Found datasets

The datasets explained below can be found on the spreadsheet: https://docs.google.com/spreadsheets/d/1VOfFNOAoOclL5dduwW4R5ojYEokGohtg0ih_eCW2HpA/edit?usp=sharing

Fuel Economy:

- This dataset is provided by the office of energy efficiency of the United States.
- The dataset provides access to fuel economy information for 1984-current model year vehicles in either JSON or XML format.
- There are several resources available, including those for returning specific vehicle records, lists of emission records, and fuel prices.
- The data can be downloaded in CSV and XML formats for both vehicles and emissions.
- The EPA has revised MPG estimates for several Audi, Bentley, Porsche, Volkswagen, MINI Cooper, Cooper S, Mercedes C300 4matic, and Ford vehicles.
- The resources at /ws/rest/vehicle/menu/* are used to select a vehicle using a series of four menus: year, make, model, and options.
- The dataset also provides access to My MPG shared data via vehicle make and model.
- The data description includes information on alternative fuel or advanced technology vehicles, annual petroleum consumption, charging times, MPG, CO2 emissions, engine cylinders and displacement, drive axle type, and more.

TruCars:

- This dataset was scraped from Trucars, which is a website that allows users to browse and purchase new and used cars.
- The dataset includes information about different car models, including their make, model, year, price, MSRP, seller location, body class, exterior and interior colors, engine, fuel type, MPG, drive type, transmission, listed date, popular features, standard features, and images.
- The prices listed for each car are in US dollars.
- The dataset includes information on the cars' features, such as premium wheels, moonroof, navigation, heated front seats, adaptive cruise control, blind spot system, and backup camera.
- The dataset also includes information on the cars' standard features, such as wireless charging, driver lumbar support, and parking sensors.
- The dataset includes links to images of each car, which can be used to view the car's exterior and interior.

Kavak:

- Kavak is a Mexican business that sells second-hand cars online.
- The company was founded in 2016 and has since expanded to several Latin American countries, including Mexico, Argentina, and Brazil.

- The scraped database contains information about used cars that are being sold by Kavak.
- The data includes fields such as make, model, year, price, seller location, body class, exterior color, interior color, engine, fuel type, drive type, transmission, listed date, user km, images, and more.
- There are a total of 24 fields in the database.
- The dataset includes cars from different makes such as Nissan, Toyota, BMW, and Mini Cooper.
- The dataset also includes cars from different years, ranging from 2008 to 2021.
- The database includes cars with varying prices, from a few thousand dollars to over $100,000.
- The dataset also provides information about the car's features such as airbags, fuel consumption, number of doors, air conditioning, ABS, and more.
- The data is structured in a tabular format and can be used for various analytical purposes, such as data visualization, machine learning, and predictive modeling.

Process of data scraping

For the scraping process we searched for different sites with car listings and reviewed the type of data they had. For example True cars and Kavak were our top choices as they had a lot of cars on sale, but in the end we decided Kavak was the superior choice as its data was from México. We then thought of a two step process to scrape all of the cars data from Kavak.

First we obtained all of the car urls from the site by paginating in the search page. For this we used golang, using get requests and an html parser called go query to extract all links. Once we had all urls we visited each of them and extracted the relevant data into a JSON file, by the end we managed to scrape around 11k car listings from México.

Script to scrape the data:
https://github.com/IvanDLar/MOVU-BackEnd/tree/main/scripts/scrapper

Data processing

After scraping the data, we had to do a processing process to adequate it to our data model.

Since every car was stored as a json file, it was very easy to process all the data, and store the information we wanted. We used a google colab notebook to process all the files as fast and possible and used python to map all of the json files. Since the processing was done to store data on elastic for the search, we needed to define which variables were the most important ones from what was scrapped.

At the end we decided to store the following variables: Make, model, variant_name, full_name, type, year, used_km, fuel_type, transmission, passengers, price, geo_loc, region, colors, images, features and a description (which was made taking into account all of the previous values).

Script that processes the data:
https://colab.research.google.com/drive/1mIEzksvW66nOiS-WdDx8wfTgUjbcHj7e?usp=sharing

Uploading the data to Elastic

To upload data to ElasticSearch, it is necessary to use the ElasticSearch library to create an index and perform a bulk query using the Elastic API. This process requires the data to be in a specific format called .jsonl, which stands for JSON Lines. The .jsonl format is a JSON-encoded file where each line represents a single JSON object. It is commonly used for processing large amounts of data in a streaming fashion, where each JSON object is a separate record.

To accomplish this, we created a Python script that utilizes the ElasticSearch library to create the necessary index and bulk query. The index is a collection of documents that share a similar structure, and the bulk query is used to add or update multiple documents at once. The bulk query is faster and more efficient than adding documents one by one.

However, our data was not initially in the correct .jsonl format, so we had to reformat it to upload it correctly. This required additional processing steps to ensure that the data was in the correct format before uploading it to ElasticSearch.

The Python script and the .jsonl file containing the processed data can be found under our backend repository. The script and file serve as valuable resources for future updates to the data or for anyone interested in exploring the data further.

Link to the script and data: https://github.com/IvanDLar/MOVU-BackEnd/tree/main/scripts/opensearch

Conclusion

For our use case , we decided that the database scraped from Kavak is the best option. This is due several reasons:

- One of the main reasons why we chose Kavak is because it provides information about cars in Mexico, which is where MOVU will be launched.
- Another advantage of the Kavak dataset is that it is in Spanish, which is highly preferred for our target audience.
- We also found that the Kavak dataset had a good fit with our data model, requiring less processing than the other datasets.
- In contrast, the Fuel Economy dataset had a lot of fields that were not relevant for our use cases, and the TruCars dataset did not provide location information for the cars.
- It's important to note that we had to do some processing on the Kavak dataset as well, but it was not nearly as extensive as with the other datasets.
- Overall, we believe that the Kavak dataset will provide us with the most relevant and useful information for our application, and we are confident that it will help us deliver a better user experience to our customers.

- As a final step, we generated a test dataset based on the Kavak data, which we will use to test and refine our algorithms and features before launching the application.