Tutor project
UFR STGI - Master1 IoT

Professor: Philippe Canalda
Students: Amina, Azra, Nadina and Nerma

Eco-friendly Carpool Optimization Using Clustering and Travel Data - Developer's Guide

The carpooling algorithm was implemented in Google Colab, Eco-friendly Carpool Optimization Using Clustering and Travel Data.ipynb. In order to run the code, an access to Google Colab service is required, or a locally available environment with at least Python 3.8+ installed.

Generating datasets

In order for the algorithm to compile, three inputs in the form of a csv are required: a csv containing information about missions, a csv containing data about passengers, and one with CO2 emissions for various vehicles. The missions csv contains following columns:

- mission_id (int64)
- person_id (int64)
- start_city (object)
- end_city (object)
- start_date (object)
- end_date (object)
- event (object)
- real_move (bool)
- travel_type (object)
- vehicle_type (object)
- is_return_trip (bool)
- km (int64)
- parking_cost (int64)
- hotel_cost (int64)
- plane_cost (int64)
- reimbursement (int64)
- total_cost (int64)
- nearest_airport (object).

The person csv contains:

- person_id (int64)
- first_name (object)

- last_name (object)
- home_address (object)
- admin_address (object)
- admin_city (object)
- has_car (bool)
- car_type (object)
- car_model (object)
- car_capacity (float64)
- fiscal_hp (float64).

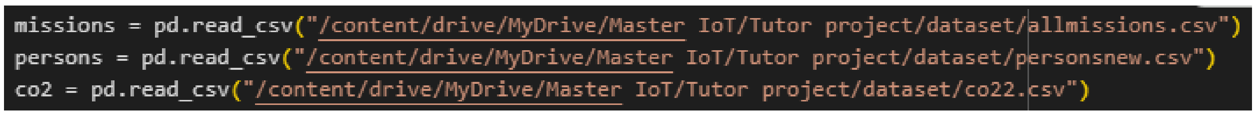
For the CO2 csv, the following columns are required:

- car_model (object)
- car_type (object)
- co2_per_km (float64)
- energy_per_km (float64).

These csv files can be generated using the Simulating data.ipynb file, where parameters such as the amount of data, content of different columns, output file paths etc., can be adjusted as needed, according to the detailed instructions in the file itself.

Running the code

Once all the data is in the correct format, the user needs to connect to the runtime in Google Colab, and then mount Google Drive, or upload the necessary csv files to the content folder in Google Colab. Afterwards, the correct path to the csv files needs to be specified in the cells containing the following lines of code:



```
missions = pd.read_csv("/content/drive/MyDrive/Master IoT/Tutor project/dataset/allmissions.csv")
persons = pd.read_csv("/content/drive/MyDrive/Master IoT/Tutor project/dataset/personsnew.csv")
co2 = pd.read_csv("/content/drive/MyDrive/Master IoT/Tutor project/dataset/co22.csv")
```

Once this step is completed, all the cells need to be run in order they appear in, either by clicking on the play icon in each cell, or choosing Runtime -> Run all from the toolbar at the top. After a while, the algorithms will produce the results which can be plotted by running the following cells and the results interpreted. It is worth noting that the algorithms were trained on datasets with 300 instances. In case of a slow execution for larger datasets, runtime may need to be upgraded to GPU or TPU, or additional computing units bought through Google Colab services.

Environment setup

In order for the algorithms to run properly, the required Python libraries must be included, via the following imports:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.metrics import silhouette_score
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans, AgglomerativeClustering
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.cluster.hierarchy as sch
import warnings
import os
from datetime import datetime
```

In Google Colab, most libraries are preinstalled. Locally, you can install dependencies via pip install command, like in the example below:

```
pip install gower
```

Preprocessing of data

The code first preprocesses data by encoding relevant categories using OneHotEncoding and scaling numerical features using StandardScaler.

Clustering algorithms

Once this step is completed, clustering algorithms, KMeans and Agnes are applied, using the Gower distance matrix to handle mixed data effectively. Clustering parameters, as well as the number of features, were adjusted accordingly so that the best results are achieved. This was decided after evaluation and visualization, where Silhouette scores were calculated to assess cluster quality, Principal Component Analysis (PCA) used for dimensionality reduction and to visualize clusters, and intermediate results such as cluster assignments, group sizes, and cost/emission metrics output for inspection.

Optimization phase

Finally, the code enters the optimization phase, where carpool groups are formed by matching users with similar start/end cities, close travel dates, and available car capacity.
Additionally, car trips are replaced with train trips when train travel is at least 2 hours faster, in order to further reduce CO2 emissions. Finally, updated cost and CO2 emissions are recalculated accordingly. For more details on what each cell does, please consult the detailed explanations in Google Colab notebook.