New York City (NYC) Taxi and Limousine Commission (TLC) Trip Record Data

Big Data Project 2024/25

General instructions

Your objective is to perform a comprehensive data analysis on a moderately sized dataset (relative to big data standards). You should adhere to the CRISP-DM methodology [1] and structure your report accordingly. **You must work in teams of two students**; exceptions only by permission.

Tools

Choose appropriate tools from the Python data science ecosystem, e. g. Dask, Pandas, Numpy, Matplotlib, Streamz, Parquet, DuckDB, Kafka, … Your experiments must be performed and evaluated in Arnes HPC cluster and implemented in a proper distributed manner (Dask+SLURM), whenever possible.

Data

Your primary data source is the full New York City TLC Trip Record Data [2]. You will augment it with additional information from the data sources of your choice (as per instructions below). TLC trip record data consist of four principal datasets (Yellow Taxi, Green Taxi, FHV, FHVHV):

- **Yellow Taxi:** This dataset includes trip records from traditional yellow medallion taxis, which primarily serve Manhattan and the airports. These taxis are unique in that they can pick up passengers via street hails anywhere in the city, and the data includes detailed fare breakdowns. Our focus will be on data from 2012-01 onwards.
- **Green Taxi:** This dataset covers trips made by green "boro taxis" (Street Hail Livery), which are authorized to pick up passengers via street hail mainly in the outer boroughs (outside Manhattan's core) and Upper Manhattan. Like yellow taxis, they can also do pre-arranged trips citywide and record detailed fare information. Our focus will be on data from 2014-01 onwards.
- **For-Hire Vehicles (FHV):** This dataset contains records for trips from traditional For-Hire Vehicle services like community liveries, black cars, and limousines, which must be pre-arranged through a licensed base. It generally excludes trips dispatched by the major app-based, high-volume services and contains less detailed fare information than taxi data. Our focus will be on data from 2015-01 onwards.
- **High Volume For-Hire Vehicles (FHVHV):** This dataset tracks trips dispatched by licensed high-volume services, predominantly the major app-based companies like Uber and Lyft. All trips are pre-arranged through the company's platform, and the data captures pickup/dropoff times and location zones but not itemized fare details. Our focus will be on data from 2019-02 onwards.

You can download the parquet data files from the source [2], or from the Arnes cluster [3] (directory `/d/hpc/projects/FRI/bigdata/data/Taxi`). Metadata, data dictionaries and additional relevant data can also be found from [2].

Tasks

Your project consists of the following tasks, primarily involving big data processing, augmentation, and analytics (see [6] for an illustrative example):

T1. Read the original parquet files (for each dataset) and repartition them by year (you will need to introduce a new attribute, based on the pickup time). Ensure the row groups of 100-200 MB (approx. 2M rows). Take special care of schemas and attribute names, as they may slightly differ between files. You may approach the problem using either pyarrow parquet datasets or Dask dataframes. I suggest the first approach, as it is more flexible and less memory consuming. All processing in this task can and should be performed in the same step.

T2. Data quality issues with respect to pickup datetimes. After completing T1 you will notice some unexpected partitions (eg. years before 2010 or after 2026) which indicate problems in timestamps. Looking at the **original parquet files**, some possible problems include:

- Year outside the current year (but in some cases this is OK, eg. 31. 12., or 1. 1., due to delayed data entry).
- Pickup and dropoff timestamps are the same.
- Dropoff timestamp is smaller (earlier) than the related pickup timestamp.
- Trip distance equals to zero.
- … (suggest some more, also related to other attributes)

Process the original parquet files for all four datasets in a distributed map-reduce pattern (parallelize operations on a list of filenames using Dask). Analyze the problems and present the results aggregated by year with a table and an appropriate chart; one table and one chart for each of the four datasets. You shall also use the data quality insights in later data processing steps.

T3. Focus on a moderately sized year-partition, such as 2024 for Green Taxi. The parquet size for this partition is approximately 16 MB. Export this partition to CSV format (original and gzipped), HDF5 table, and DuckDB database file. Compare the size and speed of reading a Pandas dataframe from each.

T4. Programmatically perform an introductory exploratory data analysis for all four datasets. Propose and calculate appropriate data aggregates (consider also temporal and spatial aggregations, analysis, and visualization). Also, explore if some kind of similarity can be detected between pairs of datasets. Use the usual Python data science tools, combined with big data tools:

- Dask and/or Dask-SQL,
- DuckDB

When using DuckDB on Parquet files, balance between SQL and Python/Pandas processing (use DuckDB for aggregation and Pandas for final processing, when necessary). **Note:** finalized tasks on full-size datasets should be performed in the Arnes cluster.

T5. Augment the original TLC locations with additional information (as much as possible):

- Weather information
- Vicinity/locations of primary and high schools
- Information about events in vicinity
- Vicinity/locations of major businesses
- Vicinity/locations of major attractions

You can find many (but not all) sources in the New York City open data repository [9]. You will need to link the data with respect to pickup/dropoff location and time (where applicable, e. g. for weather and event data). For locations, you will need to combine Taxi Zone lookup table and Taxi Zone shapefile.

T6. Perform the data analysis in a "streaming" manner (treat datasets as streams). Use Kafka as a message broker and write a Python custom producer (reading data from processed Parquet datasets, as created in T1, ordered by pickup timestamp - important for producing Kafka messages) and a Python custom consumer (for processing data). Combine FHVHV and Yellow Taxi data sources for one calendar year (any year after 2020). Document if and how using different topics can help you. Show rolling descriptive statistics (mean, standard deviation, …, for at least three different attributes) for boroughs, and for the 10 most interesting locations (with the highest numbers of pickups and dropoffs overall, or by your qualified choice. For the same data, choose, implement, and apply a stream clustering algorithm of your choice [7]. Optionally, consider also using ksqlDB [8] for stream processing.

Note: stream processing is better performed outside the Arnes HPC cluster.

T7. Perform advanced data analysis in a "batch" manner using machine learning. You may need to appropriately transform your augmented data. Propose, justify an execute at least one relevant machine learning problem in a properly distributed manner in the Arnes HPC cluster, evaluate the quality of the results, their usefulness, and the impact of your data augmentation. Some suggestions for machine learning problems (feel free to propose your own):

a) Identifying Gaps in On-Demand Service Coverage. Ensuring equitable access to transportation is a key goal for urban planners. Identify geographic areas and/or time periods with low availability of both Taxi and FHV/FHVHV services to pinpoint potential "mobility deserts" where residents may lack adequate on-demand options.

b) Assessing Differential Response to External Events. Model and analyze on-demand transportation system reactions to external shocks—such as major public transit failures, large-scale public gatherings, severe weather events, or health crises (e.g. COVID-19 to assess resilience and the adaptability of each transport mode.

c) City-Wide Anomaly and Event Detection. Establish a system to automatically detect unusual patterns across the combined Taxi and FHV/FHVHV network to identify data quality issues, potentially fraudulent activities (e.g., meter tampering, artificially inflated trips), and detecting the impact of significant real-world events (e.g., accidents, road closures, public transport disruptions) on mobility.

d) Characterizing and Detecting Differential Service Usage Patterns. Understanding whether Taxis and FHVHVs primarily compete for the same trips or cater to distinct market segments and travel needs is fundamental to comprehending the overall mobility ecosystem. Do they differ significantly in typical trip lengths, durations, or purposes served, and can they be predicted within useful error margins?

e) City-Wide Demand Forecasting. Accurate forecasting of total on-demand ride demand across the entire city is essential for proactive resource allocation, infrastructure planning, and policy development that considers the full spectrum of services. This moves beyond optimizing within a single service type to understanding system-wide needs, so all four datasets should be used. Compare a single model vs. aggregation of four separate models (FHV, FHVHV, Green, Yellow).

Ensure that each single worker **will not have** enough memory to store and process the entire dataset and evaluate cluster scalability by gradually increasing the number of workers. Use at least two of the three kinds of distributed/out-of-core machine learning algorithms:

- One of the existing distributed algorithms from Dask-ML
- A sophisticated third-party algorithm which "natively" supports distributed computing (such as XGBoost or LightGBM)
- One of the more common scikit-learn algorithms utilizing partial_fit.

For all scenarios compare their performance in terms of loss (error), scalability, time, and total memory consumption.
Note: scalability should be tested in the Arnes HPC cluster by increasing the number of workers and observing the total processing time.

T8. Calculate and visualize how the emergence of FHVHV operators (such as Uber or Lyft) influences green and yellow taxi services (as well as smaller FHVHV competition, such as Juno or Via), both in absolute and relative terms (with respect to all trips).

T9. (optional) Visualize your results from T2 and T4 on a map (Google Maps, Open Street Maps, or a map of your choice); see for example prettymaps [4, 5].

Reporting and grading

You are required to submit a report with corresponding code, present your findings, and defend your project work.

- Deliverables:
  - PDF Report: Prepare a comprehensive report (up to 10 pages), formatted as a scientific paper.
  - Code Files: Submit all corresponding code files, packaged together in a single ZIP archive, or as a GitHub repository.
- Content Requirements:
  - Visualizations: Include relevant and insightful visualizations as needed to support your analysis and findings.
  - References: Find and include accurate and appropriately formatted references for all sources used.
  - Data Usage Demonstration: You must clearly demonstrate, both within the report and in the code comments/documentation, that all relevant data has been utilized for processing.
- Presentation, code review, grading:
  - Each team will present their preliminary results in a brief 5-minute presentation at the end of the semester (end of May).
  - Code reviews will be organized in May/June.
  - All materials should be submitted by the June 15th
  - To achieve the minimum passing grade, tasks T1 through T4 must be completed correctly and fully. It is strongly recommended that you also complete at least one additional task beyond T4 to ensure you meet the requirements.

References

[1] CRISP-DM, http://lyle.smu.edu/~mhd/8331f03/crisp.pdf

[2] New York City TLC Trip Record Data, https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[3] Data folder on Arnes HPC cluster: /d/hpc/projects/FRI/bigdata/data/Taxi

[4] Marcelo de Oliveira Rosa Prates, Prettymaps, https://github.com/marceloprates/prettymaps

[5] Prettymaps, New York example, https://www.reddit.com/r/prettymaps_/comments/qpl31e/code_for_local_use_new_york_example/

[6] Big Data Analytics of Taxi Operations in New York City, https://www.scirp.org/journal/paperinformation?paperid=94087

[7] Data stream clustering, https://en.wikipedia.org/wiki/Data_stream_clustering

[8] Confluent ksqlDB, https://www.confluent.io/product/ksqldb

[9] NYC Open Data, https://opendata.cityofnewyork.us/