**Project\_Methodology**

**Files and sort explanations**

**File 1:** **cleaning\_funs**

In this project, we are cleaning and preprocessing various columns in our dataset as part of a machine learning assignment. To manage the complexity and maintain clean, modular code, we have organized our data cleaning functions into a separate Python script. Each function is designed to clean a specific column in the dataset. By importing this script into our main Jupyter Notebook, we can easily apply these cleaning functions to the dataset. This approach not only helps in keeping our main notebook tidy and readable but also promotes code reusability and maintainability.

Here's a step-by-step outline of the process:

Separate Cleaning Functions: Each column in the dataset has a dedicated function that handles its cleaning process. These functions are defined in a separate Python script.

Importing the Script: The script containing the cleaning functions is imported into the main Jupyter Notebook using the import statement.

Applying the Functions: Once imported, these functions are applied to the respective columns in the dataset within the main notebook. This ensures that the data is cleaned systematically and consistently.

**File 2: column\_prediction\_models**

In this project, we are predicting and filling missing values in the 'color' and 'total ownership' columns of our dataset using predictive models as part of a machine learning assignment. We decided to use a predictive model for these two columns because they had a significant number of missing values. After consideration, we chose the K-Nearest Neighbors (KNN) model since it is well-suited for predicting categorical columns. The KNN model provided good prediction metrics, allowing us to fill in the missing values more accurately.

Importing the Script: The script containing the modeling functions is imported into the main Jupyter Notebook using the import statement.

**File 3: Data correlations and scatter**

This notebook is designed for initial data exploration using graphical representations and statistical correlations. It focuses on understanding data distributions, relationships between variables, and identifying patterns that can inform subsequent predictive modeling. We will use Python libraries such as pandas for data manipulation, matplotlib and seaborn for data visualization, and scipy for statistical analysis.

This entire process is conducted on our dataset after preprocessing to better understand the data and ensure that we have filled the columns correctly.

**File 4: Features\_important**

Use techniques like Univariate Selection, RFE, PCA, and model-based importance to identify key features.

In practice, we found that the best model was achieved using all the available data rather than selecting a specific number of features. As a result, we decided not to use the feature selection code in our final approach. However, by analyzing which features were deemed important, we were able to improve our understanding of the dataset and assess how well we filled in the missing values.

**File 5: model**

This is the main file where the overall code execution takes place. It imports code from the other attached files and uses them for cleaning missing values and problematic entries in the columns, filling the 'color' and 'total ownership' columns using models, selecting the best features for predicting car prices, and finally, building the prediction model. Additionally, the original dataset is also attached.

**Libraries**

* pandas
* numpy
* datetime
* re
* seaborn
* matplotlib.pyplot
* warnings
* collections
* sklearn.model\_selection
* sklearn.linear\_model
* sklearn.preprocessing
* sklearn.impute
* sklearn.pipeline
* sklearn.compose
* sklearn.metrics
* sklearn.feature\_selection
* joblib