# Clustering and Federated Learning in V2V adhoc Network

Bibek Paudyal
bibek.paudyal@edu.univ-fcomte.fr

Mohhomad saim khan
mohammad_saim.khan@edu.univ-fcomte.fr

Supervisors
Prof . Oumaya Baala
oumaya.baala@utbm.fr

Prof .Cherifa Boucetta
cherifa.boucetta@univ-eiffel.fr

# Abstract

This study explores the potential of  intelligent based technique for clustering on the fast moving vehicles and federated learning to enhance communication between ad-hoc vehicular networks (VANETs) and  potential use case of the federated learning for use high speed data transmitted from the vehicles .

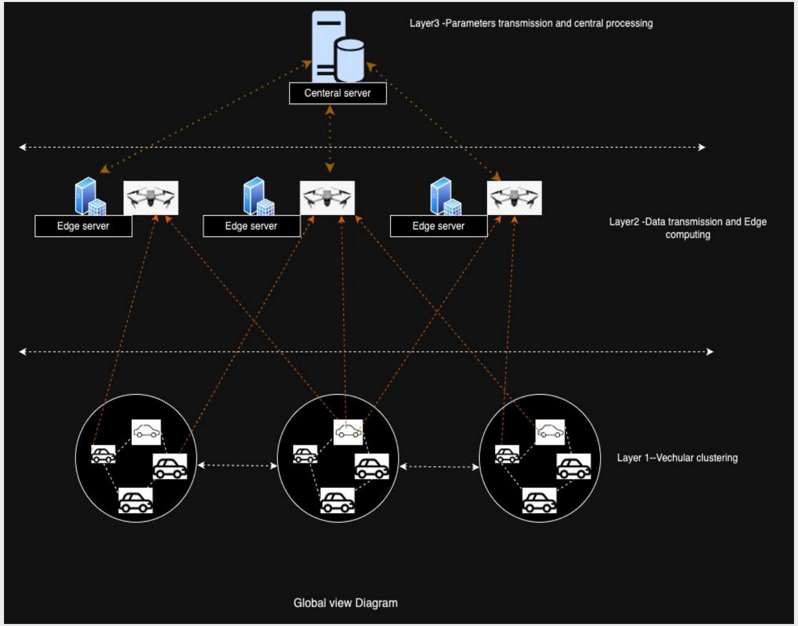
The study was conducted with the real time simulation network called sumo(SUMO) where 50 vehicles were simulated using the open street map of berlin  for 20,000 sec and collected the data from the vechiles for the study.

# Introduction

Intelligent transportation systems (ITS) are transforming traffic management by enabling cooperative interactions between vehicles and infrastructure.Ad-hoc vehicular networks (VANETs) play a key role in ITS, allowing vehicles to communicate directly and share information. Unmanned Aerial Vehicles (UAVs), with their maneuverability and mobility, offer exciting possibilities for enhancing data.

This project comprises the study of the vehicular communication in the fast moving vehicles and effective communication and data transfers and use of federated learning for the purpose of the edge computing to preserve the privacy and maintain the accuracy of the model.The vehicles are clustered using several machine learning approach for the real time communication and use of Edges for the test of the federated learning models

# Global View Architecture




**Overview**: The architecture integrates UAVs and vehicular networks to enable real-time data sharing and decision-making.

**Components**: Vehicles and UAVs equipped with sensors and communication modules.

**Communication:** Federated learning allows each edge UAVs to train local models and share updates without compromising privacy.

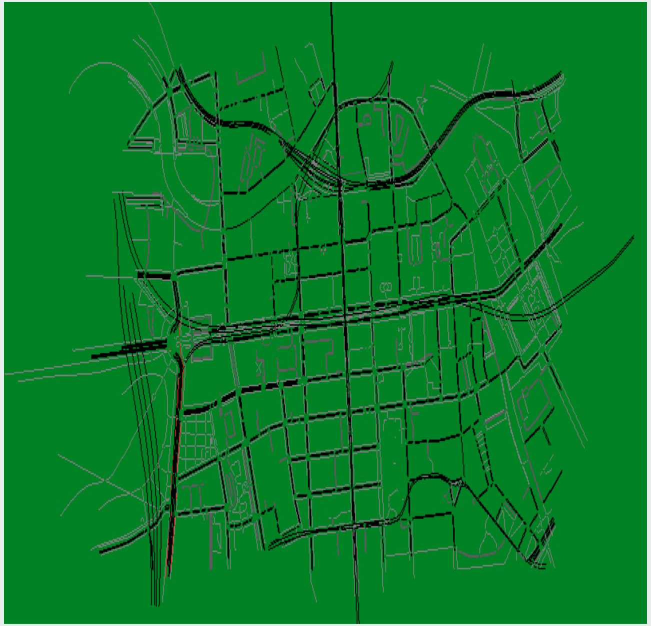
# Data Extraction and pre-processing

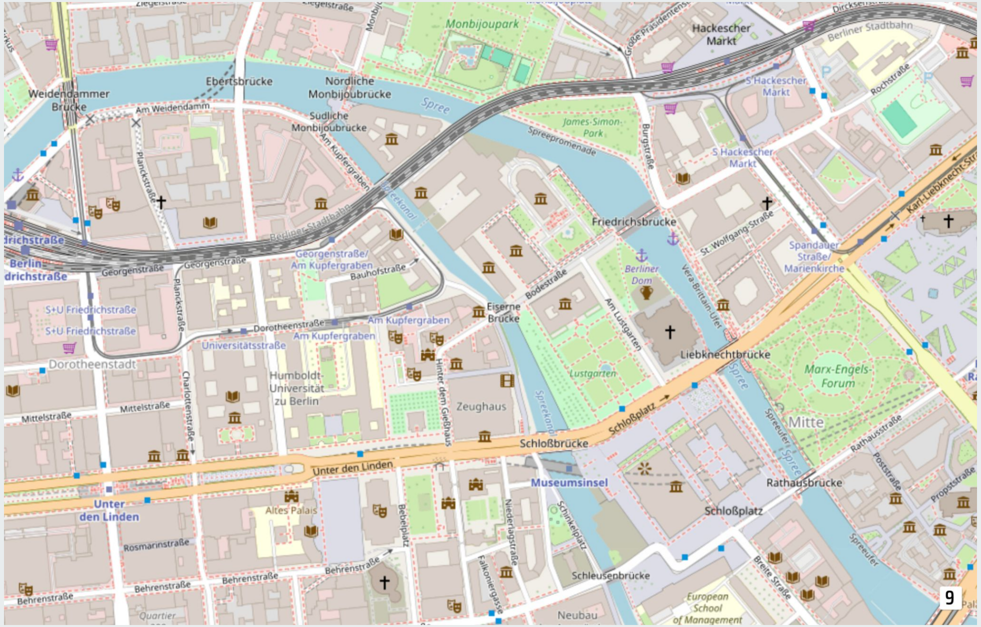
The initial step was to extract the data from the vehicles and used for the pre-preprocessing  using SUMO.**SUMO**, standing for Simulation of Urban MObility, is an open-source, microscopic traffic simulation package designed to model the behavior of individual vehicles and pedestrians within an urban environment. It offers a comprehensive suite of features for creating realistic traffic scenarios, simulating different transportation modes, and analyzing traffic flow dynamics.

50 vehicles were simulated for 20,000 sec in the real time enviroment within the map of Berlin.

Extracted Parameters

# OSM and SUMO extracted map of Berlin





# Extracted Features for Clustering

```
   time   id        x        y   angle  speed   pos            lane  slope
0   0.0  0.0  2569.42  1692.44  356.56   0.00  5.10  195397561#5_0    0.0
1   1.0  0.0  2569.34  1693.81  356.56   1.38  6.48  195397561#5_0    0.0
2   1.0  1.0  2452.87   830.62  265.12   0.00  5.10   -4588219#1_0    0.0
3   2.0  0.0  2569.18  1696.51  356.53   2.70  9.18  195397561#5_0    0.0
4   2.0  1.0  2451.29   830.48  265.12   1.58  6.68   -4588219#1_0    0.0
<class 'pandas.core.frame.DataFrame'>
<bound method NDFrame.describe of         time    id        x        y   angle
0        0.0   0.0  2569.42  1692.44  356.56   0.00   5.10  195397561#5_0
1        1.0   0.0  2569.34  1693.81  356.56   1.38   6.48  195397561#5_0
2        1.0   1.0  2452.87   830.62  265.12   0.00   5.10   -4588219#1_0
3        2.0   0.0  2569.18  1696.51  356.53   2.70   9.18  195397561#5_0
4        2.0   1.0  2451.29   830.48  265.12   1.58   6.68   -4588219#1_0
...      ...   ...      ...      ...     ...    ...    ...            ...
7816   384.0  49.0  2872.09  1488.30   80.06   8.41  26.16    4610479#0_0
7817   385.0  49.0  2879.96  1489.68   80.06   7.99  34.16    4610479#0_0
7818   386.0  49.0  2888.84  1491.24   80.06   9.01  43.17    4610479#0_0
7819   387.0  49.0  2897.21  1492.70   80.06   8.49  51.66    4610479#0_0
7820   388.0  49.0  2905.07  1494.08   80.06   7.99  59.65    4610479#0_0

      slope
0       0.0
1       0.0
2       0.0
3       0.0
4       0.0
...     ...
7816    0.0
7817    0.0
7818    0.0
7819    0.0
7820    0.0
```

1. **time**: Represents the time attribute of the vehicle's state.
2. **id**: Indicates the unique identifier of the vehicle. It's stored as an integer.
3. **x**: Represents the x-coordinate of the vehicle's position.
4. **y**: Represents the y-coordinate of the vehicle's position.
5. **angle**: Indicates the angle (in degrees) of the vehicle's orientation.
6. **speed**: Represents the speed of the vehicle.
7. **pos**: Indicates the position of the vehicle along its route.
8. **lane**: Represents the lane identifier in which the vehicle is located.
9. **slope**: Indicates the slope of the vehicle's path.

(7821 rows * 9 columns)

# Extracted Features for Federated Learning

| | time | id | eclass | CO2 | CO | HC | NOx | PMx | fuel \ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.00 | 0 | HBEFA3/PC_G_EU4 | 2624.72 | 164.78 | 0.81 | 1.20 | 0.07 | 837.22 |
| 1 | 1.00 | 0 | HBEFA3/PC_G_EU4 | 2955.05 | 148.04 | 0.74 | 1.32 | 0.07 | 942.57 |
| 2 | 1.00 | 1 | HBEFA3/PC_G_EU4 | 2624.72 | 164.78 | 0.81 | 1.20 | 0.07 | 837.22 |
| 3 | 2.00 | 0 | HBEFA3/PC_G_EU4 | 3267.70 | 133.26 | 0.68 | 1.43 | 0.07 | 1042.28 |
| 4 | 2.00 | 1 | HBEFA3/PC_G_EU4 | 3089.80 | 147.47 | 0.74 | 1.38 | 0.07 | 985.55 |

| | electricity | noise | route | type | waiting | lane | pos \ |
|---|---|---|---|---|---|---|---|
| 0 | 0.0 | 55.94 | !0 | DEFAULT_VEHTYPE | 0.0 | 195397561#5_0 | 5.10 |
| 1 | 0.0 | 62.52 | !0 | DEFAULT_VEHTYPE | 0.0 | 195397561#5_0 | 6.48 |
| 2 | 0.0 | 55.94 | !1 | DEFAULT_VEHTYPE | 0.0 | -4588219#1_0 | 5.10 |
| 3 | 0.0 | 62.85 | !0 | DEFAULT_VEHTYPE | 0.0 | 195397561#5_0 | 9.18 |
| 4 | 0.0 | 63.49 | !1 | DEFAULT_VEHTYPE | 0.0 | -4588219#1_0 | 6.68 |

1. **CO2**: Emission level of carbon dioxide.
2. **CO**: Emission level of carbon monoxide.
3. **HC**: Emission level of hydrocarbons.
4. **NOx**: Emission level of nitrogen oxides.
5. **PMx**: Emission level of particulate matter.
6. **speed**: Speed of the vehicle.
7. **fuel**: Amount of fuel consumed by the vehicle.

# Data Pre-Processing

Pre-processing refers to the transformations applied to our data before feeding it to the algorithm. Data preprocessing is a technique that is used to convert the raw data into a clean data

We Collect data for UAVs from vehicular Ad-hoc networks.The parameters we use are time, id, x-coordinates, y-coordinates, angle, speed, POV, lane, slope, route, noise
etc.
Data Cleaning -> Collected raw data with noise and missing values goes through the process of identifying and correcting errors, filling missing values using interpolation
and imputation.
Normalization/Standardization -> The next step is to scale the data to a standard range.

# Standard Scaling and Normalization

StandardScaler is a preprocessing technique used for standardizing features by removing the mean and scaling to unit variance. It offers a simple yet effective way to standardize feature values.

Normalization Process

StandardScaler operates on the principle of normalization, transforming the distribution of each feature to have a mean of zero and a standard deviation of one. This process ensures that all features are on the same scale, preventing any single feature from dominating the learning process due to its larger magnitude.

All the data are standard scaled and Normalized

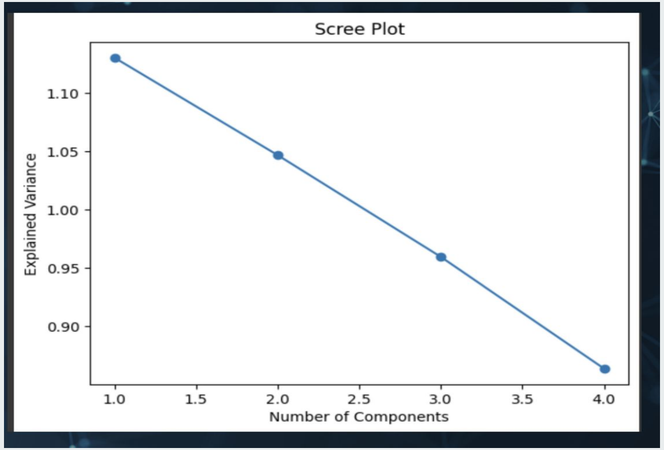
# Standard Scaling and Normalization

Min-Max Scaling (Normalization)
Rescales features to a specific range, typically between 0 and 1.
Formula: X_scaled = (X — X_min) / (X_max — X_min), where X is the original feature, X_scaled is the scaled feature, X_min is the minimum value of X, and X_max is the maximum value of X.

Standardization (Z-score Scaling)
Transforms features to have zero mean and unit variance.
Formula: X_scaled = (X — X_mean) / X_std, where X is the original feature, X_scaled is the scaled feature, X_mean is the mean of X, and X_std is the standard deviation of X.
Suitable for features with unknown or non-normal distributions.

# PCA(Principal Components analysis)

Principal component analysis (PCA) is a dimensionality reduction and machine learning method used to simplify a large data set into a smaller set while still maintaining significant patterns and trends.




Applied PCA to reduce the dimension of data

Screen plot : At component 3, 80%are preserved so we use screen plot.

# Clustering

Clustering is a way of organizing things or data into groups where items in the same group are more similar to each other compared to those in other groups.
We applied several clustering algorithm to our pre-processed Data.

# Evaluation:Silhouette Score,Davies-Bouldin Index

Silhouette Score is a tool for assessing the appropriateness of clustering results by providing a quantitative measure of how well-defined and distinct the clusters are. The Silhouette Score quantifies how well a data point fits into its assigned cluster and how distinct it is from other clusters. It measures the cohesion and separation of data points within clusters and helps determine whether the clusters are well-separated and internally homogeneous

The Davies-Bouldin Index is a validation metric that is used to evaluate clustering models. It is calculated as the average similarity measure of each cluster with the cluster most similar to it. In this context, similarity is defined as the ratio between inter-cluster and intra-cluster distances. As such, this index ranks well-separated clusters with less dispersion as having a better score.

# Silhouette Score

$$[s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}]$$

where:

- a(i) is the average distance between the data point i and all other points in the same cluster.
- b(i) is the minimum average distance between the data point i and points in a different cluster, minimized over clusters.

The overall silhouette score is the mean silhouette score for all data points:

$$[S = \frac{1}{N} \sum_{i=1}^{N} s(i)]$$

# Davies-Bouldin Index

The Davies-Bouldin Index (DBI) is defined as:

$$[\text{DBI} = \frac{1}{K}\sum_{i=1}^{K} R_i]$$

where Ri is the maximum similarity ratio between cluster i and any other cluster j:

$$[R_i = \max_{j \neq i}\left(\frac{S_i + S_j}{M_{ij}}\right)]$$

and:

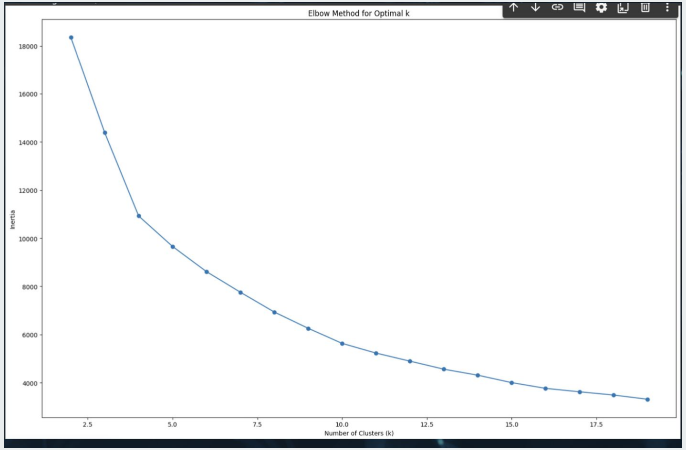
- Si is the average distance between each point in cluster i and the centroid of cluster i.
- Mij is the distance between the centroids of cluster i and cluster j.

$$[S_i = \frac{1}{|C_i|}\sum_{x \in C_i} \|x - \mu_i\|]$$

$$[M_{ij} = \|\mu_i - \mu_j\|]$$

# K-means Clustering

Kmeans algorithm is an iterative algorithm that tries to partition the dataset into the pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible.




Elbow method showing the optimum number of clusters with K-means

# K-means Clustering

### Assignment Step

Assign each data point xi to the nearest centroid μk:

$$C_i = \arg\min_k \|x_i - \mu_k\|^2$$

### Update Step

Recompute the centroid μk of each cluster Ck:

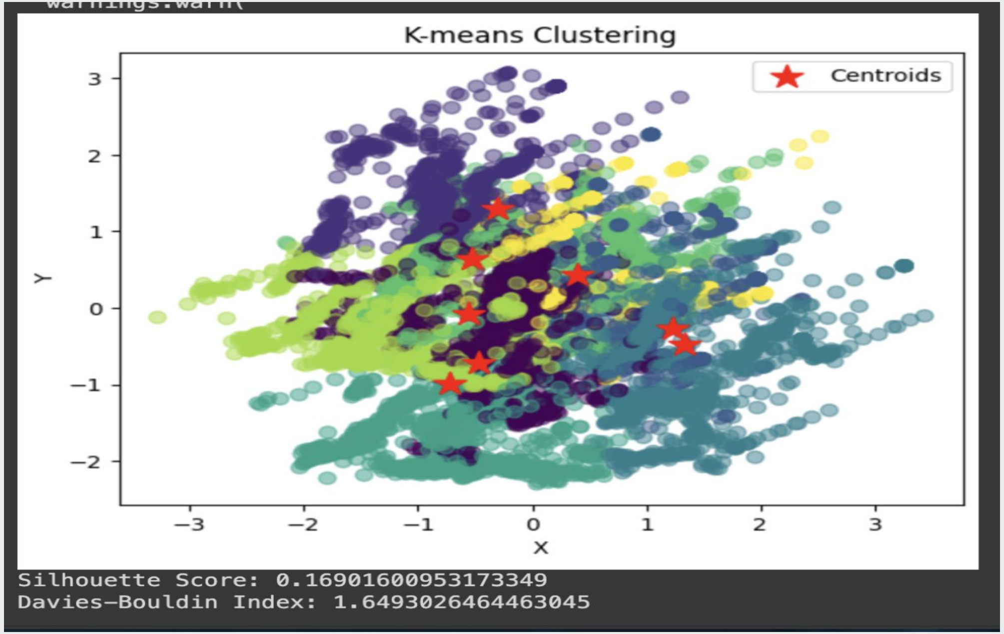
$$\mu_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i$$

### Objective Function

Minimize the within-cluster sum of squares (WCSS):

$$\text{WCSS} = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

# Evaluation


K-means Clustering
Centroids
Y
X
Silhouette Score: 0.16901600953173349
Davies-Bouldin Index: 1.6493026464463045

# Mean-shift Clustering

1. **Kernel Density Estimation**: The kernel density estimate at a point x is given by:

$$[f(x) = \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)]$$

where:

   - K is the kernel function (e.g., Gaussian kernel).
   - h is the bandwidth parameter.
   - xi are the data points.

2. **Mean Shift Vector**: The mean shift vector m(x) is calculated as:

$$[m(x) = \frac{\sum_{i=1}^{n} x_i K\left(\frac{x-x_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right)} - x]$$

This vector points towards the direction of the maximum increase in the density.

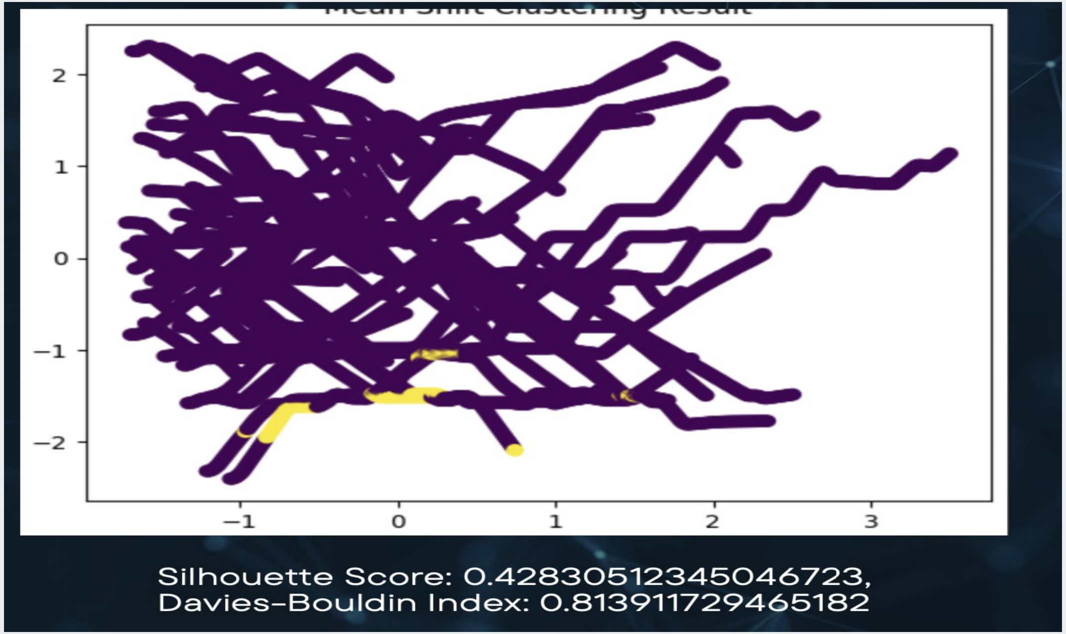
3. **Update Rule**: The data points are iteratively shifted by the mean shift vector:

$$[x \leftarrow x + m(x)]$$

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

UBFC
UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ

# Mean-shift clustering



# DBScan clustering Algorithm

% Core Points

A point pp is a core point if:

$$[|\{q \in D \mid \|p - q\| \leq \epsilon\}| \geq \text{minPts}]$$

% Directly Density-Reachable

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

$$[\|p - q\| \leq \epsilon \quad \text{and} \quad |\{q \in D \mid \|p - q\| \leq \epsilon\}| \geq \text{minPts}]$$

% Density-Reachable

A point q is density-reachable from p if there exists a chain of points p1,p2,...,pn such that:

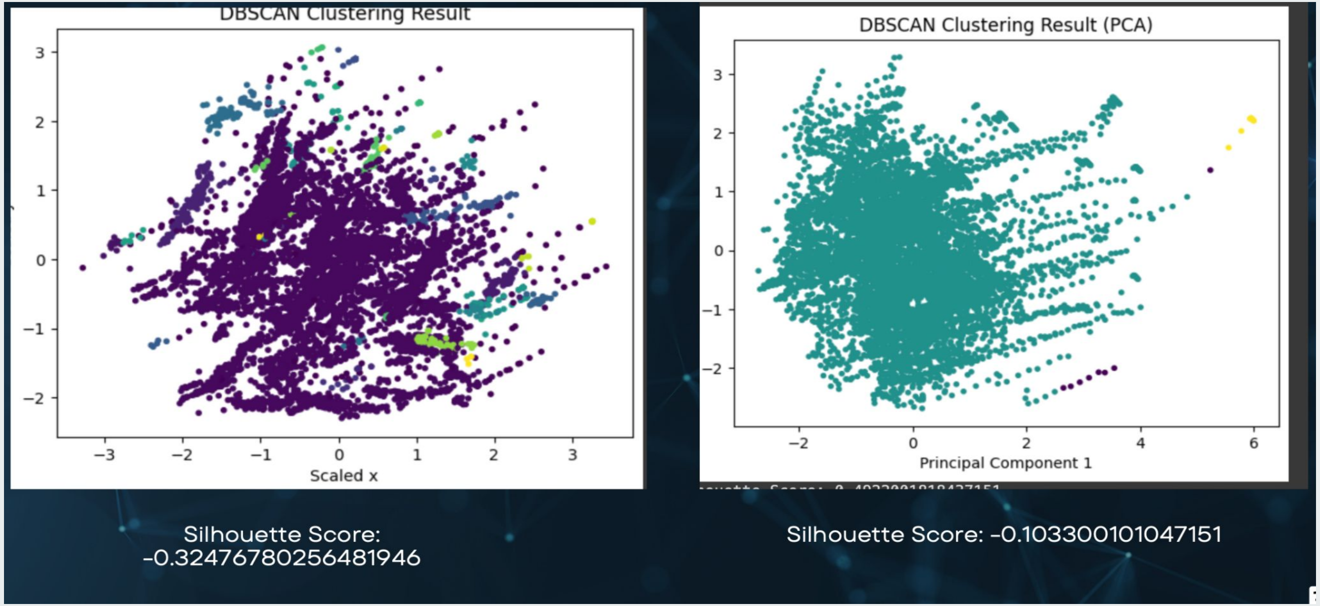
$$[\text{Density-Reachable: } \exists\, p_1, p_2, \ldots, p_n \text{ such that } p_1 = p, p_n = q, \forall i \in \{1, \ldots, n-1\}, p_{i+1} \text{ is directly density-reachable from } p_i]$$

% Density-Connected

Two points p and q are density-connected if there exists a point o such that:

$$[\text{Density-Connected: } \exists\, o \text{ such that } p \text{ and } q \text{ are density-reachable from } o]$$

# DBScan clustering Algorithm



# Gaussian Mixture Model

Gaussian Component: Each Gaussian component is defined by its mean μk and covariance matrix Σk.

Probability Density Function: $$[\mathcal{N}(x \mid \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)\right)]$$

Mixture Model: The mixture model is the weighted sum of K Gaussian Component: $$[p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k)]$$

where πk are the mixture weights, and $[\sum_{k=1}^{N} \pi_k = 1]$

Expectation maximization (EM) Algorithm:

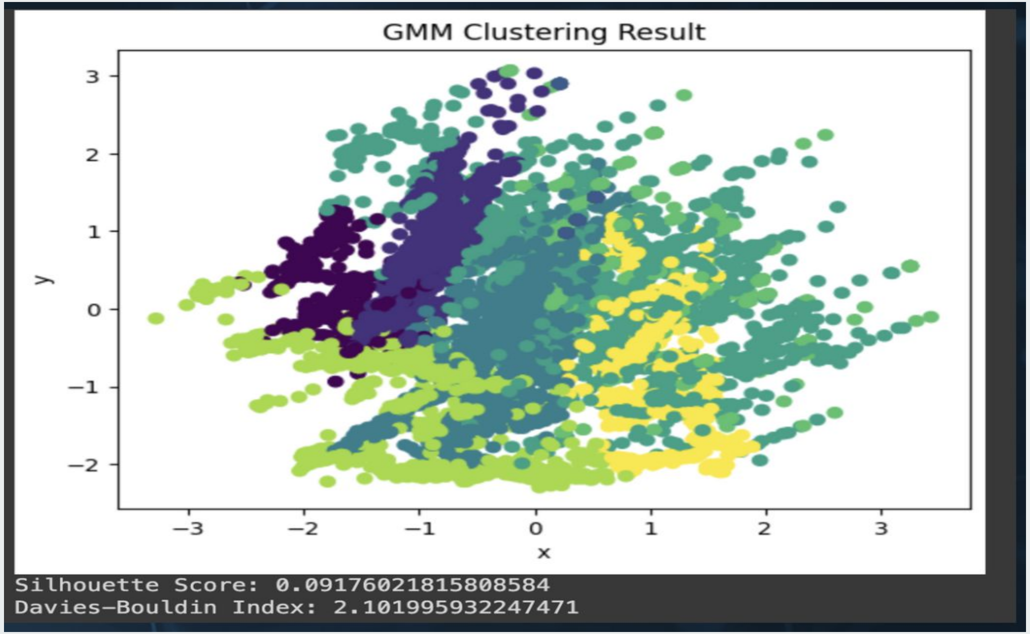
E-step: Calculate the responsibility ɣ(zik) that component k takes for data point xi: $$[\gamma(z_{ik}) = \frac{\pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}]$$

M-step: Update the parameters $\pi$k, $\mu$k and Σk using the responsibilities: $$[\pi_k = \frac{N_k}{N}]$$

$$[\Sigma_k = \frac{1}{N_k}\sum_{i=1}^{N} \gamma(z_{ik})(x_i-\mu_k)(x_i-\mu_k)^T]$$

Where

# Gaussian Mixture Model



# The BIRCH Clustering

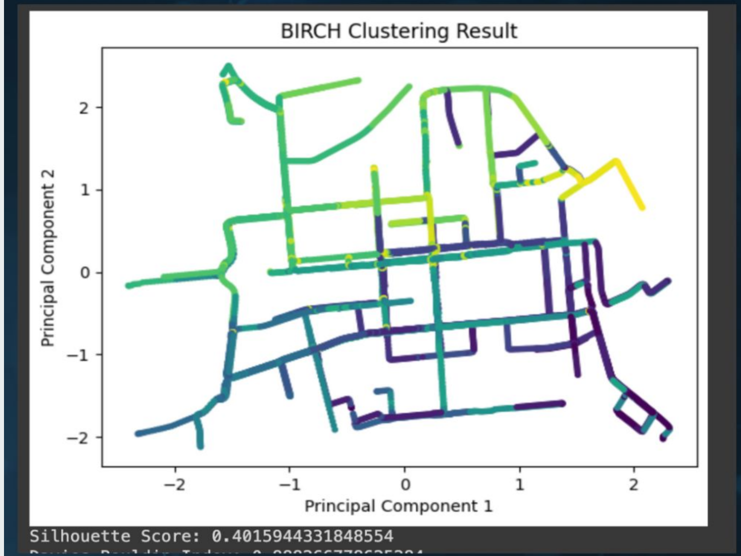
$$[\mathrm{CF} = (N, LS, SS)]$$

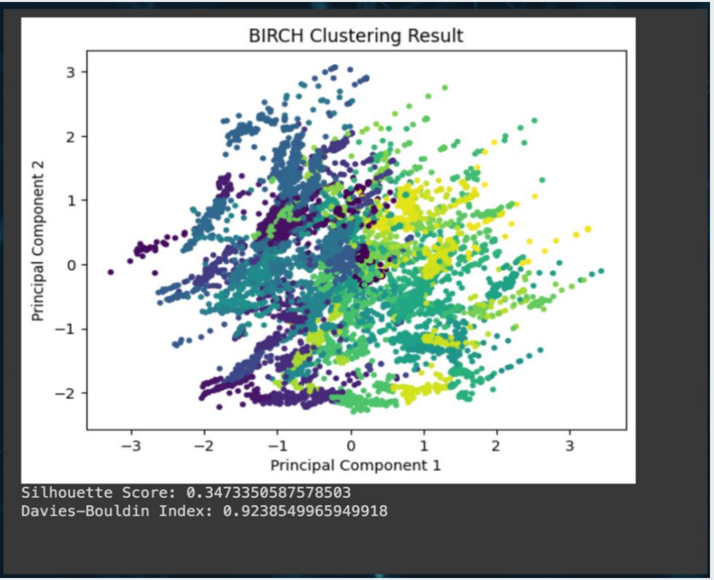
N is the number of points in the  cluster.
LS is the linear sum of all points in the cluster.
SS is the squared sum of all pints in the cluster.

$$[\text{CF Tree} = \text{Hierarchical tree structure of CFs}]$$

$$[B\text{: Maximum number of entries in a node}][T\text{: Maximum diameter threshold within a node}]$$

# The BIRCH Clustering



# Comparison

| K-means | Agglomerative Clustering | Mean-Shift Clustering | DBSCAN | GMM | OPTICS | BIRCH | Result |
|---|---|---|---|---|---|---|---|
| 0.169 | 0.122 | 0.428 | -0.3247 | 0.0917 | | 0.4015 | Silhouette Score |
| 1.649 | 2.619 | 0.8139 | -0.1033 | 2.1019 | | 0.9238 | Davies-Bouldin Index |
| | | | | | | | |

# Federated learning

Federated learning is a method of training machine learning models where instead of gathering all data in one place, the learning process happens across multiple decentralized devices or servers.
Here we process the data using the 4 different edge computing methods.

$$[\theta_{\text{new}} = \theta - \eta \sum_i \frac{|\mathcal{D}_i|}{|\mathcal{D}|} \nabla_\theta \mathcal{L}_i(\theta)]$$

$$[\hat{g} = \frac{1}{n} \sum_{i=1}^{n} (g_i \text{noise}_i)]$$

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

UBFC
UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ

# Results for Federated Learning

We applied the regression time series data within  the extracted data and to predict the value of fuel consumption and we go the following result.

Algorithm used XGBoost

Results

RMSE of federated models:

Model 1: 31.45577266001291

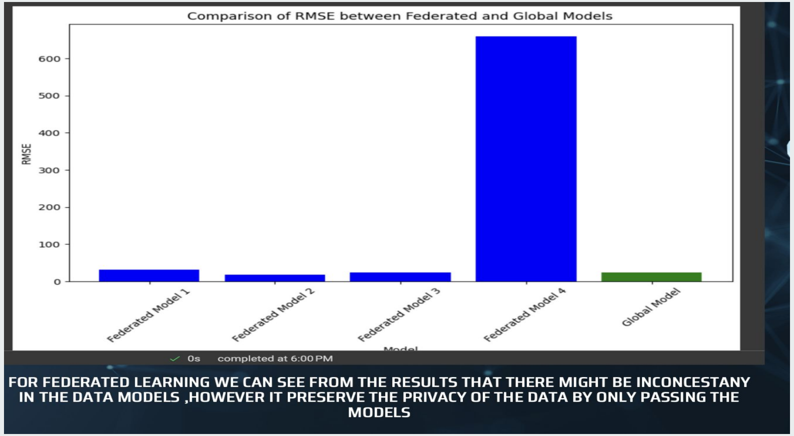
Model 2: 18.244440219391787

Model 3: 23.47979683879943

Model 4: 659.8860493294914

Average RMSE of federated models: 183.26651476192387

RMSE of the global model: 23.46016774998053



# Future Goals

For future : we cannot use sumo for adding uavs as a non stationary node, so for future work we can use unity and other simulation software and add messaging/communication protocol within uavs and vehicles and see the data transfer rate and it's speed.and with this collected data we can use the concept of reinforcement learning to create effective communication network between UAVs and vehicular adhoc network.

# Conclusion

For federated learning, we can observe from the results that there might be inconsistency in the data models. However, it preserves the privacy of the data by only passing the models. And for the clustering of the data the data shown by Birch is seems to be more effective.