Problem description:

You are tasked with categorizing shopping trip types based on the items that customers purchased. To give a few hypothetical examples of trip types: a customer may make a small daily dinner trip, a weekly large grocery trip, a trip to buy gifts for an upcoming holiday, or a seasonal trip to buy clothes. The trips contained in this data are categorized into 38 distinct types. You are challenged to recreate this categorization/clustering. The training set (**train2026.csv**) contains a large number of customer visits with the TripType included. Each visit may only have one TripType. You will not be provided with more information than what is given in the data (e.g. what the TripTypes represent or more product information).

The objective is to predict the TripType for each customer visit.

Data fields description:
- TripType: a categorical id representing the type of shopping trip the customer made. This is the ground truth that you are predicting. TripType ‘999’ is an "other" category;
- VisitNumber: an id corresponding to a single trip by a single customer;
- Weekday: the weekday of the trip;
- Upc: the UPC number of the product purchased;
- ScanCount: the number of the given item that was purchased. A negative value indicates a product return;
- DepartmentDescription: a high-level description of the item's department;
- FinelineNumber: a more refined category for each of the products.

Notes regarding the deep dive interview:
- In the next stage of the recruitment process there will be a technical interview where we will do a deep dive into this challenge.
- The main focus of this challenge is not for you to obtain the best solution, but for us to measure how you perform globally when faced with it. Therefore, you should focus on demonstrating knowledge in data preparation, data pre-processing, Machine Learning algorithms, hyperparameter optimization, evaluation metrics, etc.
- The short time frame you are given for this challenge is designed to force you to make decisions, as there is not enough time to explore a large number of paths. Therefore, we recommend you first concentrate on a simple end-to-end solution before exploring more complex approaches that would improve the initial solution.
- All code must be written in Python;
- All code should be properly written: variable/functions names should be clear and self-explanatory, comments should be used to explain the purpose of functions, code blocks or not-so-obvious commands, etc.;
- Please double check if all submitted files are readable (ex: they were not corrupted due to zipping or conversion to PDF format);

Files to be submitted by e-mail before the deadline:
- A zipped file containing all scripts/notebooks with the implemented code:
  - The minimum requirements are a main file and a text file describing how to execute it;
  - If special or not-so-common libraries are used, please state which ones (including version) and how they can be installed (preferably, they should be installed automatically);
- A text-style (ex: .docx → .pdf) or presentation-style (ex: .pptx → .pdf) report in PDF format describing the thought process, approach to the problem, experimental process, obtained results, ideas for further improvements (namely, what you wished you could have done further to improve

the results but did not have the time to implement), etc.. This document will be the basis for a follow-up interview to evaluate your performance in this challenge. You will be expected to do a 20-minute presentation.

Good luck!

PS: Contact us by e-mail if you need clarifications regarding any aspect of the challenge itself, the evaluation process, or the deep dive interview.