MARMARA UNIVERSITY

FACULTY OF ENGINEERING

CSE4062 – S25

Data Science Project Delivery #2
Descriptive Analytics

Car Price Prediction System

Group 8:

Muhammed Furkan Kahyaoğlu (ME) – 150420058, furkankahyaoglu@marun.edu.tr

Özlem Demirtaş (IE) – 150320006, demirtasozlem444@gmail.com

Niyazi Ozan Ateş (CSE) – 150121991, niyaziozanates@gmail.com

Doğukan Onmaz (CSE) – 150120071, dogukanonmaz@marun.edu.tr

Şükrü Can Mayda (CSE) – 150120031, canmayda@marun.edu.tr

Instructor: Dr. Murat Can Ganiz

Preprocessing

For the preprocessing of the data, we started by checking what kind of data types are there, and which data are meaningful to us.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 251079 entries, 0 to 251078
Data columns (total 15 columns):
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   Unnamed: 0               251079 non-null  int64
 1   brand                    251079 non-null  object
 2   model                    251079 non-null  object
 3   color                    250913 non-null  object
 4   registration_date        251075 non-null  object
 5   year                     251079 non-null  object
 6   price_in_euro            251079 non-null  object
 7   power_kw                 250945 non-null  object
 8   power_ps                 250950 non-null  object
 9   transmission_type        251079 non-null  object
 10  fuel_type                251079 non-null  object
 11  fuel_consumption_l_100km 224206 non-null  object
 12  fuel_consumption_g_km    251079 non-null  object
 13  mileage_in_km            250927 non-null  float64
 14  offer_description        251078 non-null  object
dtypes: float64(1), int64(1), object(13)
memory usage: 28.7+ MB
```

Figure 1: Types before preprocessing

We started by dropping the ‘id’, ‘offer_description’, and ‘registration_date’ columns. We dropped ‘id’ because it only shows the index at when data is inserted, beside that it has no contribution for our investigation, and we dropped it. The ‘offer_description’ has a lot of repetition, which is already known by the rest of the columns; hence it is meaningless. The reason for dropping the ‘registration_date’ column is based on the fact that it overlaps with the ‘year’ column. The only difference it got is that it has some months as well which will make the system more complicated and because of the correlation it won’t make a big difference if one of the two would stay.

```
brand                        0
model                        0
color                      166
year                         0
price_in_euro                0
power_kw                   134
power_ps                   129
transmission_type            0
fuel_type                    0
fuel_consumption_l_100km   26873
fuel_consumption_g_km        0
mileage_in_km              152
dtype: int64
```

Figure 2: Null values after dropping columns

The null values are checked, and we see that there is a lot of null variables. We started by filling in the null values for the categorical data ‘color’ and replaced it by ‘unknown’. Followed by this, some numeric values are stored as objects because it got its metrics as well stored. So, we had to normalize them, we cleaned and truncated these strings and converted them to numeric floats. After normalizing the data, we still had data with null values. So, we started by replacing the null values for the numeric data by their mean values.

```
brand                      0
model                      0
color                      0
year                       0
price_in_euro              0
power_kw                   0
power_ps                   0
transmission_type          0
fuel_type                  0
fuel_consumption_l_100km   0
fuel_consumption_g_km      0
mileage_in_km              0
dtype: int64
```

Figure 3: Null values after normalization and filling in their means

Now, since data fully cleaned, normalized and we don’t have any noisy data left, we can start to cluster and mine for some association rules.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 251079 entries, 0 to 251078
Data columns (total 12 columns):
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   brand                    251079 non-null  object
 1   model                    251079 non-null  object
 2   color                    251079 non-null  object
 3   year                     251079 non-null  object
 4   price_in_euro            251079 non-null  float64
 5   power_kw                 251079 non-null  float64
 6   power_ps                 251079 non-null  float64
 7   transmission_type        251079 non-null  object
 8   fuel_type                251079 non-null  object
 9   fuel_consumption_l_100km 251079 non-null  float64
 10  fuel_consumption_g_km    251079 non-null  float64
 11  mileage_in_km            251079 non-null  float64
dtypes: float64(6), object(6)
memory usage: 23.0+ MB
```

Types after preprocessing

Analysing the Data

We see in our data that most cars range between the costs of 1000 and 50000 euros. While there are also some outliers. The selling price distribution shown below:

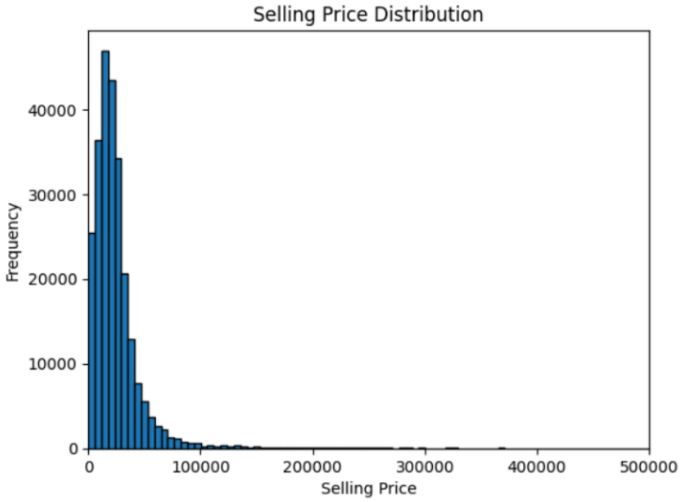

Figure 4: Selling Price Distribution

To understand what the prices are based on we made a scatterplot between the columns 'price_in_euro' and 'mileage_in_km'. In here we can see that the prices of cars which has a higher odometer are lower compared to those with a lower odometer.

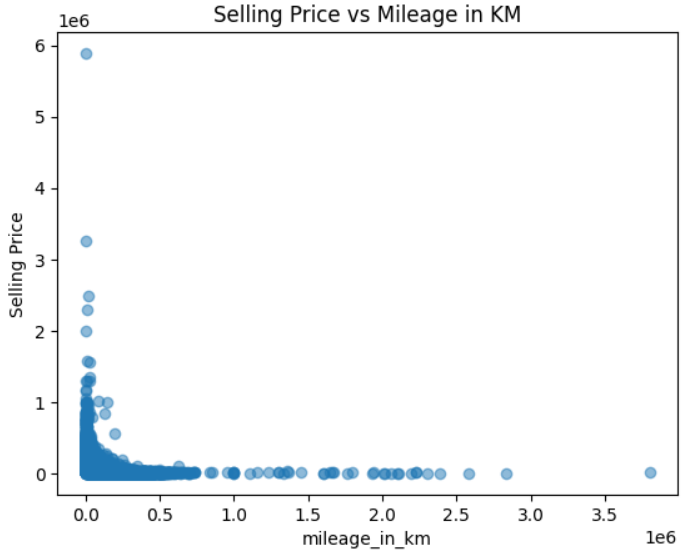

Figure 5: Scatterplot for the selling price and the odometer

The correlation matrix can only be obtained by numeric values. So, we created the matrix and realized that 'power_ps' and 'power_kw' are strongly correlated, while 'milage_in_km' and 'price_in_euro' are negatively correlated. Below I leave the correlation matrix:

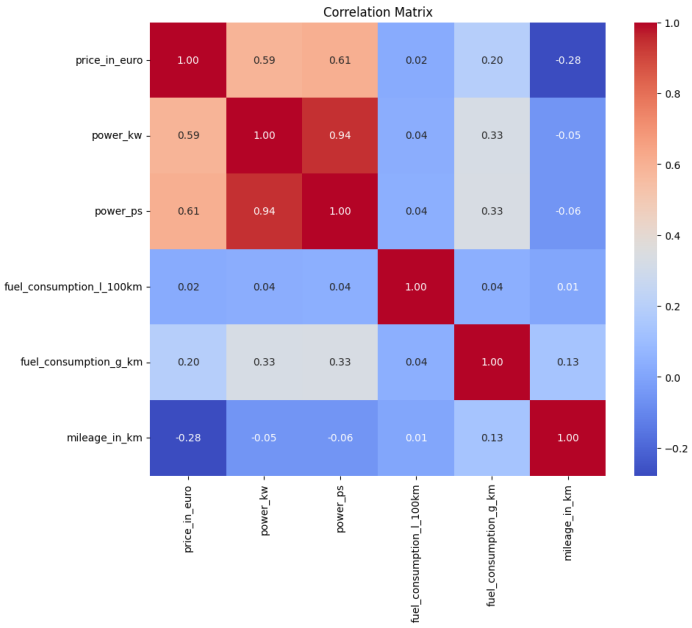

Figure 6: The Correlation Matrix for Numeric Values

Furthermore, we made a boxplot for the selling price. For readability, we ranged the values between 0 and 500000 euros. In the boxplot below we see the that the mean is around 3000 euros and the 3rd quartile around 7000 euros, meaning that there are a lot of outliers:

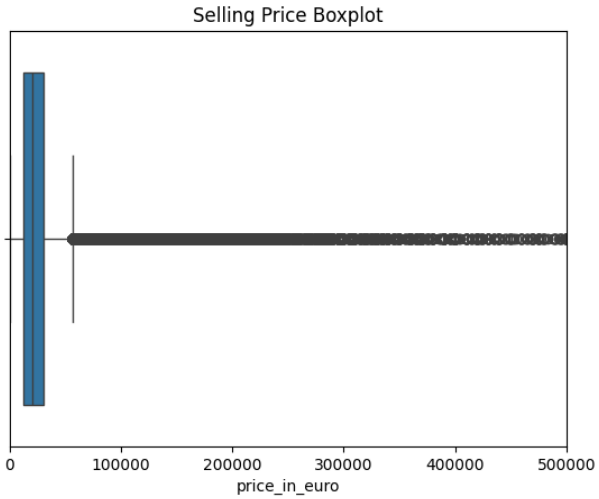

Figure 7: Boxplot for the Selling Price of the Cars

Clustering

Z-Score Normalization

To ensure all numerical features are on a comparable scale, we applied Z-score normalization to the dataset. First, numeric columns were identified using their data types (e.g., float64, int64). Then, the StandardScaler from the Scikit-learn library was used to standardize these columns. This process transforms the data so that each numeric feature has a mean of 0 and a standard deviation of 1.

This normalization is important for distance-based clustering algorithms such as DBSCAN, where varying scales can bias the clustering process. Below is an example of the mean and standard deviation before scaling:

	price_in_euro	power_kw	power_ps	fuel_consumption_l_100km	fuel_consumption_g_km	mileage_in_km
mean	26137.530002	126.477379	171.809526	6.487279	140.802646	85340.015985
std	36973.292968	75.257813	99.150710	26.746251	61.596270	78693.230409

Figure 8: Mean and std dev before scaling

Below are the first five rows showing how the data appears after Z-score normalization. A value of 0 indicates that the data point is at the average, negative values indicate it is below the average, and positive values indicate it is above the average. The larger the absolute value, the further the data point is from the mean, meaning it is more distinct compared to other values. The normalized data table is presented below:

	price_in_euro	power_kw	power_ps	fuel_consumption_l_100km	fuel_consumption_g_km	mileage_in_km
0	-0.671771	0.285986	0.294406	0.164985	1.935143	0.955103
1	-0.033471	0.857356	0.889461	0.000000	0.000000	1.329977
2	-0.547356	-0.218946	-0.219964	0.000000	0.000000	0.554814
3	-0.574403	-0.218946	-0.219964	0.112641	1.366926	1.323623
4	-0.221445	0.073383	0.072521	0.026648	0.000000	0.137077

Figure 9: Normalized data

K-Means Clustering and Elbow Method

To explore alternative clustering methods beyond DBSCAN, we also applied K-means clustering. To determine the optimal number of clusters (k), the Elbow Method was used. This technique involves plotting the Within-Cluster Sum of Squares (WCSS) against various values of k and identifying the point where the rate of decrease sharply changes.

We standardized the dataset using StandardScaler prior to clustering to ensure that all features contribute equally to the distance calculations. The elbow plot shown below suggests a clear bend, which can be interpreted as the optimal k value for K-means.

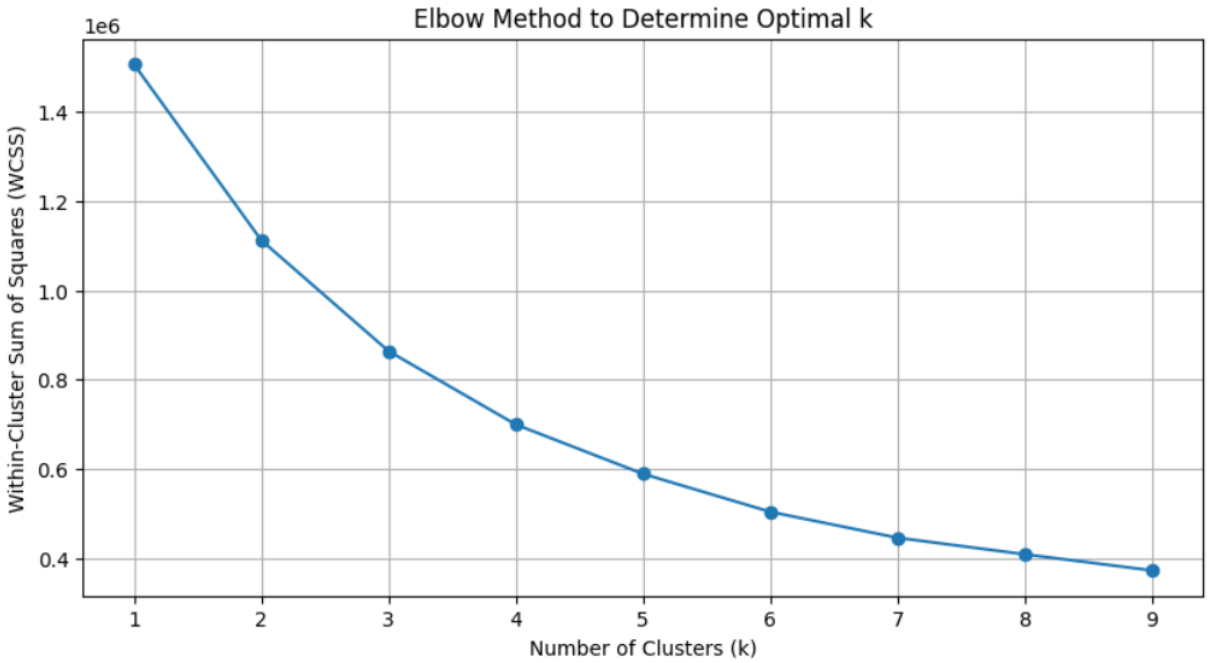

Figure 10: WCSS changes

The graph shows us that how WCSS changes in number of clusters. The bigger number of clusters means the smaller WWCS. As seen from 1 cluster to 2 clusters WWCS decreases sharply.

Hierarchical Clustering Analysis

In addition to Z-Score and K-means, we studied Hierarchical Clustering to visualize the relationships between data points in a tree-like structure called a dendrogram. Due to computational complexity, we randomly sampled 1,000 rows from the normalized dataset.

The ward linkage method was chosen because it minimizes the variance within each cluster during the merging process. The dendrogram below represents the distances (using Euclidean distance) between merged clusters and helps reveal the number of natural groupings in the data.

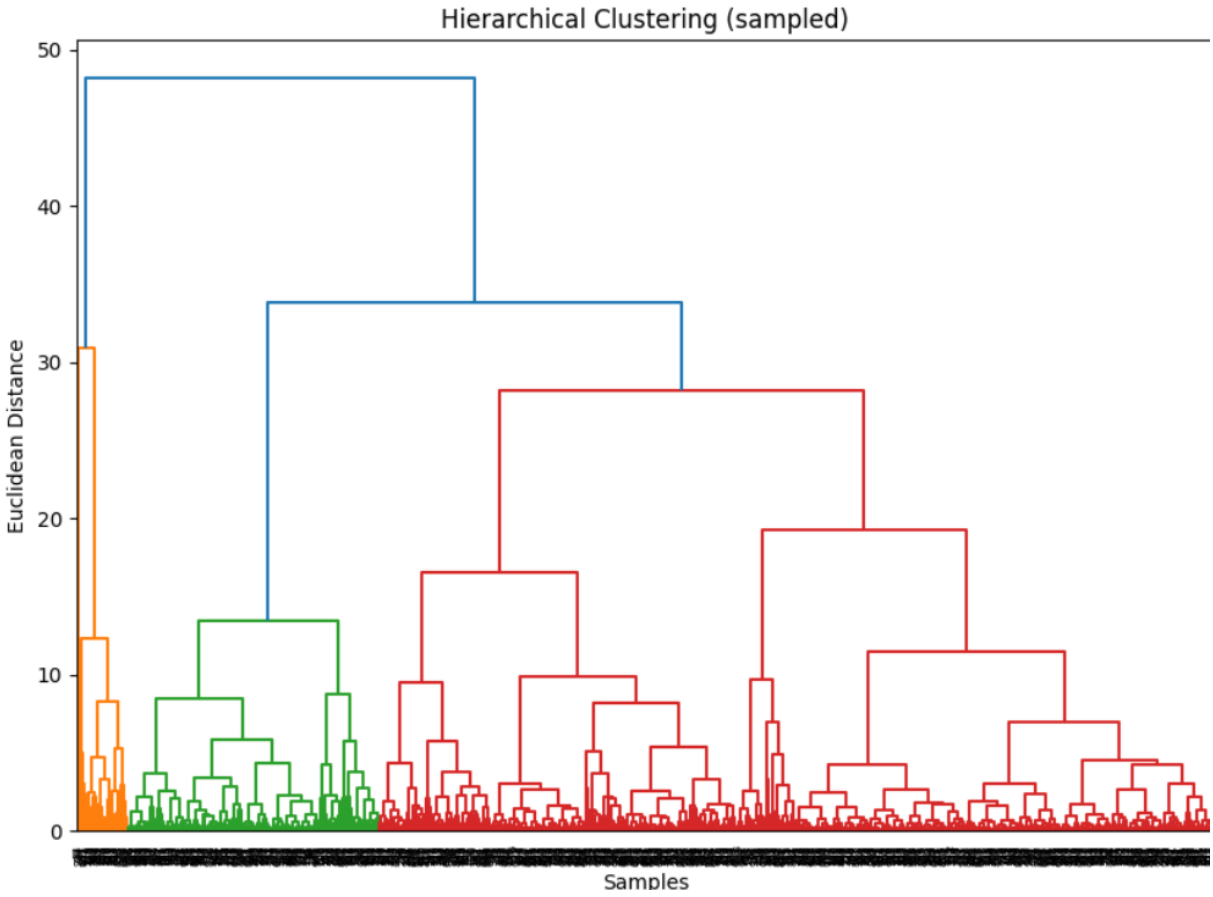

Figure 11: The Dendrogram

DBSCAN

The aim of this analysis is to identify **descriptive patterns** in vehicle-related data that can help in understanding customer or product segments. Using **DBSCAN** (Density-Based Spatial Clustering of Applications with Noise), we explore clusters in a 10% sampled dataset (~25,000 rows) with numerical attributes such as price, power, mileage, year, and fuel consumption.

Methodology

1. **Data Cleaning**
    a. Non-numeric characters (e.g., commas in fuel consumption) were cleaned.
    b. Columns like price_in_euro, power_kw, mileage_in_km, and year were converted to numeric types.
    c. Missing values were imputed using column means.
2. **Sampling**
    a. A 10% random sample was taken from the full dataset to reduce computational cost while preserving representativeness.
3. **Feature Scaling**
    a. Features were standardized using StandardScaler to ensure fair distance measurements for DBSCAN.
4. **Clustering with DBSCAN**
    a. DBSCAN was applied with eps=1.0 and min_samples=5.

b. The algorithm identified **7 clusters**, including noise points (cluster = -1).

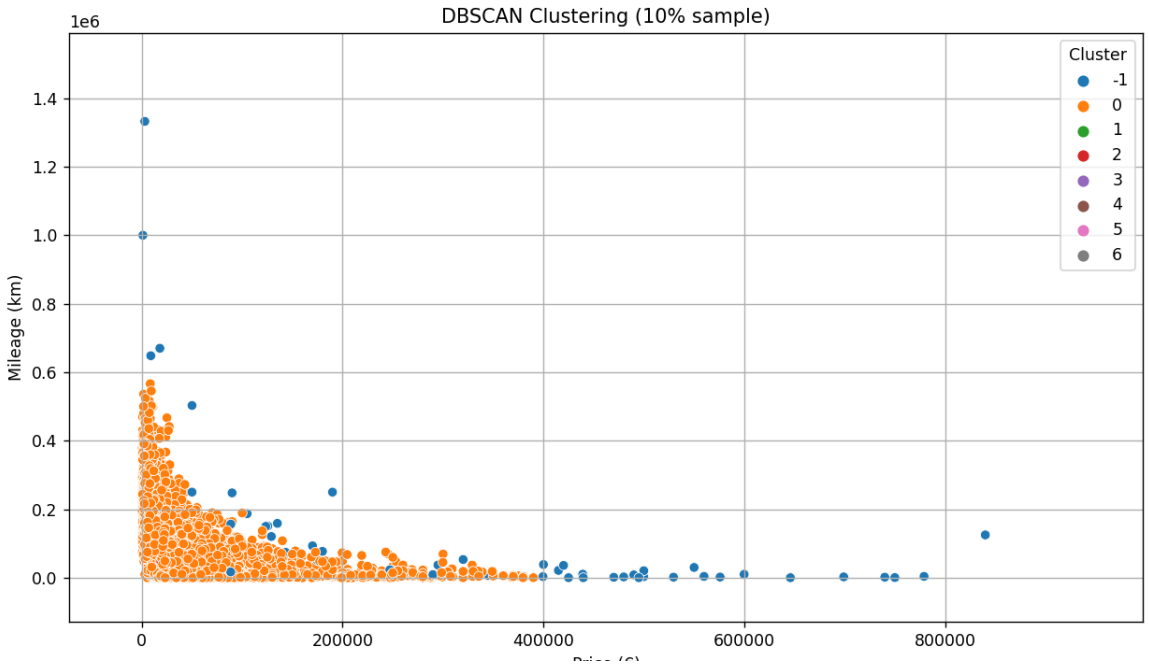

Figure 12: DBSCAN Clustering

```
Cluster counts:
 0    24867
 2      105
-1      101
 6       10
 1        9
 3        6
 5        5
 4        5
Name: cluster, dtype: int64

Cluster-wise mean and median values:
           year         price_in_euro  ... fuel_consumption_l_100km mileage_in_km
           mean  median          mean  ...                   median          mean   median
cluster                                ...
0       2016.27  2018.0      25234.02  ...                      5.7      85417.42  67700.0
1       2021.11  2021.0      27186.22  ...                    316.0      14173.00  11800.0
2       2019.67  2020.0      24846.31  ...                    100.0      40624.23  30300.0
3           NaN     NaN           NaN  ...                      NaN           NaN      NaN
4           NaN     NaN           NaN  ...                   2020.0           NaN      NaN
5       2019.80  2020.0      23654.00  ...                    392.0      24048.40  25500.0
6       2022.00  2022.0      33591.90  ...                    479.0       5459.20   3175.0

[7 rows x 10 columns]
```

Figure 13: DBSCAN Clustering

Results:

- Cluster 0 is the dominant group, containing the vast majority of samples.
- Clusters 1, 3, 4, 5, and 6 are small, potentially outlier segments.

- Noise points (-1) indicate samples that do not belong to any cluster

Cluster 0: Represents average used vehicles — moderate mileage (85k km), moderate price (€25k), and fuel consumption (~5.7 L/100km). This likely reflects the mass market or mainstream segment.

Cluster 6: Very new and expensive vehicles, possibly electric or luxury — very low mileage (5.5k km) and high price (€33k). Could indicate new inventory or premium segment.

Cluster 1: Fairly recent cars (2021), very low mileage, but abnormally high fuel consumption (316 L/100km) suggests potential data quality issues or incorrect parsing. Needs further inspection.

Clusters 2 and 5: Mid-tier groups, newer than Cluster 0 and with moderate mileage and prices. Could reflect fleet vehicles or lease returns.

Outliers & Noise: Clusters 3, 4, and noise (-1) likely represent vehicles with missing/incorrect data or genuinely rare configurations.

Association Rule Mining (ARM)

This analysis applies the Apriori algorithm to extract frequent patterns and association rules from a dataset of used vehicles. The objective is to identify meaningful relationships between attributes such as brand, model, transmission type, fuel type, price range, year of production, and mileage. These insights can be useful for strategic business decisions such as pricing, inventory planning, and targeted marketing.

Data Preparation

- The dataset contains structured information about used vehicles.
- Key numerical attributes (e.g., price_in_euro, mileage_in_km, and year) were cleaned and transformed into categorical bins to enable association rule mining.
    - price_in_euro was grouped into: low, mid-low, mid, mid-high, high.
    - year was grouped into: old, mid-old, recent, new, very new.
    - mileage was grouped into: very low, low, medium, high, very high.
- One-hot encoding was applied to transform all categorical features into a binary format suitable for Aprior

Frequent Pattern Mining

- Apriori algorithm was applied with:
    - Minimum support: 0.2
- Association rules were then generated with:
    - Minimum lift threshold: 1.0
- Rules were filtered further to find interesting patterns, defined as:

- Lift > 1.2
- Confidence > 0.6

```
 Interesting Apriori Rules:
                 antecedents                 consequents  ...  confidence      lift
2  (transmission_type_Manual)          (fuel_type_Petrol)  ...    0.693864  1.215903
5       (price_range_mid-low)  (transmission_type_Manual)  ...    0.639288  1.361782
```

Figure 14: Apriori Rules

Interpretation of Results

1. **Manual transmission vehicles are strongly associated with petrol engines**
    a. With a **confidence of ~69%** and **lift of 1.21**, this rule suggests that if a car has a manual transmission, it is significantly more likely to have a petrol engine than would be expected by chance.
    b. **Business implication**: When stocking manual transmission vehicles, dealerships can prioritize petrol models as a compatible offering.
2. **Mid-low priced vehicles are commonly manual transmission**
    a. With **~64% confidence** and a **lift of 1.36**, this rule shows a strong association between mid-low pricing and manual transmissions.
    b. **Business implication**: In pricing strategies or budget car offerings, manual vehicles are likely to dominate. This can influence marketing campaigns or promotional bundles.

Another Apriori Method From Reprocessing

To discover meaningful patterns among categorical variables, we applied Association Rule Mining (ARM) again using the Apriori algorithm. First, missing values in categorical features were replaced with 'unknown', and the data was transformed into a list of transactions. Then, one-hot encoding was applied using TransactionEncoder from the mlxtend library to prepare the data for frequent itemset generation.

Frequent itemsets were identified with a minimum support of 0.02, and association rules were generated based on confidence ≥ 0.4. The top 10 rules were then sorted by lift, which measures how much more likely the consequent is given the antecedent than by random chance.

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhangs_metric	jaccard	certainty	kulczynski
173	(Volkswagen Golf)	(volkswagen, Petrol)	0.030222	0.072240	0.020631	0.682657	9.449818	1.0	0.018448	2.923522	0.922043	0.252117	0.657947	0.484123
170	(Volkswagen Golf)	(Manual, volkswagen)	0.030222	0.076737	0.020137	0.666315	8.683125	1.0	0.017818	2.766872	0.912408	0.231937	0.638581	0.464366
172	(Volkswagen Golf, Petrol)	(volkswagen)	0.020631	0.132552	0.020631	1.000000	7.544214	1.0	0.017896	inf	0.885721	0.155644	1.000000	0.577822
81	(Volkswagen Golf)	(volkswagen)	0.030222	0.132552	0.030222	1.000000	7.544214	1.0	0.026216	inf	0.894481	0.227998	1.000000	0.613999
168	(Volkswagen Golf, Manual)	(volkswagen)	0.020137	0.132552	0.020137	1.000000	7.544214	1.0	0.017468	inf	0.885275	0.151919	1.000000	0.575959
29	(Electric)	(Automatic)	0.023769	0.524731	0.022929	0.964645	1.838360	1.0	0.010456	13.442672	0.467140	0.043627	0.925610	0.504171
30	(Hybrid)	(Automatic)	0.050307	0.524731	0.046619	0.926688	1.766025	1.0	0.020221	6.482850	0.456733	0.088223	0.845747	0.507766
147	(ford, Petrol)	(Manual)	0.041198	0.469450	0.032563	0.790410	1.683694	1.0	0.013223	2.531370	0.423516	0.068112	0.604957	0.429888
130	(mercedes-benz)	(Automatic, Petrol)	0.108436	0.241784	0.043687	0.402887	1.666306	1.0	0.017469	1.269802	0.448504	0.142521	0.212476	0.291787
153	(opel)	(Manual, Petrol)	0.081202	0.325846	0.044026	0.542182	1.663922	1.0	0.017567	1.472537	0.434274	0.121277	0.320900	0.338647

Figure 15: Top 10 rules

The interesting thing that observed is that if a car is Volkswagen Golf, then it is likely to be Petrol, Lift = 9.44 and Confidence = 0.68. Also the confidence of electric car is automatic is 0.96, confidence of hybrid car is automatic is 0.92. These are pretty high confidence scores. However the list is ascending by lift, so that it is not shown in the first row.

Results and Conclusion

In this study, we developed a descriptive analytics pipeline for a used car dataset to uncover hidden structures and meaningful relationships. The analysis was conducted through muliple techniques: clustering with DBSCAN, K-Means, Hierarchical Clustering, and association rule mining with the Apriori algorithm.

Clustering analysis revealed distinct customer or product segments based on key numerical features suc as mileage, price, and production year. DBSCAN successfully identified major clusters and potential outlier segments, with cluster 0 representing the average second-hand vehicle. K-Means clustering, guided by the Elbow Method, provided an alternative perspective by grouping vehicles based on their numerical similarity, while Hierarchical Clustering offered a tree-based visualization of how data points are related, revealing natural groupings in the sample. The use of multiple clustering methods allowed for more robust segmentation and supported validation across techniques.

The association rule mining further contributed by uncovering frequently co-occurring attribute patterns in the data. For instance, manual transmission vehicles showed a strong association with petrol engines, and mid-low priced cars were often manual. Additional Apriori analyses on reprocessed categorical data yielded meaningful insights, such as the strong association between Volkswagen Golf and petrol engines, or between electric/hybrid cars and automatic transmissions.

Overall, this project highlighted the practical value of descriptive analytics in organizing and interpreting vehicle-related data. Further improvements, especially regarding data quality in certain clusters, would enhance the results. Nevertheless, the methodology provides a solid foundation for developing predictive models and supporting strategic decisions in the automotive sector.