Used Car Characteristics

An Explanatory Data Analysis

Dávid Fodor
Faisal Zayeem Ahmed
Márió Palágyi

Submission date: TBD

Datascience Capstone Project - IMC Krems

Abstract

A brief overview of the project, including problem statement, methodology, key findings, and conclusions.

1 Introduction

Problem Statement: Describe the problem.

Objectives: List the objectives and tasks required to be completed.

Data: Describe your dataset.

Background Information: Provide context and challenges based on your understanding of the problem.

2 Literature Review

Summary of relevant research and related works in the field. This section does not need to be exhaustive; include background knowledge on how the problem has been addressed before and how you plan to solve it.

Luckily, there is plenty of literature on the topic of used car prices, particularly the prediction car prices seems to be a popular topic. Searching on Google Scholar for "used car price prediction" yields almost a million results.

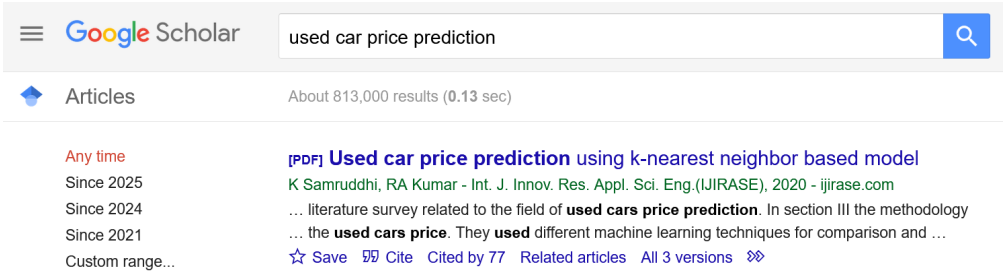

Figure 1: Screenshot of Google Scholar Search

However, looking for papers that solely try and explore the data and find interesting patterns is lot more difficult. Mostly, the exploration of all the data is done in the context of a prediction model, which is why we also aim to create a usable prediction model as part of our assignment.

Here, as the literature review is a small portion of the report, I will just mention some of the interesting conclusions we have found from others. Firstly, there is seemingly no

real consensus on what the best model is for predicting car prices. Some papers like [1] for example...

3 Methodology

- **Data Collection, Description and Management:** Describe the steps involved in data collection or preprocessing, including EDA.
- **Analysis Techniques:** Describe your approach to solving the problem. You can include a model diagram to illustrate your method.
- **Tools and Software Used:** List the tools, frameworks, and software used for the project.

3.1 Data Collection

In our case the data collection is a rather interesting part of the project, which took quite some time. Unlike other groups, we had to find our own primary datasource, seeking a used car marketplace which we could scrape. After several trials of scraping different websites, we finally ended up with one that was quite lenient with their robots.txt file, and we could scrape it without any issues. The website we used was `https://www.autoscout24.com/`, which is one of the largest used car marketplaces in Europe boasting more than 2 million listings.

The scraping took place from to , running practically 24/7 on a laptop even though parallelization was used. In the end we have collected *505804* rows of data, which is a lot more than we could have hoped for in, and wrote it to a **parquet** file. The scraping consisted of two main parts, which we will discuss in the following subsections along with the bigger challanges we have faced.

3.1.1 Scraping Car Listing

Due to the nature of the website, we needed user interaction for scraping the car listing, so we used **Selenium** to scrape the URLs pointing to specific car listings. This was achieved through a headless Chromium browser. We faced two hurdles in this change which we had to overcome with simulated interactions.

- **Pagination:** The website uses a classic pagination system on page, meaning we needed to be able to navigate through each page.

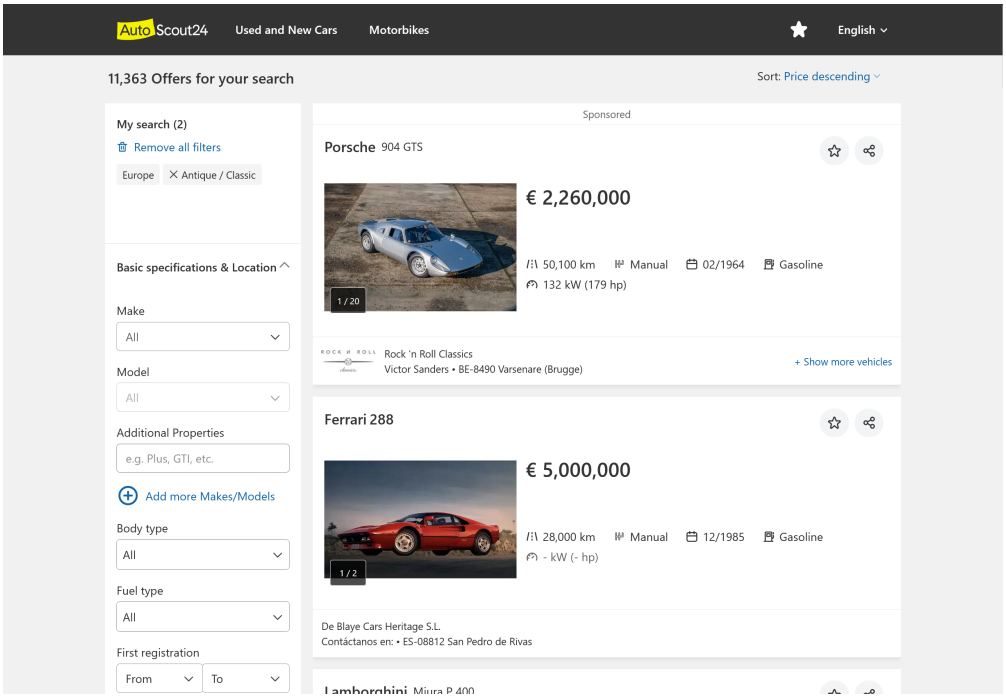

Figure 2: Screenshot of Autoscout24 Homepage

- **Max 20 pages per search:** The website limits the number of returned pages to *20*, so we had to find a way to methodically filter search parameters to find as many sets of *20* pages as possible.

In theory almost every common feature of cars can be filtered on the page, so with enough patience one may find all 2 million listings. However, we did not have so much time, nor so many cores on our hands, so we tried finding a middle ground, which was the following:

1. Loop through each car make available
2. For each car make, loop through a range of dynamically changing power ranges in kw based on their presumed likelihood e.g. `0-15`, `15-20`, `20-25`, `25-26`, `26`, `27`...
3. For each power range of the car make, loop through the pages available of the pagination
4. On each pagination page, collect the URLs of the car listings
5. Store the URLs in a JSON file, keyed by the make

This constituted most of our time scraping, as the user interaction needed to be waited upon and there were simply a lot of possible permutations of make, power, pagination

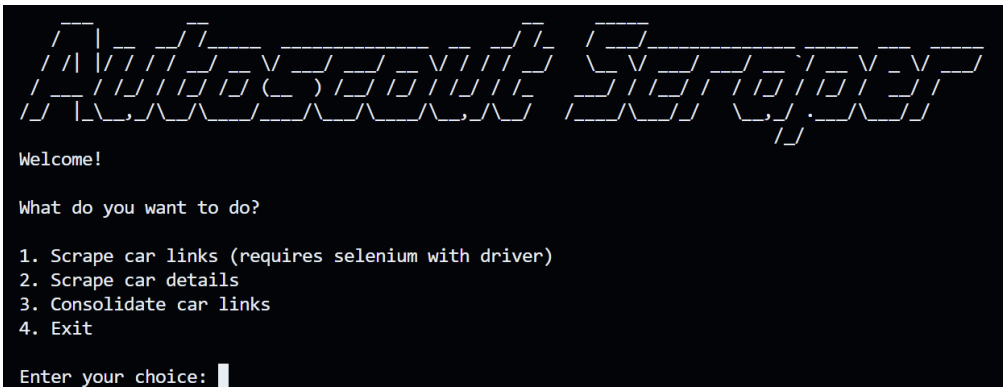

Figure 3: Screenshot of Autoscout Scraper

possibilities to go through. We of course parallelized and worked on multiple cores, also introducing shortcuts, e.g. if we found a make with less than *500* lisitng we did not need to loop through power ranges at all.

3.1.2 Scraping Car Details

Once we have collected all the car URLs, the scraping was quite straightforward. We have tested the scraping on individual listings first, trying to find out what categories of data we can scrape and how we may achieve that. At this stage, we were lucky, as no user interaction was needed at all, not even for **declining cookies**, as the underlying HTML structure already contained the data we needed. This allowed for us to scrape the data in a much more efficient way, as we could simply use the **requests** package and **BeautifulSoup** to parse the HTML and extract the data we needed. This had much lower overhead and allowed us to scrape more listings simultaneously. First fetching into a list in memory, then converting to a df, to then export into a **parquet** file.

Interesting to note that between the two scraping phases, many links seem to have moved, perhaps due to being sold, or delisted so this case also needed to be addressed.

3.2 Preprocessing

We worked separately in numeral Jupyter notebooks, there have been separate notebooks created per each member exploring different aspects of the data. We first started by cleaning the data at the start of each of our own separate notebooks, but later on we decided to do a comprehensive cleaning of the dataset in a separate joint notebook, which can serve as the basis for all of our future work.

The logic can be found in the appropriately named **data_cleaning.ipynb** notebook and it contains not only the cleaning logic, but our reasoning behind each of the cleaning steps. We will now summarize the the steps taken in a nutshell.

First off, we got rid of useless features, this included completely empty, almost empty columns and and simply features that we would not need for any of our analysis, like **manufacturer_colour** for example.

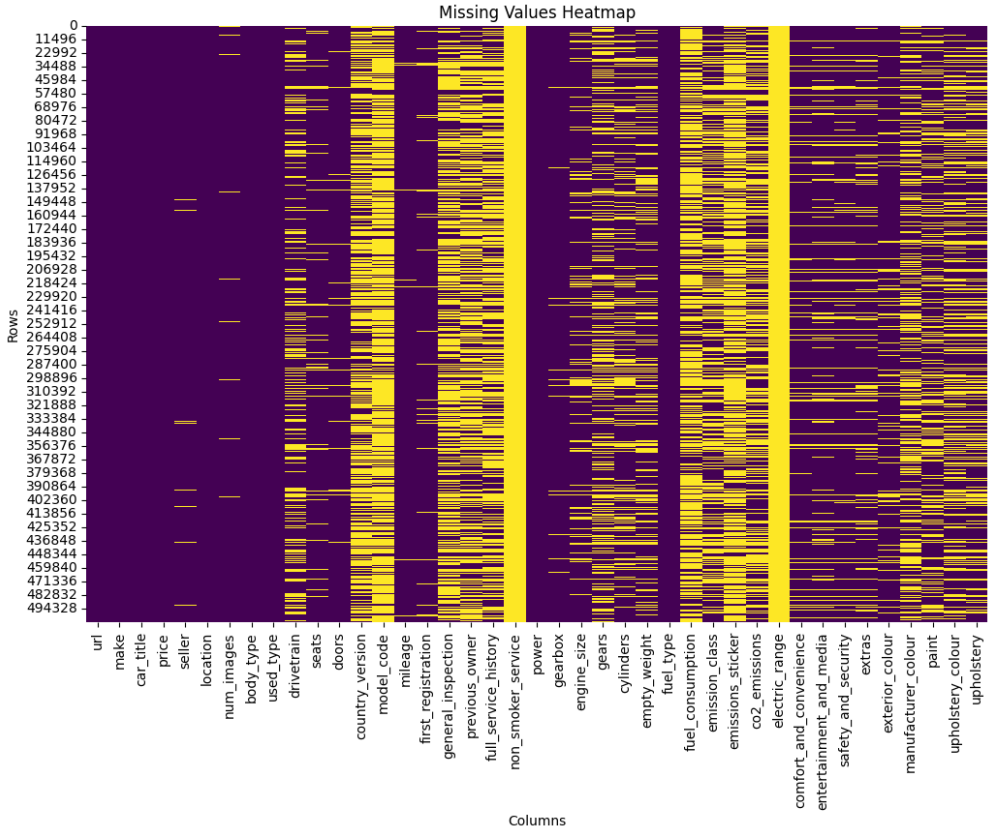

Figure 4: Missing Values Heatmap

Then, having seen that there are not too many missing values for our main features, **price**, **mileage** and **power**, we decided to drop the rows with missing values, removing *9879* rows, from our 504k large dataset. This seemed like a reasonable choice.

We also had to transform some columns to make them usable for our purposes:

- **Location:** was transformed from the format of `[city, country code]` to just the country code, as we were not interested in the fine grained location of the car.
- **Fuel consumption:** was similarly transformed from `[float] l/100 km (comb.)` string format to just the float value.
- **First registration:** was transformed to **age_months** by subtracting the first registration date from the current date, as we were only interested in the age of the car, not when it was registered.

We found a few imputable / inferable values in our dataset as well:

- **Age:** had seeminlgy *21454* missing values, wehich we could all infer from the column **use_type**, whenever this feature was **New** or **Demonstration**, we filled the missing values with *0*, as these cars were not used at all. This left us with no missing values at all in this column.
- **Fuel Type:** given that the fuel type was electric, we could safely infer that the **co2_emission** was **0** and that the **gearbox** was **automatic**.

In the end came the most interesting part, the outlier cleaning. We have inspected 8 of our important numerical features and found that there were some serious outliers in the dataset.

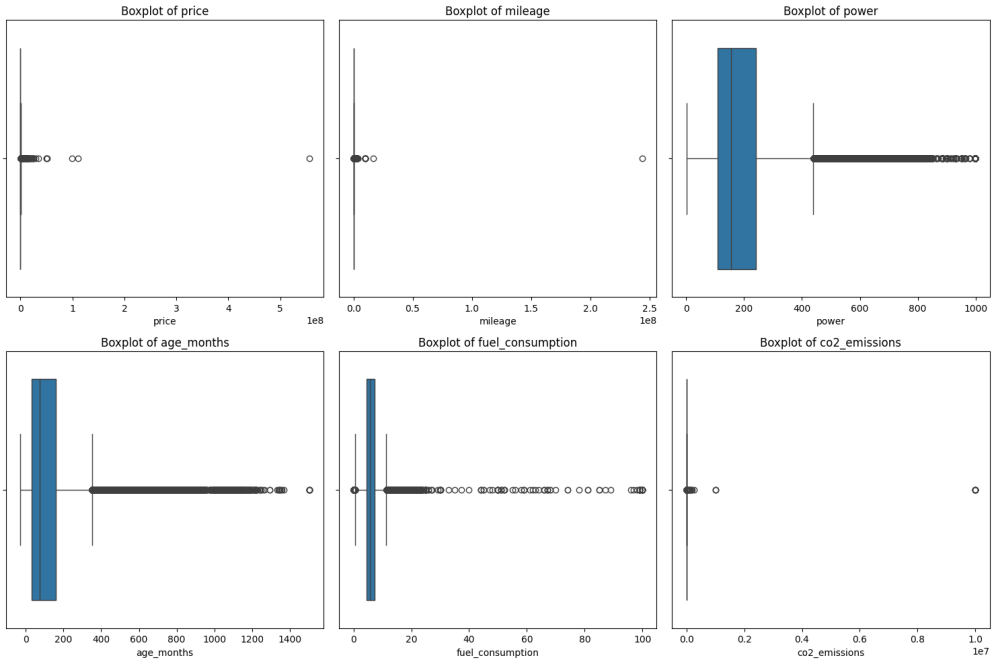

Figure 5: Boxplots of Numerical Features Before Cleaning

We have used many different strategies to clean the outliers, each based on what felt right for the feature, based on our domain knowledge. Sometimes, we used hard numerical caps, other times used IQR or even 0.05 quantile based capping. As we had so many records to work with, we did not believe winsorization was necessary, so we simply dropped the outliers. One of the more interesting features to clean was the **price** as, this had many outliers on both ends and was arguably our most important feature of all. Here we applied a two step process:

1. We cleaned the outlers with a lenient 2 * IQR based capping, per car make, as this allowed us to keep values that would have been outliers in the general dataset, but were not outliers at all, e.g. Lamborghini.

2. Then we manually inspected both ends of the distribution and set a hard cap of *500* EUR on the lower end and *1000000* EUR on the upper end, cleaning out the most extreme outliers. (This inspection led us to notice interesting things, e.g. towable trailers was listed as a car make, which we removed.)

After removing the outliers for other features, with similar care, but less manual inspection, we were left with a much cleaner dataset, as visible from the following boxplots.

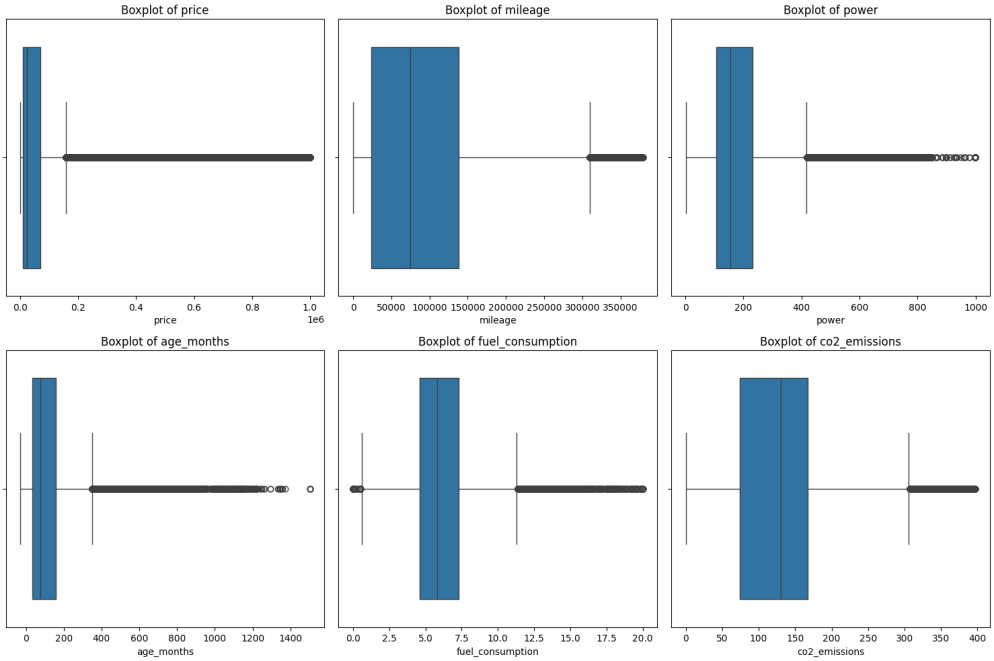

Figure 6: Boxplots of Numerical Features After Cleaning

The final dataset was then saved to an alternative parquet file, which we will be using for our further analysis. The cleaning here was on purpose not too harsh, such that each subsequent analysis can decide to further clean the data if needed, but we believe we have already found a good balance between keeping the data and cleaning it. In the end this cleaning process removed *36430* rows from our dataset, which is just *7.2%* of the original dataset. This seems reasonable, and we have addressed the the possible downsides of our process as losing all the data for some ultra premium car brands, like Koenigsegg, Paganni and losing some data in case a regular car brand happens to have premium models as well, e.g. Nissan GTR.

4 Results

Describe experiments and the evaluation protocol. Include tables, graphs, and charts as needed to present your findings and results.

5 Discussion

Interpret and analyze the results, and discuss possible future work and improvements.

6 Conclusion

Provide a summary of the project, key findings, and recommendations. (Approximately 200–350 words.)

7 Bibliography

Include relevant references such as websites, blogs, articles, research papers, etc.

References

[1] A. AlShared, “Used Cars Price Prediction and Valuation using Data Mining Techniques,” *Theses*, Dec. 1, 2021. [Online]. Available: `https://repository.rit.edu/theses/11086`.

8 Self-Reflection

Add a paragraph or half-page note reflecting on the project.