a useful tool in data science. It is a method for finding cluster structure in a data set that is characterized by the greatest

similarity within the same cluster and the greatest dissimilarity between different clusters."[6]

2.1.2.1. Density-Based Clustering with DBSCAN

DBSCAN (Density-Based Algorithm for Discovering clusters in Large Spatial Databases with Noise) was introduced by Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiowei Xu as an algorithm to identify clusters of arbitrary shape in spatial databases and distinguish noise. It relies on the notion of density-reachability among points based on a distance threshold *Eps* and a minimum number of points *MinPts*.[7]

Let D be a database of points and let *dist(p, q)* be the distance between points p and q (typically Euclidean).

- The **Eps-neighborhood** of a point p is defined as:

$$N_{Eps}(p) = \{q \in D \mid dist(p,\ q) \leq Eps\}$$

- A point p is a **core point** if:

$$|N_{Eps}(p)| \geq MinPts$$

- A point q is **directly density-reachable** from p if:

$$q \in N_{Eps}(p)\ and\ p\ is\ a\ core\ point$$

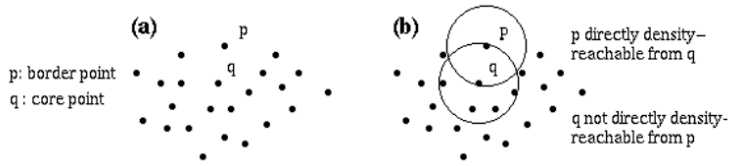


Figure 2.1.: DBSCAN: Core points and border points [7]

- A point q is **density-reachable** from p if there is a chain of points $p_1, p_2, ..., p_n$ such that $p_1 = p$, $p_n = q$, and each p_{i+1} is directly density-reachable from p_i.
- Two points p and q are **density-connected** if there exists a point o such that both p and q are density-reachable from o.

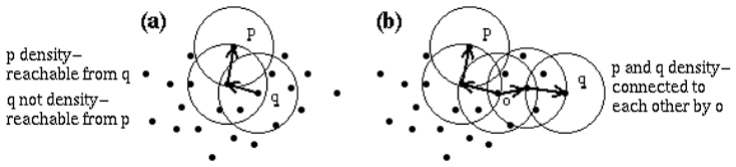


Figure 2.2.: DBSCAN: (a) Density-reachability and (b) density-connectivity [7]

This concept allows DBSCAN to detect clusters without requiring the number of clusters as input, unlike other clustering algorithms such as K-Means and CLARANS, making it effective for outlier detection. [7]

2.1.2.2. K-Means Clustering

K-Means is the first unsupervised algorithm used to cluster data in an euclidean space and is still widely used to partition a dataset $\{x_1, x_2, \ldots, x_n\}$ of size n into $k \leq n$ sets $S = \{S_1, S_2, \ldots, S_k\}$ [6]

The goal to minimize the within-cluster sum of squares (WCSS) can be shown with following function:

$$J = \sum_{i=1}^{k} \sum_{x_j \in S_i} \|x_j - \mu_i\|^2$$

The implementation of the algorithm uses an iterative refinement technique, also known as Lloyd's Algorithm. An initial set of k means is selected, randomly or are more sophisticatedly initialized in adaptations such as K-Means++. Then the algorithm repeats the following two steps until the assignments do not change and K-Means therefore converges:

1. **Assignment:** Assign each data point to the cluster with the least squared Euclidean distance (mean):

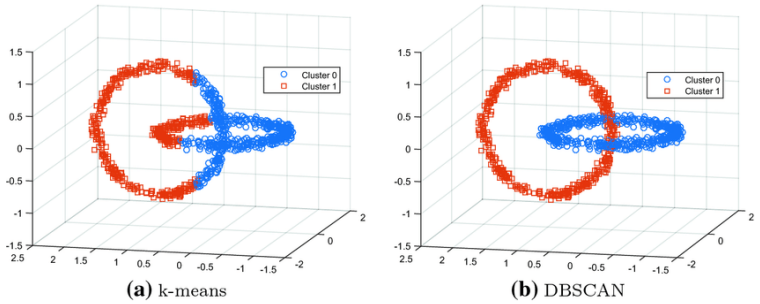
$$S_i = \{x_j : \|x_j - \mu_i\|^2 \leq \|x_j - \mu_l\|^2, \forall\, 1 \leq l \leq k\}$$

2. **Update:** Update the means of data points assigned to each cluster:

$$\mu_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$

2.1.2.3. Combining DBSCAN and K-Means

DBSCAN can be used to identify anomalies by dividing the data set into two clusters. Cluster 0 with the valid data and cluster -1 with anomalies. Using the output of DBSCAN and removing the detected anomaly cluster -1 as the input for K-Means, a more effective partitioning of the data into k number of groups can be achieved. Without DBSCAN, K-Means clustering will likely assign one cluster to outliers of faulty data. In the context of waste collection, where GPS signals may be faulty, this combined method improves the robustness of the clustering by eliminating irregular datapoints, which do not show regular operation of the vehicle.



(a) k-means **(b)** DBSCAN

Figure 2.3.: Comparison: K-Means, DBSCAN clustering algorithms [8]

2.1.3. Machine Learning Model Deployment with APIs

To enable the integration of machine learning models into other applications, model deployment via Application Programming Interfaces (APIs) is a crucial step. APIs allow the model to be exposed and queried by external systems and return predictions based on the user-provided inputs. This thesis uses FastAPI to deploy the trained model and follows modern software engineering principles [9], [10].

2.2. Related Work

While much research concerned with GPS-Data analysis and classification problems focused on transportation mode detection and trajectory mining. Few studies have investigated the classification of urban structure based on waste collection trackings. This thesis addresses this gap by proposing a pipeline to distinguish between urban, suburban, town and rural structures from tracking data alone.

2.2.1. Urban structure analytics

The structure of urban areas has been analysed in many studies. Song et al., proposes a framework that uses access structure to describe and evaluate spatial relationships between streets, plots and buildings [11]. While Zhou et al., proposes a big data approach to analyse the urban spatial structures of different areas [12].

2.2.2. Transportation Mode Prediction with Feature Engineering

Etemad, Soared and Matwin [13] propose a five step framework to predict the underlying transportation mode from GPS traces. The framework includes in order the preparation of data, point feature extraction (e.g., distance, speed, bearing rate and its derivatives), trajectory feature extraction (e.g., min, max, mean and std., percentiles), noise removal and normalization.

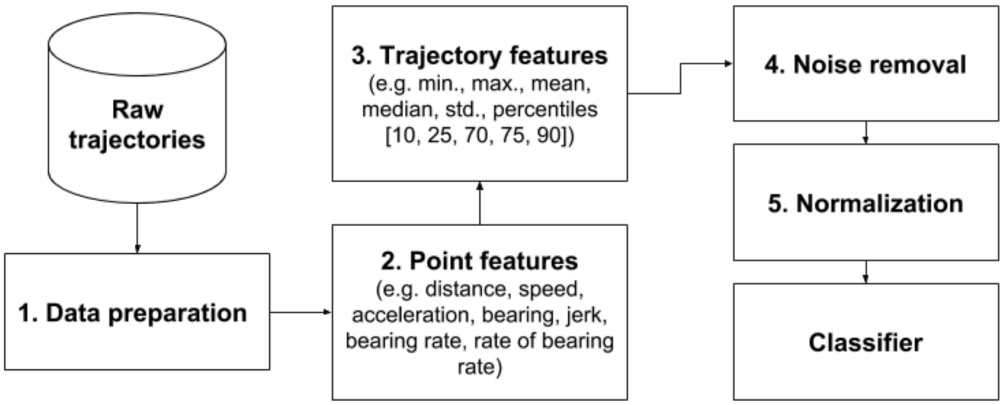


Figure 2.4.: Steps of the framework for predicting of transportation modes proposed by Etemad et al. [13]

Their framework achieves competitive results with the classifier scoring 96.5% accuracy and a f1 score of 96.3%. The study shows the importance of noise reduction and feature engineering when working with GPS trace data. [13]

While their work focuses on the classification of transportation modes (e.g., walking, bike, car, etc.), the techniques used for extracting features and noise reduction, are also applicable for the structural classification of waste collection routes, which is the focus of this thesis.

2.2.3. Clustering of GPS Data using Distance-based Features

Koh et al. [14] present a framework to cluster users based on their movement patterns captured via an application on their mobile phones. The authors propose a new metric, *Daily Characteristic Distance (DCD)*, which is used to create a fair comparison between working and nonworking users on workdays and offdays and extract features in combination with *Origin-Destination (OD)* matrix features.

The features derived from the DCD are combined with OD matrix features and are used in a K-Means clustering algorithm to group the users into three behavior groups. The study analyses the resulting clusters with two newly proposed metrics: *User Commonality* (percentage of users that visited each point of interest (POI) category) and *Average Frequency* (average percentage of trips to each POI category). [14]

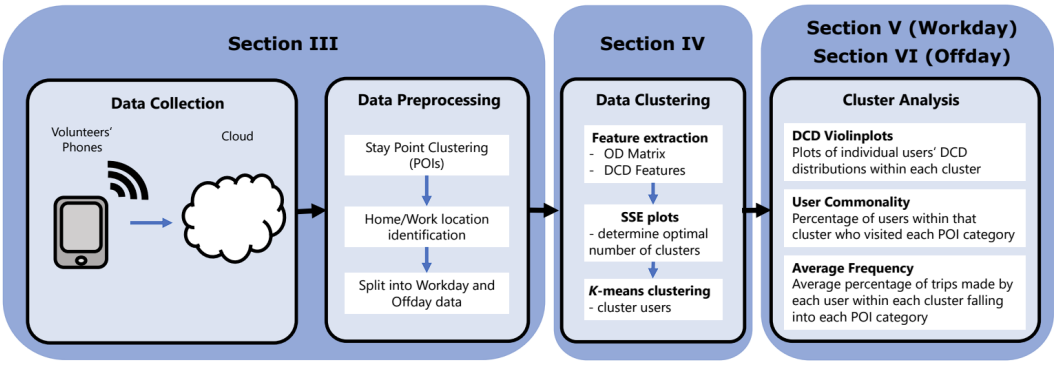


Figure 2.5.: Steps of the framework proposed by Koh et al. [14]

The work of Koh et al. shows that meaningful patterns can be extracted from GPS data using unsupervised clustering methods. This approach is highly relevant for this thesis, as it demonstrates that structural characteristics can be derived from engineered features and effectively used for clustering and classification. Although the papers focus is on GPS traces of individual people and captures data with mobile phones, the underlying methods can be applied for identifying patterns and similarities in GPS track data collected by waste collection vehicles.

3. Problem Definition and Solution Approach

This chapter introduces the core problem addressed by the thesis, describes the dataset used, and outlines the solution approach. The proposed method involves analyzing GPS tracking data from waste collection vehicles, extracting features and using machine learning techniques to classify the structure of waste collection vehicle trackings.

3.1. Big Picture

GPS trackings are collected by waste collection vehicles during real-world operation in the DACH region. These raw trackings are compressed by *infeo GmbH* to reduce storage size while keeping necessary route information, and then saved in a centralized database. The data provided is a subset approved by *infeo GmbH* for analysis in this thesis.

The aim of this thesis is to develop a pipeline that can automatically classify GPS trackings to four categories (e.g., rural, town, suburban and urban).

The proposed solution follows a high-level workflow with the following steps:

1. **Data Preperation:** Data is cleaned and filtered to remove invalid or incomplete trackings.

2. **Feature Extraction:** Meaningful features such as point density and bounding box area are calculated for each tracking.

3. **Outlier Detection:** Secondary detection of invalid trackings and trackings that significantly deviate from expected patterns are removed using density-based clustering (DBSCAN).

4. **Clustering:** The cleaned tracking data is grouped into four clusters based on similarity using K-Means.

5. **Classification:** A Classifier is trained to generalize the clustering results and allow automated classification of new trackings.

6. **API:** The classifier is integrated into an API to allow the classification of trackings via REST calls.

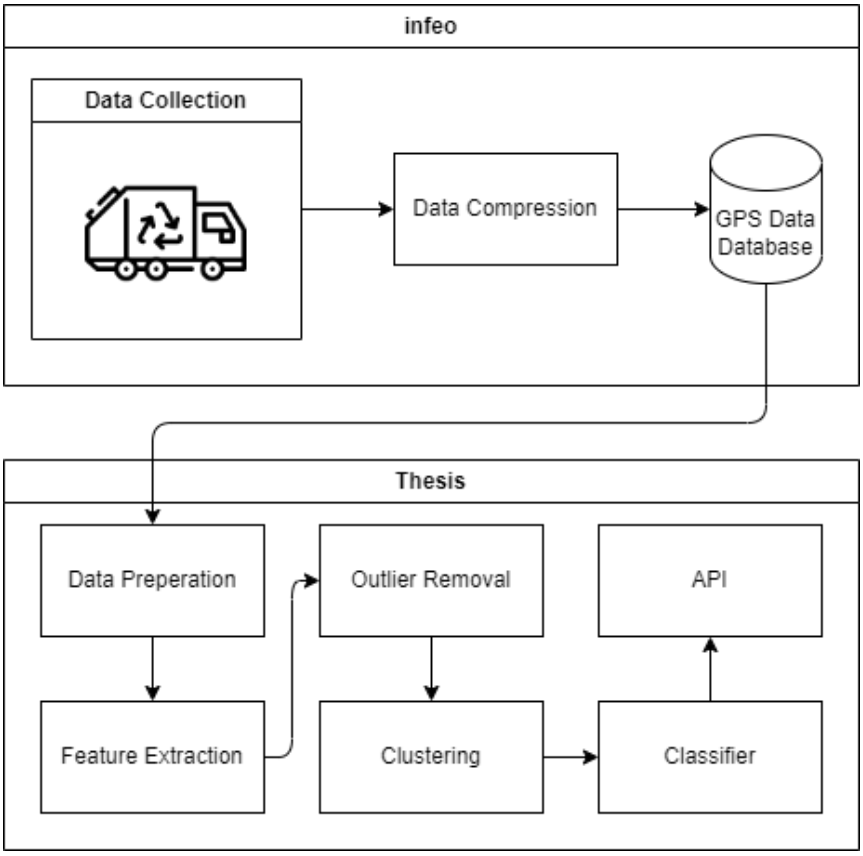


Figure 3.1.: High-level overview of training the GPS track classifier

3.2. Description of the Dataset

Gaining an understanding of the provided data structure is crucial for identifying the available information and its limitations. This analysis helps identify which types of information can be extracted from the dataset and enables the derivation of meaningful features.

3.2.1. Overview

The dataset used is a collection of GPS tracking data collected by waste collection vehicles from various waste collection businesses and provided by *infeo*

GmbH. It represents real-world data collected during regular waste collection operation in the DACH region.

3.2.2. Source and Collection Method

The data was obtained by the onboard tracking systems installed by *infeo GmbH*, which collects GPS coordinates in regular intervals during regular operation. Each tracking represents a complete waste collection route taken and includes metadata as well as a list of GPS coordinates.

3.2.3. Structure of the Data

Each dataset entry represents a single recorded route referred to as *tracking* and contains metadata as well as a ordered list of GPS coordinates.

Each tracking contains the following relevant fields:

Field	**Type**	**Description**
`id`	Integer	Unique identifier of the tracking entry.
`name`	String	Name of the tracking (randomized for anonymization) identification.
`description`	String	Route metadata, often includes internal codes.
`recorded`	DateTime	Start date and time of the tracking.
`length`	Float	Total length of the route in kilometers.
`duration`	Integer	Total duration of the tracking.
`vehicleId`	Integer / Null	ID of the vehicle (nullified for anonymization).
`tourId`	Integer / Null	ID of the associated tour (nullified for anonymization).
`isExported`	Boolean	Flag indicating if the tracking was exported.
`editState`	Integer	Edit state used by the system.

Table 3.1.: Structure of a Tracking Entry

Each GPS point contains the following relevant fields:

Field	Type	Description
`id`	Integer	Unique identifier of the GPS point.
`time`	DateTime	Timestamp of when the point was recorded.
`latitude`	Float	Latitude coordinate.
`longitude`	Float	Longitude coordinate.
`speed`	Float	Instantaneous speed at the time.
`heading`	Float	Direction of movement in degrees.
`sequence`	Integer	Position of the point in the tracking sequence.
`metaTag`	Integer	Custom metadata tag.
`metaValue`	String	Value associated with the metadata tag.
`pointBaseType`	Integer	Internal point type used by the system.

Table 3.2.: Structure of a GPS Waypoint Entry

3.2.4. Size and Coverage

The dataset consists of 101,353 individual trackings with a combined number of 123,785,460 waypoints. Trackings were recorded by waste collection vehicles during real-world operation in the DACH region.

3.2.5. Data compression

The dataset used in this thesis is pre-compressed internaly by *infeo GmbH* while keeping essential route information [15]:

1. **Delete same waypoints:** Consecutive waypoints with identical GPS position are deleted.

2. **Delete waypoints with distance smaller than (x):** Delete all Consecutive waypoints with a distance to each other of less than (x) meters.

3. **Delete waypoints with path curvature smaller than $(x)^s$:** For every group of three consecutive waypoints A, B, and C, the middle point B is deleted if all the following criteria are met:
   a) The distance $\overline{AB}$ is less than a specified threshold (e.g., 100 meters), **or** $\overline{BC}$ is less than the threshold.
   b) The distance $\overline{AC}$ is less than a specified threshold (e.g., 1000 meters).
   c) The change in heading α between segments $\overline{AB}$ and $\overline{BC}$ is smaller than a defined angle (e.g., 10°).

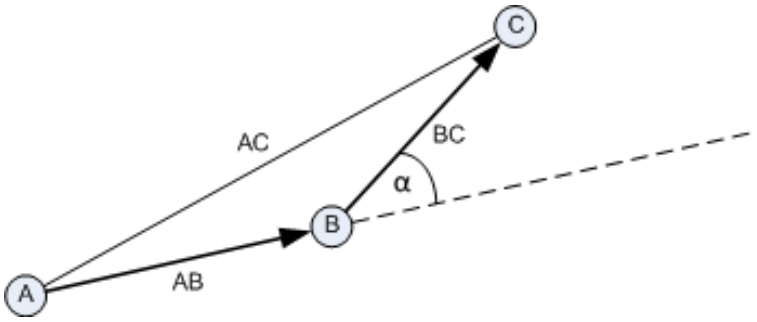


Figure 3.2.: Waypoint curvature calculation used for compression by *infeo GmbH*

[15]

3.2.6. Limitations

A significant limitation of this study is due to the nature of the dataset. While the dataset is extensive, offering a large sum of trackings, many prove to be irrelevant or misleading for analysis. According to discussion with *infeo GmBH* and manual visual inspection, the primary cause of these data issues is the manual operation of the tracking device in the waste collection vehicles.

The most common issues were:

- **Depot-only trackings:** Many trackings were exclusively recording the movement in the vehicle parking facility, capturing no real waste collection operation.
- **Non-operational trackings:** Tracking devices were left active for long periods of time, outside of operational hours, such as overnight, resulting in long trackings with little movement.
- **Large Signal gaps:** Some trackings suffered from significant gaps in the GPS track, resulting from signal loss of the tracker or possibly from the compression step introduced by *infeo GmbH*.

As a consequence of these issues in the dataset, a comprehensive preprocessing and filtering was necessary to ensure the quality of the dataset used for analysis and classification. While the cleaning process was designed to be as robust as possible, it is possible that some faulty trackings persist in the filtered dataset and that some valid trackings were wrongfully discarded.

3.3. Dataset Analysis

Before a classification model can be developed, it is important to understand the underlying data it will be trained on. This section presents an exploratory

data analysis of the cleaned dataset with the goal of identifying relationsships of extracted features that help separate the four categories.

3.3.1. Sample Analysis

A small, manually selected sample of 8 tracking routes was selected for initial exploratory data analysis. Each route was inspected on the *AWM-Map-Tool* and then categorized into one of the four area type labels: `URBAN`, `SUBURBAN`, `TOWN` or `RURAL`. Each Label is represented by 2 tracking routes in the sample data to ensure a balanced representation.

Feature extraction was performed to calculate route-level metrics such as bounding box area, point density and average distance between points.

The goal of this sample is to explore patterns, validate assumptions, and identify features useful for future automatic classification.

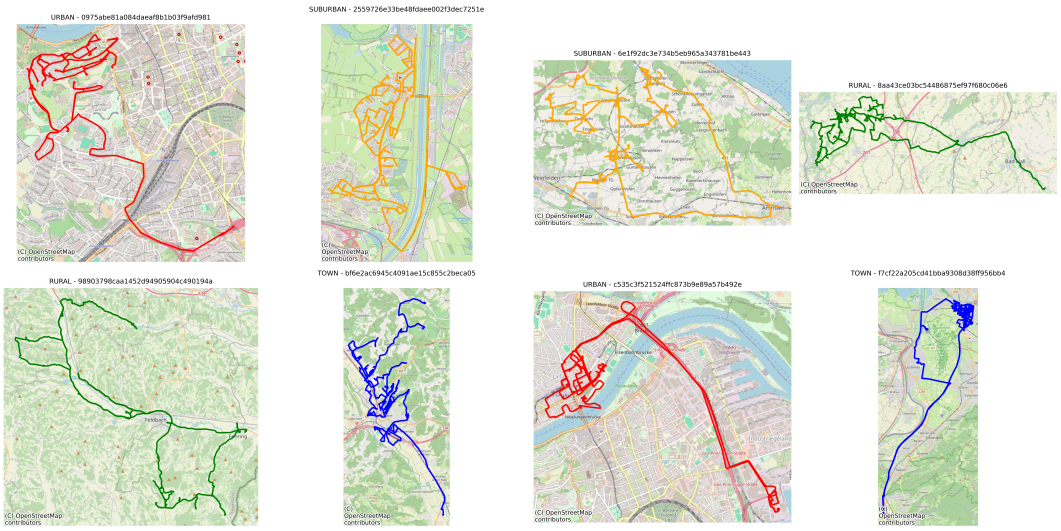


Figure 3.3.: Mapgrid for selected sample trackings

The mapgrid (Figure 3.3) shows spatial layout of each tracking route overlayed on a visual map. Each subplot corresponds to a single tracking and is colorcoded with the assigned label (RURAL=Red, SUBURBAN=Orange, TOWN=Blue, URBAN=Green). This visual representation clearly shows the difference in route shapes and sizes between the four different area types, which are arranged from highest to lowest urban density.

Urban:

- Routes are geographically compact and highly localized.

- Movement appears dense with short travel distances between stops.
- Often confined to a small cluster of city blocks or neighborhoods.

Suburban:

- Routes are more spread out than urban routes, but still relatively dense.
- Represent transition areas between dense city cores and surrounding areas.
- Moderately spaced routes, covering residential areas with more open space and lower building density.

Town:

- Routes are fairly spread out but less spread out than rural areas.
- More dispered than suburban areas, with less consistent street layout.
- Collection points are more spaced out, and routes often span small settlements with connecting streets.

Rural:

- Routes are long and span large geographical areas.
- Stops are widely spaced, often connecting small, isolated settlements.
- The shape and path vary significantly, often following main roads between distant collection points.

Figure 3.4.: Pairplot of selected GPS route features grouped by area label

The pairplot (Figure 3.4) visualizes the relationships between all pairs of extracted features. While the small sample size limits the generalization, the plot suggests clear seperation between the extremese `URBAN` and `RURAL`, especially in dimensions such as point density and bounding box area.

The main utility of this plot lies in confirming basic trends: urban routes appear denser and more compact, while rural routes cover larger spatial areas with lower density. Suburban and town routes fall in between but exhibit more overlap and variability.

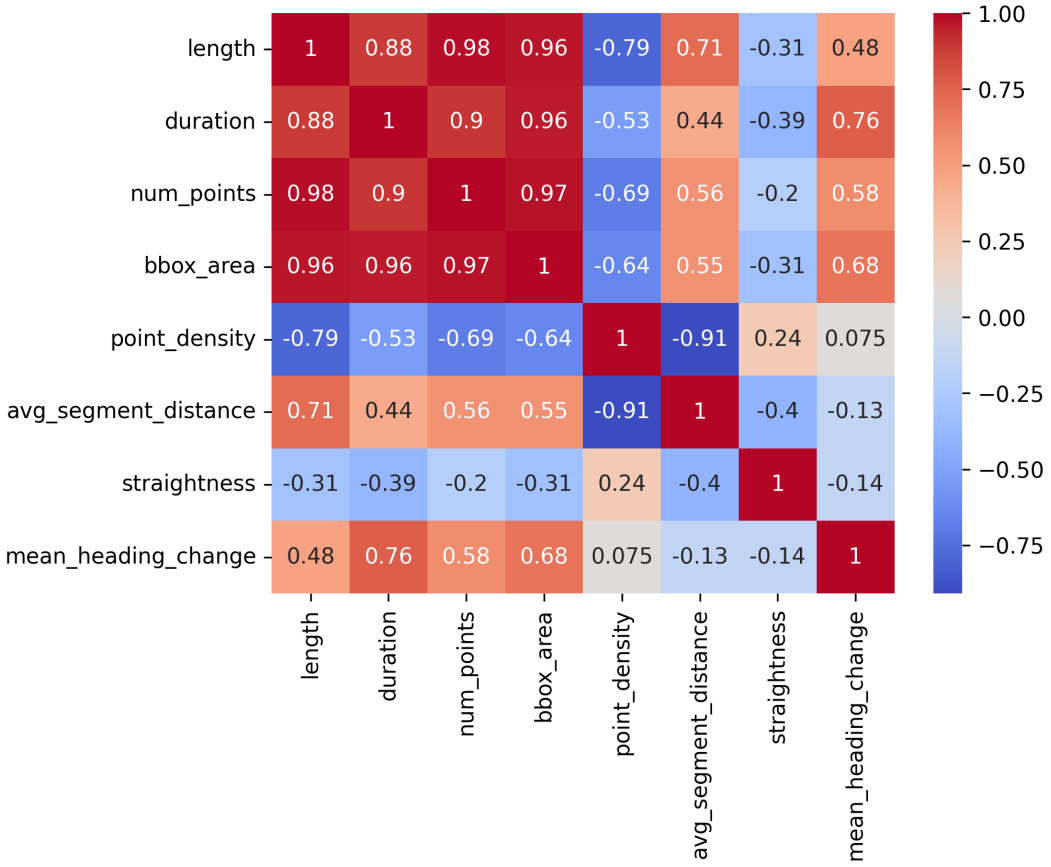


Figure 3.5.: Correlation Matrix of selected GPS route features grouped by area label

The correlation matrix (Figure 3.5) highlights expected and useful relationship between features. Strong positive correlation between `length`, `duration`, `num_points` and `bbox_area` exists, as these features all describe the scale of the tracking. `point_density` shows a strong negative correlation with those same features, indicating that shorter, more compact routes have denser GPS

point coverage. `straightness` and `mean_heading_change` show a moderate correlation to other features, suggesting that they add additional orthogonal information about the geometric characteristics of the tracking.

These relationships validate the consistency of the data and show the value of using this sample as a baseline weighting for future stages in the pipeline.

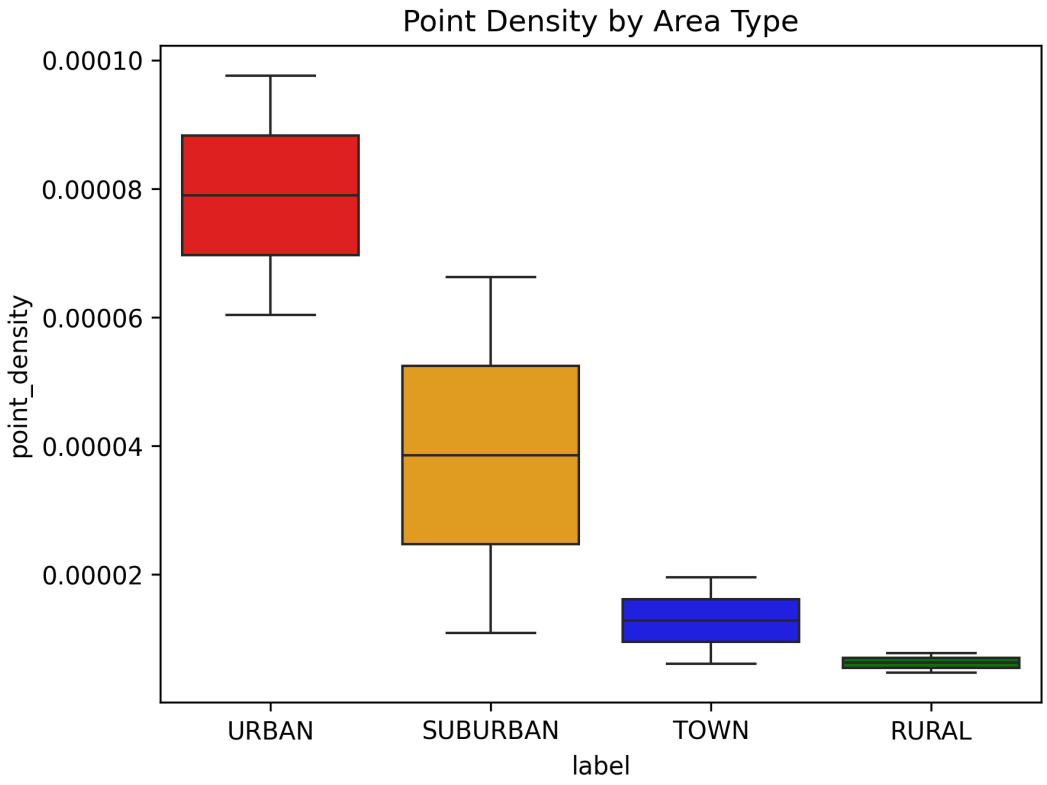


Figure 3.6.: Boxplot of point density grouped by area label

The point density boxplots for each label shows a clear difference between the labels. Urban trackings appear to have the highest density and rural trackings the lowest, with suburban and town trackings falling in the middle with a wider variability.

This strong monotonic pattern across labels supports that point density is a valuable feature for clustering and classification. Other features, such as bounding box area and average segment distance show similar characteristics as shown in Chapter 5.

3.4. Solution Approach

This chapter outlines the methods used to analyse and classify the GPS tracking data collected by waste collection vehicles. The approach is structured in multiple stages, beginning with exploratory data analysis, proceeding with feature extraction, data cleaning, outlier detection, clustering and finally training a classifier. The goal is to identify tracking patterns that allow for structural classification of trackings into four categories: Urban, Suburban, Town and Rural.

3.4.1. Exploratory Data Analysis

The foundation of this thesis is the dataset provided by *infeo GmbH*. It contains anonymized GPS track data collected by waste collection vehicles during real world operation. Due to its large size of more than 100,000 GPS tracks and its anonymization, a clear picture of the data needed to be established as the first step. With visual examination of different random trackings a sample of eight manually labeled trackings was gathered. The selected trackings were labeled to four categories, two trackings per category.

- **Urban:** Dense building structures, complex road networks, and minimal spacing between stops.
- **Suburban:** Moderately spaced residential areas with organized street layouts.
- **Town:** Smaller clustered settlements with simpler road structures.
- **Rural:** Sparse housing, long roads, large distances between collection points.

With the manually selected trackings varying in geographical structure, a exploratory data analysis helped transfer the visual distinction to clear differences found in the data of each tracking category.

3.4.2. Feature Extraction from Sample Dataset

To enable the use of machine learning algorithms, each tracking originally represented by a consecutive list of GPS waypoints with latitude and longitude must be transformed into feature vectors. Feature extraction methods were applied to derive representative metrics for each tracking.

This approach is well established and used in similar studies such as the classification of transportation modes by Etemad et al. [13] and clustering of GPS data using distance-based features by Koh et al. [14]

3.4.2.1. Features Available in Trackings

1. **Length:** Total distance of the tracking, in kilometers as recorded in the database.
2. **Duration:** Total time duration for the tracking, captured and recorded in the databse.

3.4.2.2. Features Extracted from Waypoints

1. **Number of Points:** The total number of GPS waypoints recorded in the tracking.
2. **Bounding Box Area:** The area of the minimum bounding box that encloses all latitude and longitude waypoints in one tracking.
3. **Point density:** The number of waypoints divided by the bounding box area. This feature indicates how closely packed the points are.
4. **Average Segment Distance:** The mean distance between consecutive waypoints, computed using geodisc distance.
5. **Straightness:** The ratio between the geodesic distance from the first to the last point and the total distance of the route. Values close to 1 indicate a straight path, while lower values indicate loops and detours.
6. **Mean Heading Change:** The average change in bearing between consecutive segments of the path, in radians. It reflects how often and how severe the vehicle changes direction.

3.4.3. Filtering and Preprocessing the Full Dataset

There are a lot of faulty datapoints that are not relevant for the clustering, that are filtered out. Before applying machine learning methods, extensive filtering was required to remove invalid and irrelevant data from the dataset. These included:

1. Trackings with a too low duration to be realistic collection trackings.
2. Trackings that recorded movement only in and around a parking lot.
3. Trackings left running overnight with no significant movement.
4. Trackings with unrealistic GPS jumps and anomalies (possibly due to device errors or compression errors introduced by *infeo GmbH*).

Removing these faulty datapoints was curcial for ensuring the quality of the clustering and reducing the computation times.

3.4.4. Clustering and Pattern Discovery

To discover patterns in the unlabeled dataset, clustering algorithms were applied. Initially, K-Means++ was used to cluster the data into four clusters. However the result showed that one cluster mostly consisted of outlier routes, with unrealistic GPS jumps.

To solve this issue, the process was improved by detecting and removing outliers using DBSCAN (Density-Based Spatial Clustering for Applications with Noise). DBSCAN effectively removed most anomalies in the dataset. After filtering the outliers with DBSCAN, K-Means clustering was applied again, resulting clusters that represented the four defined categories better.

3.4.5. Training a Classification Model

To automatically classify new trackings into one of the four categories (Urban, Suburban, Town, Rural), a supervised machine learning model was trained using the labeled dataset from the clustering step. The labeled dataset was split into training and test sets using an 80/20 split. Evaluation of the classifier on the test set showed a high overall accuracy. The trained classifier was saved for the later use in a sandbox API to classify new trackings.

3.4.6. API Integration

To make the trained classification model accessible and usable in a practical sense, the solution includes the integration of a lightweight REST API. This enables external systems to send requests to the API endpoint, and receive a classification result in real-time. The API is designed to receive a `tracking_id` and fetch the tracking data directly from the *infeo GmbH* platform, preprocess it using the same logic as the training pipeline, and apply the Random Forest classifier to return the predicted label.

3.5. Structure of the Work

The remainder of this thesis is structured in the following chapters:

- **Chapter 4: Implementation:** Provides a detailed description of how the previously introduced methos and algorithms were implemented. This

includes the data preprocessing, feature extraction, clustering pipeline and classifier training.

- **Chapter 5: Evaluation and Discussion:** Presents the results of the clustering and classification. Performance metrics, visualizations, and analysis of the results are discussed alongside future reasearch and application.
- **Chapter 6: Conclusion:** Summarization of the contributions of the thesis, highlighting key takeaways and shortcomings.

This structure follows the development of the project, from understanding the problem and analysing the data, to buildinng and evaluating a solution to address the objective of classifying GPS trackings into structural categories.

4. Implementation

This Chapter translates the conceptual pipeline from Chapter 3 into its concrete Python-based implementation. Each stage of the pipeline, from data processing to feature extraction, clustering and classification, was implemented in modular Jupyter Notebooks using libraries such as `pandas`, `scikit-learn`, `geopy` and `pyarrow`.Each module was designed to run independently, allowing for quick experimentation and reproducability of results.

The Figure 4.1 shows a more detailed view of the pipeline introduced int Chapter 3.

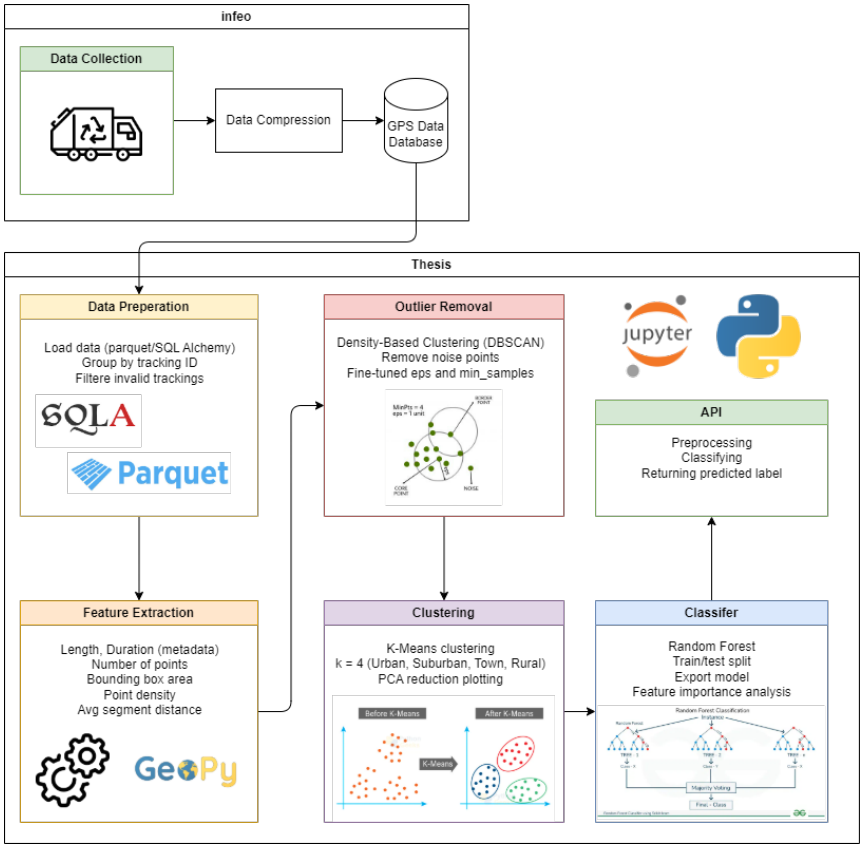


Figure 4.1.: Big picture implementation pipeline

4.1. Data Handling and Preprocessing

Efficient handling and preprocessing were essential to manage the large dataset and prepare it for the feature extraction step. This section describes the loading, filtering and preparing the raw GPS tracking data for feature extraction.

4.1.1. Data volume and loading

The data was provided via a MySQL database, containing two main tables relevant for this analysis. Trackings table containing the metadata of 101,353 trackings and the waypoints table containing the corresponding GPS waypoints of the trackings. Due to the substantial size of database **13.5 GB** in total, with **12.6 MB** from the tracking table and **11.6 GB** from the waypoint table, an efficient data handling pipeline was necessary.

To handle the data at a lower scale, the filtering query was executed directly within the database using SQL (see Listing 4.1). The resulting filtered dataset was streamed in chunks of `1,000,000` rows using `pandas.read_sql()` and saved incrementally to a compressed Apache Parquet file using `pyarrow`. The final Parquet file had a reduced size of approximately **1.5 GB**.

This chunk-wise approach ensured memory efficient loading without crashing the kernel with the limited hardware available. The full export operation took about two hours to fully complete, with an average processing speed of `6.40 iterations/second` (row group read and write cycles).

4.1.2. Filtering Faulty Trackings

The raw data included many invalid or irrelevant trackings that could impact the analysis and clustering. These trackings were removed based on these criteria:

- **Duration Filter:** Trackings shorter than 30 minutes or longer than 10 hours were excluded.
- **Length Filter:** Trackings with a total length lower than 50 km or above 850km were excluded.
- **Minimal Waypoint Count:** Trackings with less than 10 recorded waypoints were excluded.
- **Minimal Coordinate Span:** Trackings with less than 0.0005 span in latitude or longitude were excluded. Equaling to about 50 meters north-south and 37 meter east-west in the DACH region.

The filters were directly implemented as a SQL query executed in an jupyter notebook. This reduced the further computatinal times for clustering and training the classifier model. The SQL query implementing these filters as follows:

```
SELECT
    wp.id_tracking, wp.id, wp.time, wp.type, wp.sequence,
        wp.comment,
    wp.speed, wp.heading, wp.duration, wp.block_type, wp.
       log,
    wp.latitude, wp.longitude, wp.altitude, wp.meta_tag,
       wp.meta_value,
    t.length AS tracking_length,
    t.duration AS tracking_duration
FROM waypoint wp
JOIN tracking t ON wp.id_tracking = t.id
WHERE
    t.duration BETWEEN 18000000000 AND 360000000000
    -- 0.5 to 10 hours
    AND t.length BETWEEN 50 AND 850  -- 50 to 850 km
    AND EXISTS (
        SELECT 1 FROM waypoint w
        WHERE w.id_tracking = t.id
        HAVING COUNT(*) > 10  -- Min. 10 waypoints
    )
    AND (
        (SELECT MAX(latitude) FROM waypoint WHERE
           id_tracking = t.id) -
        (SELECT MIN(latitude) FROM waypoint WHERE
           id_tracking = t.id)
    ) > 0.0005  -- Min. ~50m latitude span
    AND (
        (SELECT MAX(longitude) FROM waypoint WHERE
           id_tracking = t.id) -
        (SELECT MIN(longitude) FROM waypoint WHERE
           id_tracking = t.id)
    ) > 0.0005;  -- Min. ~50m longitude span
```

Listing 4.1: SQL query used for filtering the GPS tracking dataset

In total the filtering steps reduced the dataset by 55755 (55%) trackings.

4.2. Feature Extraction

This section describes how raw GPS tracking data was transformed into meaningful features that enable the use of machine learning methods in later steps. The features were selected based on literature and domain knowledge to capture the characteristics of the waste collection routes. The resulting features were used for clustering and for training the supervised classifier. Each feature was extracted using Python libraries such as `pandas`, `numpy`, and `geopy` and stored in a CSV file.

4.2.1. Available Metadata Features

Metadata features are calculated by *infeo GmbH* and directly provided in the dataset for each tracking. They serve as a baseline for route length and duration.

4.2.1.0.1. Tracking length The total recorded travel distane in kilometers, as stored in the metadata.

4.2.1.0.2. Tracking duration The full duration from the first to the last recorded waypoint, in 100-nanosecond intervals.

4.2.2. Computed Features from Waypoints

The computed features enrich the metadata by describing the geometry and dynamics of each tracking in numerical values. All calculations were implemented using the `geopy` library for geodesic distances and `numpy` for vectorized operations. The most relevant features are described below.

4.2.2.0.1. Number of points The total number of waypoints n in a tracking:

$$num\_points = n$$

4.2.2.0.2. Bounding box area Computed from the geodesic height h and width w of the minimal enclosing rectangle:

$$bbox\_area = h \cdot w$$

$$h = d((lat_{\min}, long_{\min}), (lat_{\max}, long_{\min})), \quad w = d((lat_{\min}, lon_{\min}), (lat_{\min}, long_{\max}))$$

where $d(\cdot, \cdot)$ refers to the geodesic distance between two GPS points (in meters), calculated using the WGS84 ellipsoid model through the *geopy* Python library [16].

4.2.2.0.3. Point density

Density of GPS waypoints in the bounding box:

$$point\_density = \frac{n}{bbox\_area + \varepsilon}, \quad \varepsilon = 10^{-6}$$

where ε is a small constant to avoid division by zero.

4.2.2.0.4. Average segment distance

The average geodesic distance between consequtive waypoints:

$$avg\_segment\_distance = \frac{1}{n-1} \sum_{i=1}^{n-1} d(p_i, p_{i+1})$$

where $d(\cdot, \cdot)$ refers to geodesic distance.

4.2.2.0.5. Straightness

Ratio between the start-end distance and the total path length:

$$straightness = \frac{d(p_1, p_n)}{\sum_{i=1}^{n-1} d(p_i, p_{i+1})}$$

[17]

4.2.2.0.6. Mean Heading Change

Average angle change between consecutive direction vectors (in radians):

$$\delta_i = \min(|\theta_{i+1} - \theta_i|, 2\pi - |\theta_{i+1} - \theta_i|), \quad mean\_heading\_change = \frac{1}{n-2} \sum_{i=1}^{n-2} \delta_i$$

with bearings θ_i computed via:

$$\theta = \arctan 2\left(\sin(\Delta\lambda)\cos(\phi_2), \cos(\phi_1)\sin(\phi_2) - \sin(\phi_1)\cos(\phi_2)\cos(\Delta\lambda)\right)$$

[13]

4.2.3. Feature Storage and Output

The extracted features were stored in a structured CSV file, with one row per tracking and all corresponding features. This resulting dataset forms the basis for all future analysis, clustering and classification applications in the next steps of the pipeline.

Feature	Unit	Description
num_points	Count	Number of GPS waypoints in a tracking
bbox_area	m^2	Area of bounding box enclosing all GPS points
point_density	points/m^2	Number of points per square meter of bounding box
avg_segment_distance	m	Mean geodesic distance between consecutive waypoints
straightness	Ratio (0–1)	Direct distance from start to end / total distance
mean_heading_change	rad	Average absolute angle change between segments
duration	100 ns	Total tracking duration in nanoseconds
length	km	Total tracking length from the database

Table 4.1.: Extracted features with units and descriptions

4.3. Clustering Implementation

Through clustering the unlabeled trackings were labeled, enabling the further application of supervised machine learning in the next. A two stage approach was used, first DBSCAN for detecting and filtering outliers and second a K-Means clustering was applied to label the trackings. The implemented approach used a semi-supervised K-Means clustering. By calculating feature weights and inital centroid position for each cluster from a manually selected sample set to improve the clustering process.

4.3.1. Outlier Removal with DBSCAN

To remove extreme outliers from the dataset, DBSCAN (Density-Based Clusering of Applications with Noise) algorithm was used to filter outliers from the dataset. Outliers were assigned the cluster label of -1 and excluded from the dataset before applying the K-Means clustering.

It was chosen for its effective noise point detection, without the need of implementing manual logic.

Through testing and fine tuning the final implementation of DBSCAN used the following parameters:

- `eps` = 1
- `min_samples` = 40

DBSCAN removed 1,539 trackings from the dataset leaving a cleaner distribution for the K-Means clustering. This step was crucial for removing invalid trackings especially with large GPS gaps. Without the outlier removal, the subsequent K-Means clustering would always assign outliers to one class.

4.3.2. Semi-Supervised Clustering with Weighted and Seeded K-Means

To improve the interpretability and performance of clustering, a semi-supervised approach was used to guide the K-Means algorithm. A manually labeled subset of 8 samples, evenly distributed across the four target categories (`URBAN`, `SUBURBAN`, `TOWN`, `RURAL`), was utilized to derive feature weights and initialize cluster centroids. This allowed the unsupervised clustering algorithm to be seeded with domain knowledge.

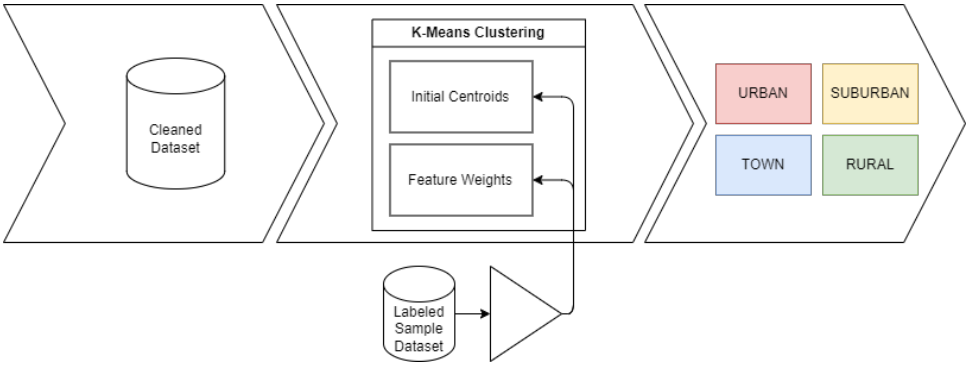


Figure 4.2.: K-Means semi supervised clustering pipeline

4.3.2.1. Feature Weighting Based on Sample Label Spread

In order to emphasize features that varied significantly across known classes and downweight features with less discriminative power, relative variability of feature medians between the categories was used. Specifically, for each feature f_i, the relative median range was calculated as:

$$w_i = \frac{max_i - min_i}{mean_i}$$

where max_i, min_i, and $mean_i$ represent the maximum, minimum, and mean median value of feature f_i across the four labeled classes. These weights were then normalized by dividing by the mean of all w_i to ensure comparability:

$$w_i^{norm} = \frac{w_i}{\frac{1}{n}\sum_{j=1}^{n} w_j}$$

Finally, the weights were clipped to the range $[0.5, 2.0]$ to avoid extreme scaling effects [18].

4.3.2.2. Seeded Cluster Initialization

The cluster centers were initialized using the scaled and weighted medians of the labeled sample set. For each class, the feature medians were calculated, scaled using the `StandardScaler`, and then multiplied by the learned feature weights. The resulting $4 \times d$ matrix was used to seed the k centroids of the K-Means algorithm:

$$\mu_k = scale(median_k) \cdot w_i^{norm} \quad foreach k \in \{1, 2, 3, 4\}$$

This strategy ensured that the initial centroids already reflected meaningful differences in the data as identified by the sample set.

4.3.2.3. Constrained K-Means Clustering

The clustering was performed using the `KMeansConstrained` algorithm with the following configuration:

- Number of clusters: 4
- Initialization: Manual (from scaled, weighted medians)
- Minimum cluster size: 5% of total data
- Maximum cluster size: 60% of total data
- Number of restarts: 10

This method produced clusters that closely aligned with the known class distribution and improved the stability and interpretability of the clustering outcome compared to random initialization.

4.4. Classification Model

To predict the category of new or unlabeled GPS trackings, a classification model was trained using the extracted and clustered features. A Random Forest classifier was chosen due to its robustness, ability to handle high-dimensional data, and interpretability via feature importance.

4.4.1. Model Selection and Training

The full labeled dataset, enriched with cluster labels from the K-Means algorithm, was split into a training and a test set using a 2:1 ratio. A grid search with 3-fold cross-validation was performed to identify the best hyperparameters for the Random Forest model. The following parameter grid was used:

- Number of estimators: `[100, 200, 300]`
- Maximum depth: `[10, 15, 20]`
- Minimum samples split: `[2, 5]`
- Minimum samples leaf: `[1, 2]`
- Class weight: `[None, 'balanced']`

The best model was found with 300 estimators, a maximum depth of 20, 2 minimum smaples split, 1 minimum sample leaf and default class weights. The trained model achieved an accuracy of 98.3% on the test set.

4.4.2. Saving and Exporting the Classifier

The trained Random Forest model was serialized using `joblib` and exported as a PKL file. This model can be directly integrated into production pipelines or reused for further research. Predictions from the classifier were stored for visualization and evaluation purposes in a CSV file.

4.5. Integration with API

To provide real-time classification, a FastAPI-based web service was implemented (Figure 4.3). The service exposes a single endpoint which takes a `tracking_id` as an input, fetches the corresponding tracking data from the *infeo GmbH* API, and processes the tracking data through the same preprocessing pipeline used during the model training.

The API loads the trained classifier from a serialized PKL file and predicts the label of processed tracking. The endpoint then returns the `tracking_id` and the predicted class label in a *json* format.

Figure 4.4 shows the automatically generated *Swagger UI* documentation of the endpoint.

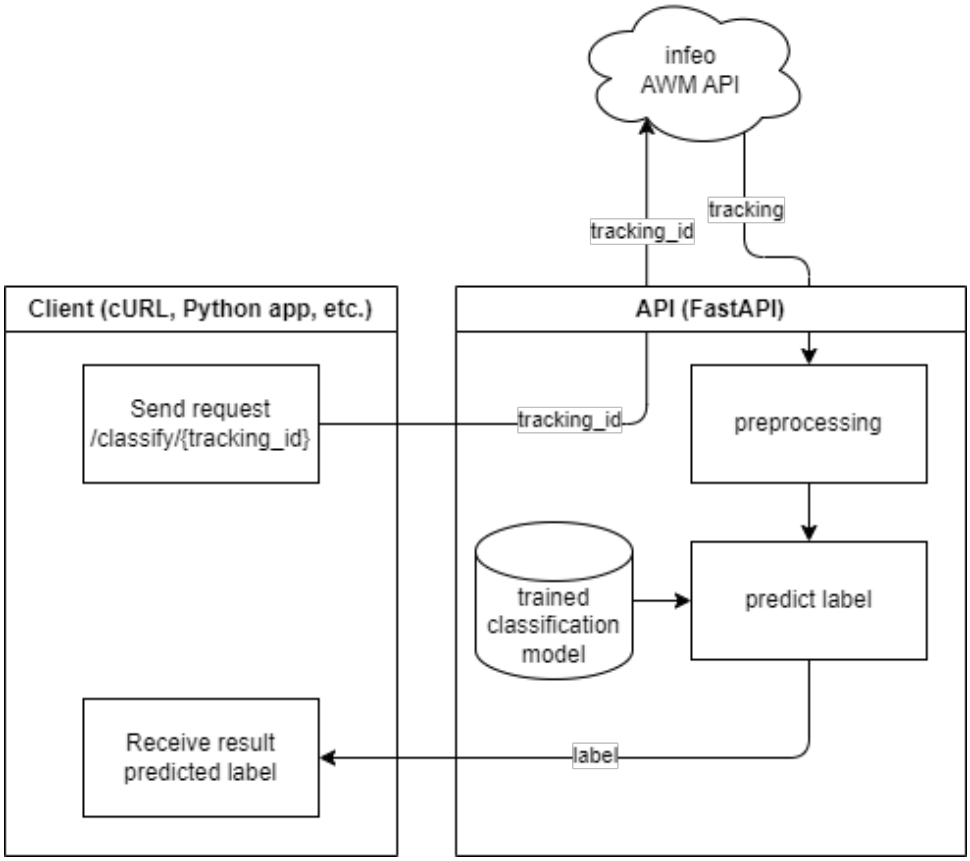


Figure 4.3.: API Implementation Diagram

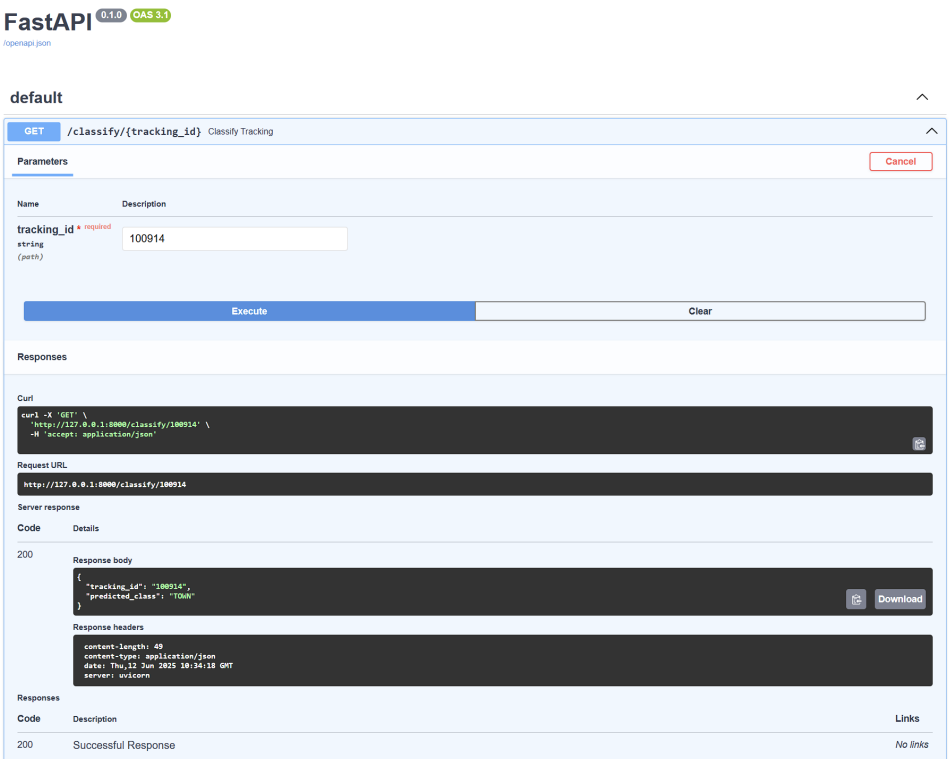


Figure 4.4.: Swagger UI documentation of FastAPI Classification Endpoint

5. Evaluation and Discussion

This chapter presents a comprehensive evaluation of the filtering, clustering and classification results. It provides insights into the performance, limitations, and robustess of the developed system.

5.1. Definition of the data sets used for the evaluation

To assess the quality of the clustering and classification steps, various datasets at different stages of the pipeline were used. The final labeled dataset was split into training and test sets using a 2:1 ratio to evaluate the classifier's generalization ability.

5.2. Evaluation of the Results

This section provides the rsults of each step in the pipeline, highlighting numerical results and visual representations of classified trackings.

5.2.1. Filtering Results

Each Filter step in the SQL statement reduces the dataset by this many trackings:

Filter Step	**Remaining Trackings**	**Removed**	**Reduction (%)**
All trackings	101353	–	–
Duration filter	66477	34876	34.410%
Length filter	45668	20809	31.303%
Minimal Waypoints filter	45599	69	0.151%
Min Lat/Long span filter	45598	1	0.002%
Total Remaining	**45598**	**55755**	**55.011%**

Table 5.1.: Tracking reduction per filtering step

The filtering was effective at reducing the amount of faulty trackings in the dataset, but could not filter out all of them. The latitude and longitude span suggested by *infeo GmbH* did not filter out many trackings. This could be due to the filtering of stationary trackings from the filters applied before the coordinate span filter, or indicate the need to increaste the minimum span.

5.2.2. Clustering Results

The DBSCAN clustering algorithm was used to remove noise points and outliers, resulting in the removal of 1,539 trackings. The PCA projection in Figure 5.1 shows the detected outlier cluster (-1), sbusequently removed resulting in a cleaner structure for the K-Means clustering.

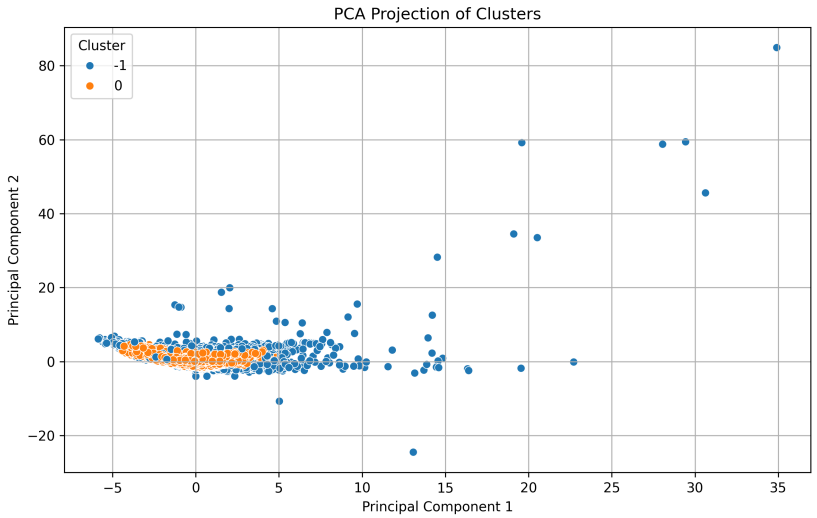


Figure 5.1.: PCA Projection of DBSCAN clustering for noise removal

The feature weights calulated from the manually selected sample data showed that `bbox_area`, `point_density`, `straigthness` and `length` were the most informative features. The concrete values are depipicted in Figure 5.2.

Weighted K-Means clustering with custom initial centroids produced four distinct clusters. Each cluster was mapped to a label (`0-URBAN`, `1-SUBURBAN`, `2-TOWN`, `3-RURAL`) based on feature statistic and visual inspection. The PCA projection in Figure 5.3 visualized the resulting cluster structure, with a final silhouette score of **0.2621** indicates a moderate clustering separation.

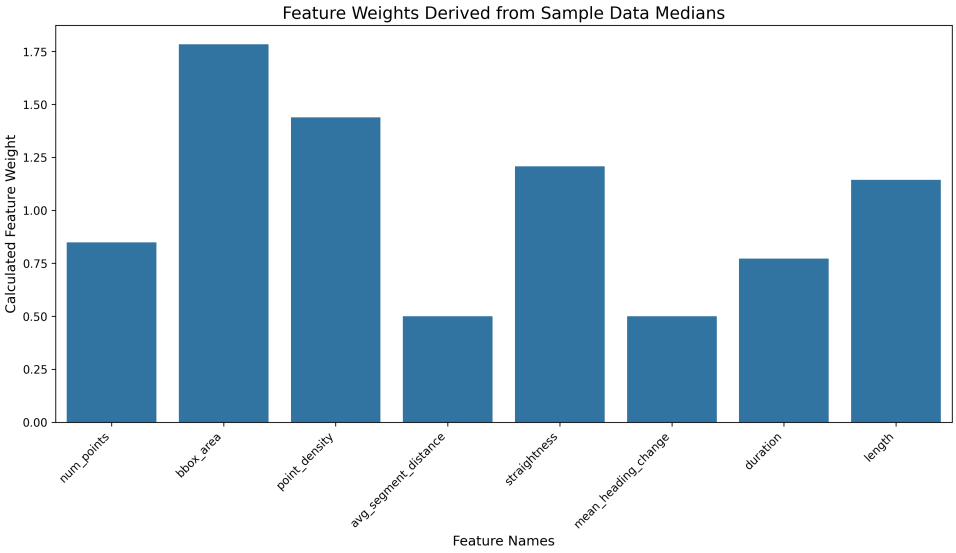


Figure 5.2.: Feature Weights used for K-Means clustering

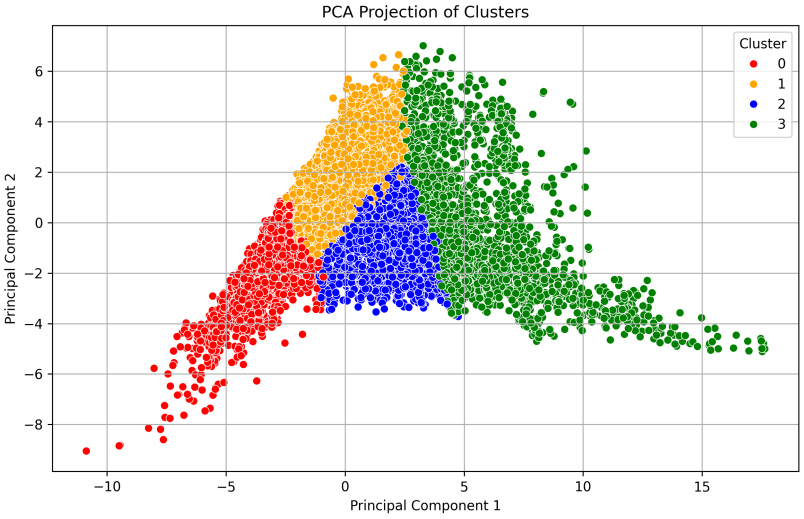


Figure 5.3.: PCA Projection of K-Means clustering

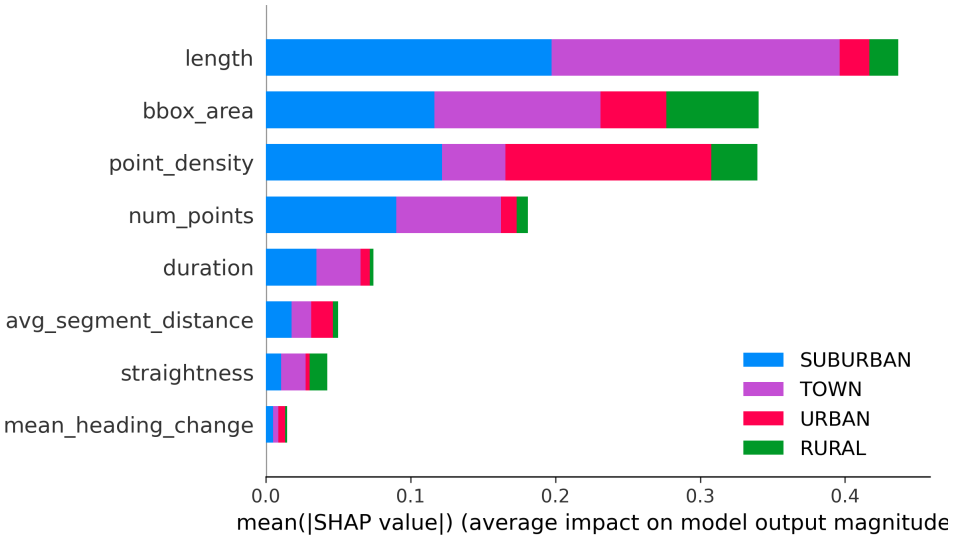
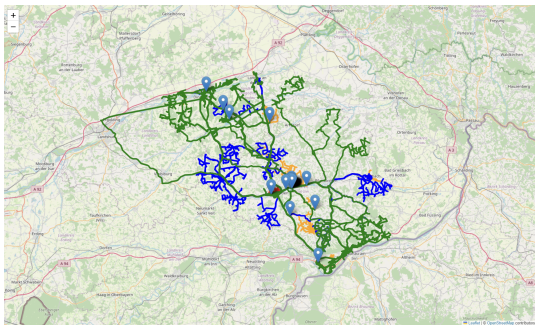
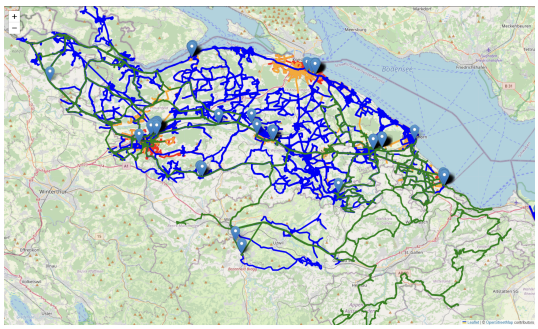


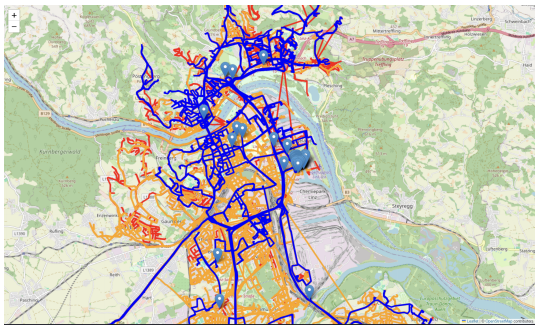
Figure 5.4.: SHAP Summary for all Classes



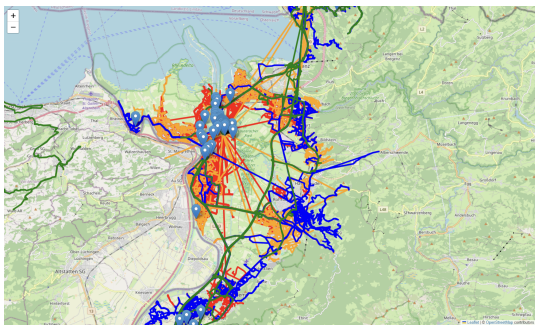
(a) Lower Bavaria



(b) Thurgau: Kreuzlingen and Konstanz



(c) Upper Austria: Linz



(d) Vorarlberg

Figure 5.5.: Map Visualization of Clustered Trackings (Red: `URBAN`, Orange: `SUBURBAN`, Blue: `TOWN`, Green: `RURAL`)

5.2.3. Classifier Results

A Random Forest classifier was trained with hyperparameter tuning. The model achieved high performance on the test set: Accuracy **98.3%**, Macro F1-score: **98.1%**, Weighted F1-score **98.3%** and Cohen's Kappa **97.3%**. The classifier showed balanced performance across all four classes, with precision and recall values above **97%** for all classes.

The feature importance analysis (Figure 5.7) highlighted [length], `bbox_area`, `point_density` and `num_points` as key contributing features.

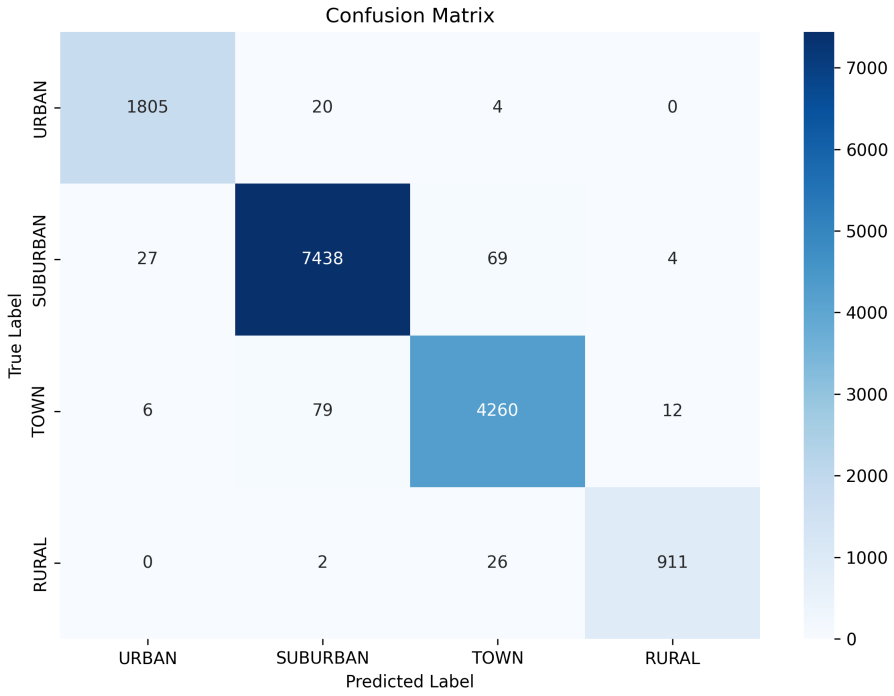


Figure 5.6.: Confusion Matrix of Random Forest Classifier

The mean feature values across all four classes (Figure 5.8) confirm a monotonic relationships between classes in increasing rurality (e.g.,`URBAN` to `SUBURBAN` to `TOWN` to `RURAL`) for the features `bbox_area`, `point_density`, `avg_segment_distance` and `length`.

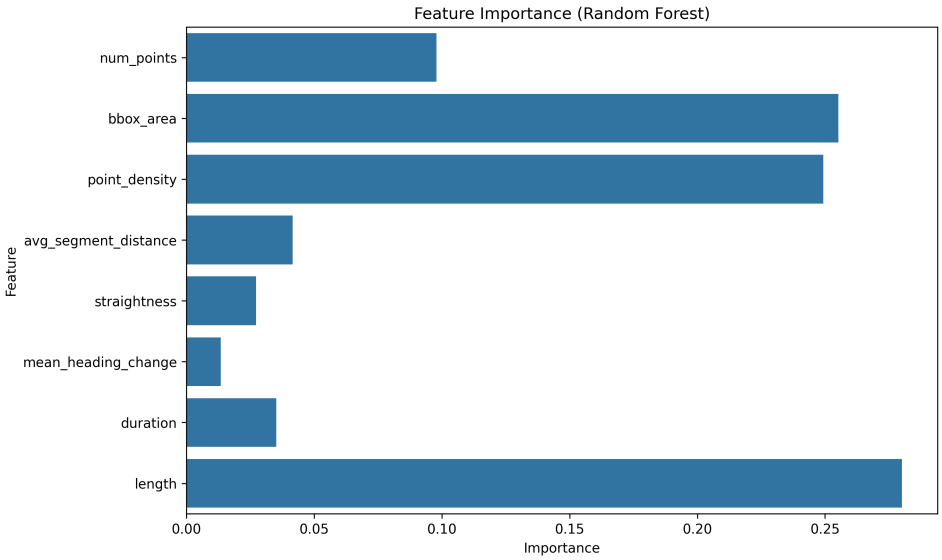


Figure 5.7.: Feature Importance Barchart of Random Forest Classifier

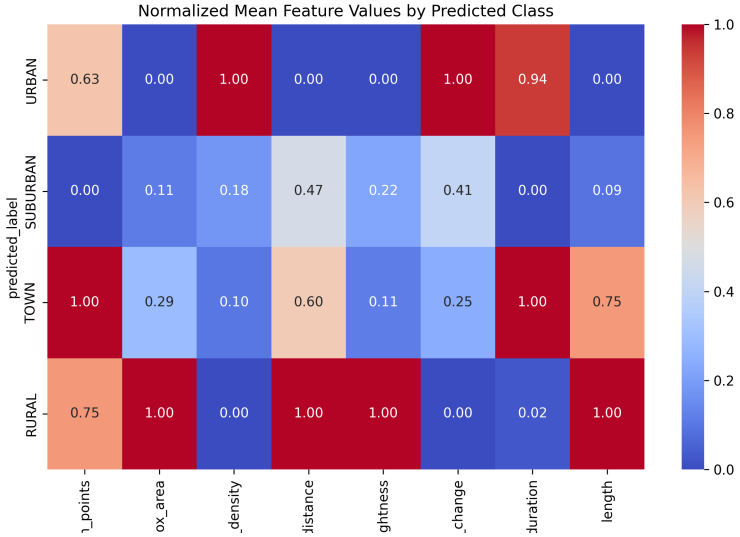


Figure 5.8.: Normalized Mean Feature Values by Predicted Class

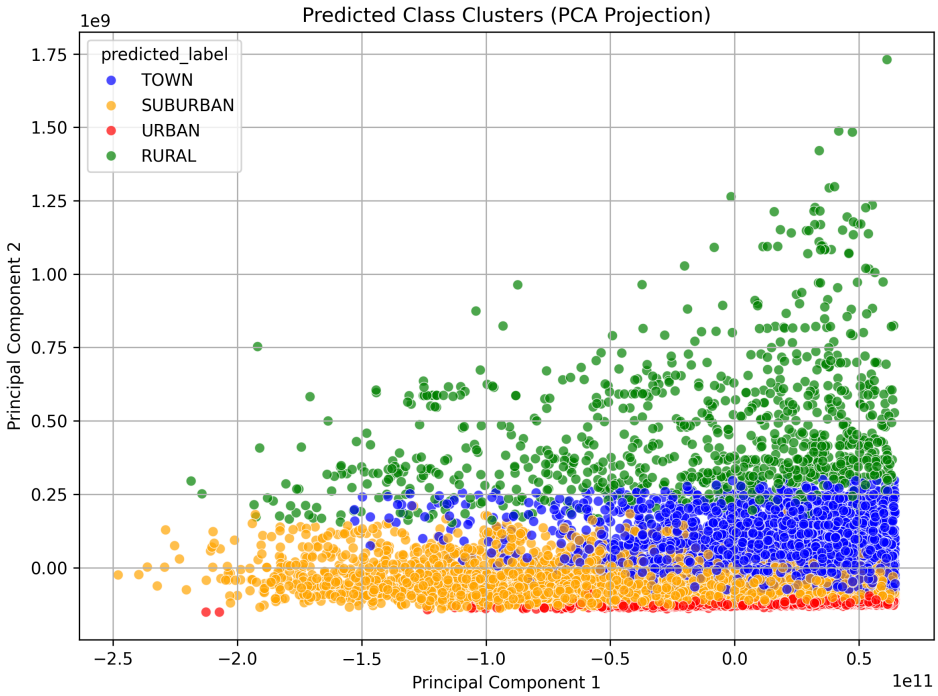


Figure 5.9.: PCA Projection of Test Set with Predicted Labels

Finally four random trackings from each category were selected from the classified test set and visualized on map segments 5.12, validating the structural and spatial differences of the labels.

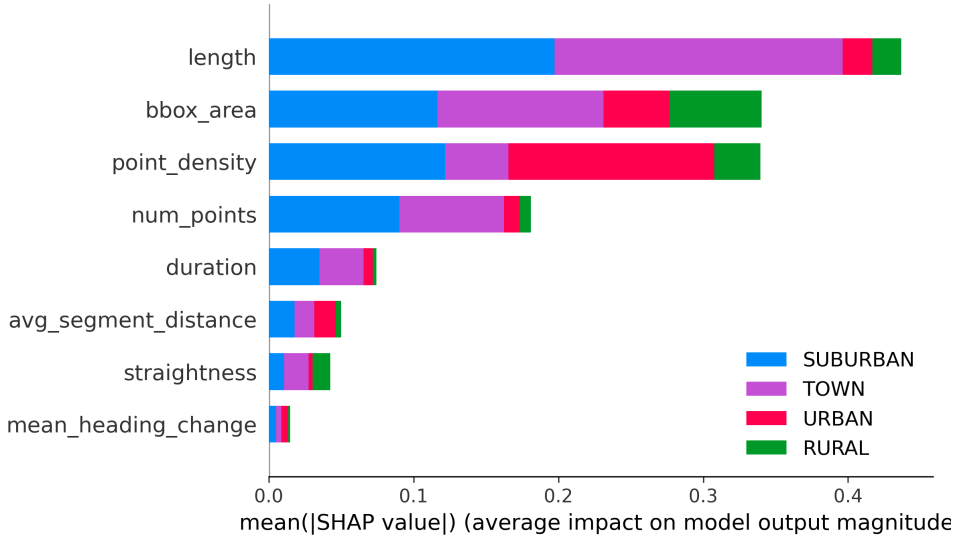
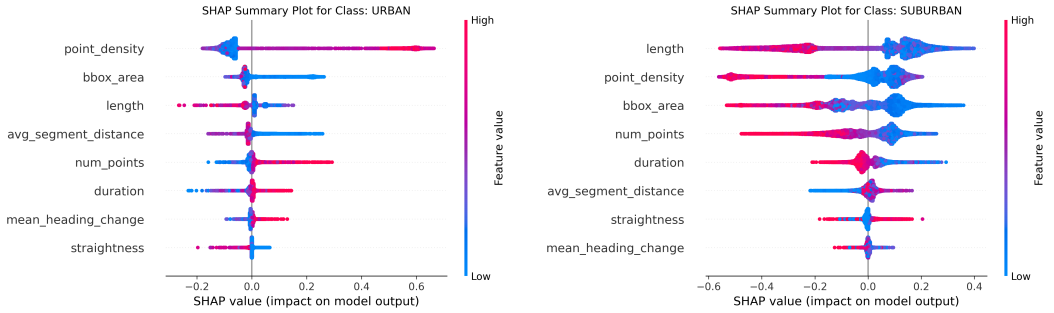
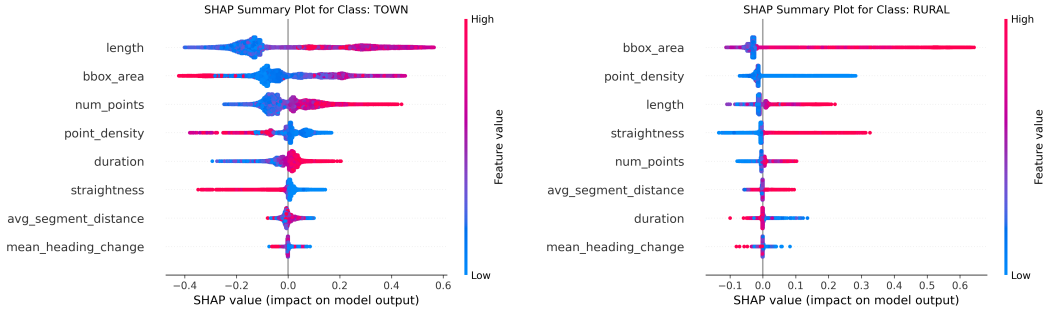


Figure 5.10.: SHAP Summary for all Classes



(a) Urban

(b) Suburban



(c) Town

(d) Rural

Figure 5.11.: SHAP summary plots for all classes

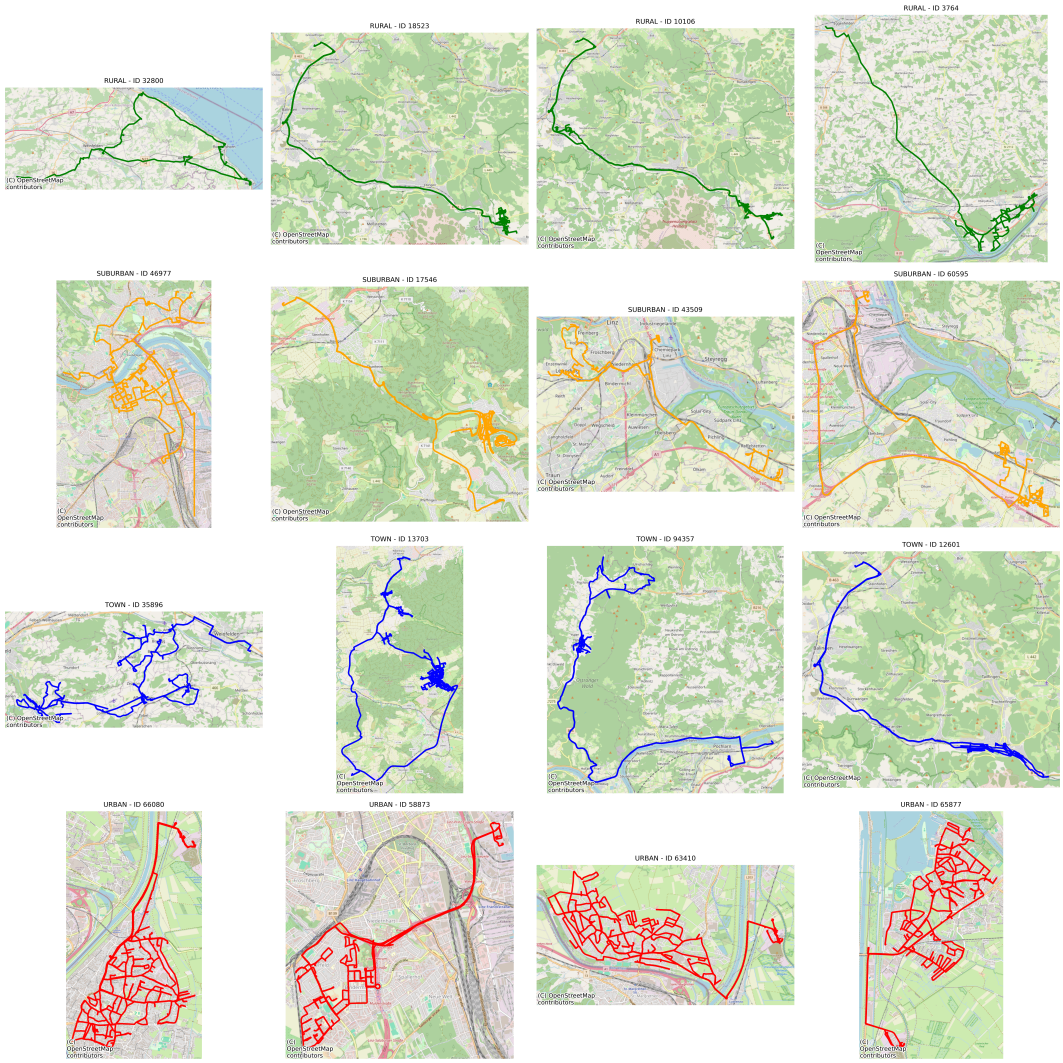


Figure 5.12.: Sample Predictions on Test Set

5.3. Reflection on the Results

The clustering and classification pipeline effectively distinguishes between urban structures based on GPS tracking data, with the classifier achieving high accuracy with over **97%** in all key metrics. While the clustering silhouette score was moderate, visual and statistical inspection validate the coherence of the clusters.

Despite the promising results, some limitations of the evaluation setup must be considered. The classifier was trained and tested using labels generated through semi-supervised K-Means clustering. While the clustering was guided by meaningful initial centroids and validated visually and statistically, the resulting labels do not represent objective ground truth. Consequently, the classifier's high accuracy of **98.3%** reflects strong alignment with the underlying clustering logic rather than verified real-world classifications. The model has yet to be validated against expert-labeled or GIS-based reference data, and its generalizability beyond the current dataset remains an open question.

The negative silhouette score on the test set shows some significant overlap between classes in the high-dimensional space. This could be adressed in future iterations by reducing te number of clusters (e.g., merging `SUBURBAN` and `TOWN`) or by incorporating additional features from OpenStreetMap.

The features `bbox_area`, `point_density` and `duration`, initially deemed suitable for distinction between classes, because of their monotonic behavior, were confirmed to be indicative discriminators. Furthermore, the classification results show that `avg_segment_distance` and `mean_heading_change` also follow monotonic behavior among classes, reinforcing the effectiveness of this pipeline.

Overall, the presented method offers a reproducible and scalable approach for GPS-based route classification, with potential applications in waste collection planning, infrastructure analysis, and beyond. With further validation and integration of additional geospatial data, the framework has the potential to evolve into a robust decision support tool.

5.4. Expert Feedback

During the development of the thesis, a meeting with Reischer Florian the founder and CEO of `infeo GmbH` was held. The of the meeting was to present intermediate results and gather feedback on the applicability of the developed pipeline in a production setting.

Reischer emphasized that the classification of the GPS trackings into structural categories is a solid starting point, the dataset in this thesis alone is

not sufficient for the comparison between new and existing service areas. He pointed out, that for a final deployable solution, additional data sources, such as detailed road networks or geographic information such as population, road and intersection count are needed. With that additional information, accurate comparisons between existing trackings and new areas seem possible for automating the planning support.

A key point of discussion was the feature `point_density`, which showed strong discriminative power in the analysis and played a central role in both clustering and classification (see Figures **??**).

However, Reischer was concerned about the robustness of calculated features such as `point_density`, as these features rely on the amount of points present in the dataset, which is impacted by the internal compression by *infeo GmbH*. This compression removed points with the same coordinates and thinned out point on long straights to reduce filesizes.

While `point_density` proved to be useful in this thesis, its reliability might be limited. Comparison with non-compressed trackings could determine how the compression step impacts featuress reliant on the amount of waypoints, such as `poin_density`.

Overall, while further progress is necessary for a live deployment, Reischer concluded that this thesis gatheres important information and builds a strong foundation for the long-term goal of automating area comparison and suggesting planning parameters.

6. Conclusion

This thesis demonstrates that it is feasible to classify the structure of road networks using only GPS tracking data collected by waste collection vehicles. A robust pipeline to categorize trackings into four distinct urban structure types: `URBAN`, `SUBURBAN`, `TOWN` and `RURAL` was developed using feature engineering, semi-supervised clustering, and supervised classification.

6.1. Future Directions

Several opportunities exist to improve and extend this thesis.

One promising direction involves the integration of OpenStreetMap (OSM) data to augment the GPS based features with real-world structural information. On the one hand this could improve the classifier's ability to differentiate between trackings with similar GPS structure, while on the other hand enabling the comparison with areas where no data is collected.

Additionally, enhancing the preprocessing pipeline with a filter to remove transit segmets between depots and services areas in the trackings can focus the analysis on the actual waste collection behavior.

Improving the selected sample for the semi-supervised clustering process, both in terms of amount and variation of manually labeled trackings could improve the feature weights and initial centroids heavily affecting the clustering process.

Finally, deploying the trained clasifier as a REST API, would enable waste management companies to automatically classify GPS trackings in real-time, potentially enabling smarter planning, anomaly detection, and route optimization.

6.2. Limitations

Despite the promising results of the classification pipeline, several limitations must be acknoladged.

Firstly, the dataset used in this thesis is not raw GPS data but rather a compressed and preprocessed version provided by *infeo GmbH*. While this process

is necessary for storage and performance, it might have removed information that could improve the clustering and classification accuracy and reveal more fine grained distinction between the categories.

Secondly, while the dataset provides accurate GPS coordinates, the metadata is limited or unreliable. Attributes such as speed are missing entirely, and loading and unloading events were only sparsely available. As a result only GPS-based features were used for feature extraction, potentially limiting the detection of behavior patterns initially deemed as valuable differentiators between classes, such as distance between loadings of the vehicles.

Furthermore, the dataset is not suitable for a direct comparison with OpenStreetMap (OSM) street data. Such a comparison would require the enrichment of the dataset with strucutral data from OSM or a complete derivation of route segments from OSM.

Lastly, the classification pipeline relies heavily on the semi-supervised clustering to generate labels. While it is effective for this dataset, it might require adjustments when extending the dataset to other cities and regions.

6.3. Final Remarks

Overall, this thesis provides a practical framework for analysing and classifying GPS tracking data in the waste management domain. It bridges the gap between raw GPS data and operational insights, laying the foundation for smarter, data-driven waste management systems.

Bibliography

[1] "The world's most valuable resource is no longer oil, but data," *The Economist*, ISSN: 0013-0613. [Online]. Available: https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data (visited on 04/06/2025).

[2] T. Petroc, *Data growth worldwide 2010-2028*, en. [Online]. Available: https://www.statista.com/statistics/871513/worldwide-data-created/ (visited on 04/11/2025).

[3] *Europe Waste Management Market Size and Share Analysis 2030*, en. [Online]. Available: https://www.nextmsc.com/report/europe-waste-management-market (visited on 04/24/2025).

[4] *Solid waste management*, en. [Online]. Available: https://www.eib.org/en/projects/topics/energy-natural-resources/solid-waste/index (visited on 04/24/2025).

[5] *Gemeinsam mit unseren Kunden revolutionieren wir die digitale Kreislaufwirtschaft*, de-AT. [Online]. Available: https://infeo.at/index.php/revolution/ (visited on 04/24/2025).

[6] K. P. Sinaga and M.-S. Yang, "(PDF) Unsupervised K-Means Clustering Algorithm," en, *ResearchGate*, Dec. 2024. DOI: 10.1109/ACCESS.2020.2988796. [Online]. Available: https://www.researchgate.net/publication/340813602_Unsupervised_K-Means_Clustering_Algorithm (visited on 04/25/2025).

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," en,

[8] J. Liang, J. Cui, J. Wang, and W. Wei, "Graph-based semi-supervised learning via improving the quality of the graph dynamically," en, *ResearchGate*, Dec. 2024. DOI: 10.1007/s10994-021-05975-y. [Online]. Available: https://www.researchgate.net/publication/351571830_Graph-based_semi-supervised_learning_via_improving_the_quality_of_the_graph_dynamically (visited on 04/25/2025).

[9] F. Voron, *Building data science applications with FastAPI: develop, manage, and deploy efficient machine learning applications with Python, second edition*, en, 2nd ed. Place of publication not identified: Packt Publishing, 2023, ISBN: 978-1-83763-274-9 978-1-83763-726-3.

[10] *Deploying ML Models as API using FastAPI*, en-US, Section: Machine Learning. [Online]. Available: https://www.geeksforgeeks.org/deploying-ml-models-as-api-using-fastapi/ (visited on 06/12/2025).

[11] Y. Song, Y. Zhang, and D. Han, "Access structure," EN, *Environment and Planning B: Urban Analytics and City Science*, vol. 48, no. 9, pp. 2808–2826, Nov. 2021, Publisher: SAGE Publications Ltd STM, ISSN: 2399-8083. DOI: 10.1177/2399808320988560. [Online]. Available: https://doi.org/10.1177/2399808320988560 (visited on 06/12/2025).

[12] N. Zhou, "Research on urban spatial structure based on the dual constraints of geographic environment and POI big data," *Journal of King Saud University - Science*, vol. 34, no. 3, p. 101 887, Apr. 2022, ISSN: 1018-3647. DOI: 10.1016/j.jksus.2022.101887. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1018364722000684 (visited on 06/12/2025).

[13] M. Etemad, A. S. Junior, and S. Matwin, "Predicting Transportation Modes of GPS Trajectories using Feature Engineering and Noise Removal," in vol. 10832, arXiv:1802.10164 [cs], 2018, pp. 259–264. DOI: 10.1007/978-3-319-89656-4_24. [Online]. Available: http://arxiv.org/abs/1802.10164 (visited on 06/05/2025).

[14] Z. Koh, Y. Zhou, B. P. L. Lau, R. Liu, K. H. Chong, and C. Yuen, *Clustering and Analysis of GPS Trajectory Data using Distance-based Features*, arXiv:2212.00206 [cs], Dec. 2022. DOI: 10.48550/arXiv.2212.00206. [Online]. Available: http://arxiv.org/abs/2212.00206 (visited on 06/05/2025).

[15] *Route zum Nachfahren bereitstellen – awm.* [Online]. Available: https://wiki.awm.infeo.at/index.php/Route_zum_Nachfahren_bereitstellen#Aufzeichnung_komprimieren (visited on 06/05/2025).

[16] *Welcome to GeoPy's documentation! — GeoPy 2.4.1 documentation.* [Online]. Available: https://geopy.readthedocs.io/en/stable/ (visited on 06/11/2025).

[17] S. Benhamou, "How to reliably estimate the tortuosity of an animal's path:: Straightness, sinuosity, or fractal dimension?" *Journal of Theoretical Biology*, vol. 229, no. 2, pp. 209–220, Jul. 2004, ISSN: 0022-5193. DOI: 10.1016/j.jtbi.2004.03.016. [Online]. Available: https://www.

sciencedirect.com/science/article/pii/S0022519304001353 (visited on 06/11/2025).

[18] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," en,

A. Data Sheet on the Use of AI

Creatiion process Working steps	**have used AI**	**AI tool(s)**	**Experiences / recommendations / irritations**
Find a topic idea	no	-	-
Narrow down topic / Formulate question	no	-	-
Find sources	yes	OpenAI GPT-4o, Perplexity.ai	Helped identify relevant keywords and topic clusters.
Explain terms	yes	OpenAI GPT-4o	Useful for quick definitions and simple explanations.
Design text structure	yes	OpenAI GPT-4o	Good for outlining sections. Required manual adjustments.
Have content read aloud	no	-	-
Translate content	yes	DeepL	Very good at translating between english and german.
Dictate content	no	-	-
Paraphrase content, summarise	yes	OpenAI GPT-4o	Helpful to generate concise summaries of long text.
Write introduction	yes	OpenAI GPT-4o	Provided inspiration but required rewording.
Write main chapter	yes	OpenAI GPT-4o	Used for structuring and rephrasing, not full writing.
Write a summary	yes	OpenAI GPT-4o	Used to summarize main points effectively.
Obtain text feedback	yes	OpenAI GPT-4o	Used to review tone, clarity, and consistency.
Revise text statement	yes	OpenAI GPT-4o	Helpful for reformulating statements.
Revise text formulation	yes	OpenAI GPT-4o	Polished sentences and improved flow.
Correct text formally	yes	OpenAI GPT-4o	Assisted with grammar and punctuation checking.
Code generation	yes	OpenAI GPT-4o, Claude Sonnet 4	Assisted with generating code and fixing bugs

Table A.1.: Data sheet on the use of AI

Affidavit

I hereby declare in lieu of oath that I have written this Bachelor thesis independently and without the use of aids other than those specified. The passages taken directly or indirectly from other sources directly or indirectly from other sources are marked as such. The thesis has not been neither in the same nor in a similar form to any other examination authority nor has it been published.

Dornbirn, on 15. May 2025 Matthias Hefel