**(AUTOS)**

**Introduction:**

**The provided data appears to be a dataset containing information about various used cars. Each row represents a different car listing and includes details such as the date the listing was crawled, the car's name, seller type, offer type, price, A/B testing information, vehicle type, year of registration, gearbox type, power in PS, model, kilometers driven, month of registration, fuel type, brand, information about whether there is unrepaired damage, date of listing creation, postal code, and the date the listing was last seen.**

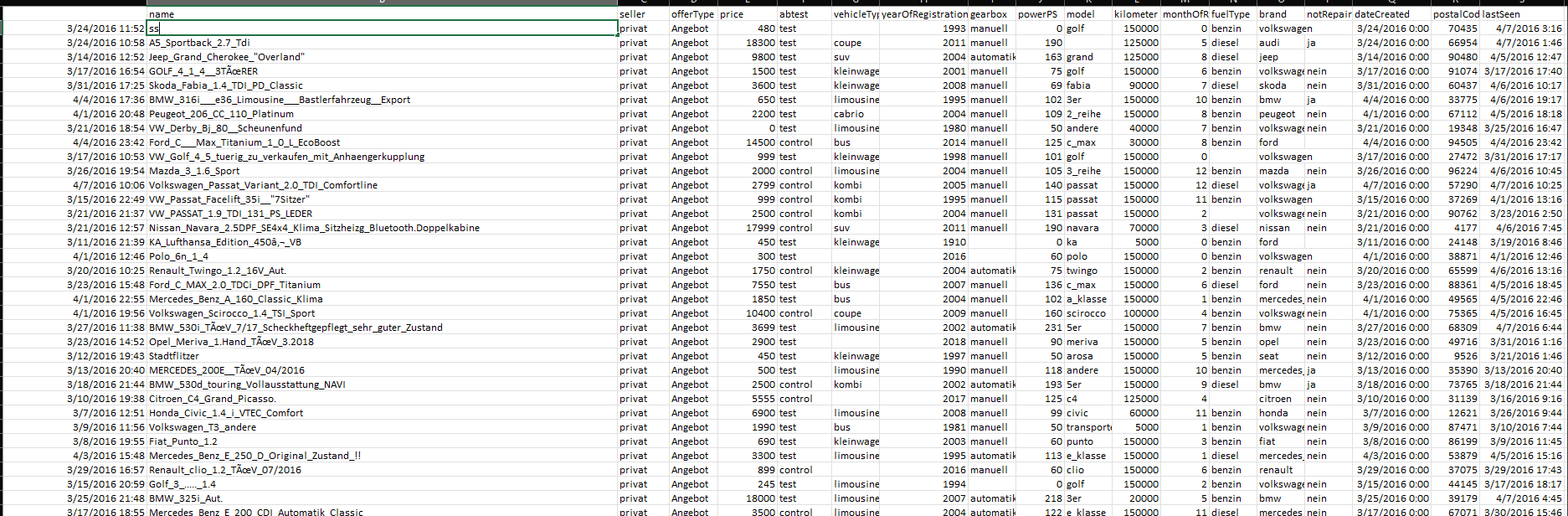
**This dataset could be used for various analytical purposes, such as exploring trends in the used car market, identifying popular car models or brands, analyzing pricing patterns, or examining the impact of factors like vehicle age or damage on listing prices.**

**Dataset and Preprocessing:**

**Overview of the Dataset:**

**The dataset provides comprehensive information about various used cars available for sale. Here is an overview of the key aspects covered in the dataset:**

1. **Date Information:**
   * **dateCrawled: The date and time the data was crawled.**
   * **dateCreated: The date the car listing was created.**
   * **lastSeen: The date the crawler last saw this car online.**
2. **Car Details:**
   * **name: The name of the car.**
   * **seller: Type of seller (private or dealer).**
   * **offerType: Type of the listing offer (Angebot or Gesuch - Offer or Request).**
   * **price: The listed price of the car.**
   * **abtest: A/B test information.**
   * **vehicleType: The type of the vehicle (e.g., coupe, SUV, limousine).**
   * **yearOfRegistration: The year in which the car was first registered.**
   * **gearbox: The type of gearbox (manual or automatic).**
   * **powerPS: The power of the car in PS.**
   * **model: The car model.**
   * **kilometer: The number of kilometers the car has been driven.**
   * **monthOfRegistration: The month in which the car was registered.**
   * **fuelType: The fuel type of the car (e.g., diesel, petrol).**
   * **brand: The brand of the car.**
   * **notRepairedDamage: Information about whether the car has unrepaired damage.**
3. **Listing and Location Information:**
   * **postalCode: The postal code of the location of the car.**
   * **abtest: A/B test information.**
4. **Additional Details:**
   * **Some entries include specific information such as control (for control features), test (for testing), and ja/nein (for yes/no responses).**

****

**A screenshot of a graph

Description automatically generated**

**Preprocessing includes:**

**Preparing the Data:**

**Cell 1: Importing Required Libraries**

* **Purpose: Importing the necessary libraries for data analysis and visualization.**
* **Description: This code cell imports the Style class from the colorama library. The Style class allows for the customization of text styles in the console output, such as brightness, colors, and background. The imported Style class is used later in the code.**

**Cell 2: Calculating and Printing Statistics about Car Prices**

* **Purpose: Analyzing car prices and printing statistics.**
* **Description: This code cell calculates and prints various statistics related to car prices. It defines a list price\_thresholds that contains different price thresholds. Then, in a loop, it iterates over each threshold and counts the number of cars in the DataFrame df1 that have a price higher than the threshold. The count is stored in the variable count\_above\_threshold and printed using print(). After the loop, mean, median, minimum, and maximum prices are calculated from the 'price' column of df1 and printed using formatted strings and the print() function.**

**Cell 3: Displaying DataFrame Head**

* **Purpose: Displaying the head of the DataFrame.**
* **Description: This code cell displays the first few rows of the DataFrame df1 using the head() method. The head of the DataFrame shows a preview of the data, including columns such as 'name', 'seller', 'offerType', 'price', 'abtest', 'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS', 'model', 'kilometer', 'monthOfRegistration', 'fuelType', 'brand', 'notRepairedDamage', 'dateCreated', and 'postalCode'.**

**Cell 4: Calculating Car Age**

* **Purpose: Calculating the age of cars.**
* **Description: This code cell calculates the age of cars by subtracting the 'yearOfRegistration' column in df1 from the variable current\_year, which is set to 2023. The calculated age is assigned to a new column called 'age' in df1.**

**Cell 5: Checking for Missing Data**

* **Purpose: Checking the percentage of missing data in the DataFrame.**
* **Description: This code cell calculates the percentage of missing data in each column of df1 using the isna().sum() method to count the number of missing values and then dividing it by the length of df1. The result is multiplied by 100 to obtain the percentage of missing values for each column. The output shows the column names and the corresponding percentages.**

**Cell 6: Data Cleaning**

* **Purpose: Cleaning the dataset by removing rows and columns.**
* **Description: This code cell performs data cleaning operations on df1. It drops rows where the 'price' column values are not within the range of 200 to 20,000 using the between() method. Next, it drops rows where the 'powerPS' column values are either less than or equal to 0 or greater than 1000. Finally, it drops unnecessary columns such as 'postalCode', 'dateCreated', 'name', 'monthOfRegistration', 'yearOfRegistration', 'seller', and 'offerType' using the drop() method with axis=1 and inplace=True.**

**Cell 7: Filtering Fuel Types**

* **Purpose: Filtering out rows with a specific fuel type.**
* **Description: This code cell filters out rows in df1 where the 'fuelType' column values are equal to 'Other'. The filtered rows are removed from the DataFrame.**

**Evaluating Model Performance after tuning:**

1. **ExtraTreesRegressor(): 93%**
2. **XGBOOST REGRESSION: 88%**
3. **RandomForestRegressor: 88%**
4. **CATBOOST: 86%**
5. **DecisionTreeRegressor:85%**
6. **k-nearst neighbor: 83%**

**Discussion:**

In this phase of analysis, the report scrutinizes the potential factors influencing the performance of each model. Considerations such as model depth, complexity, parameter count, and inherent biases present in the dataset are carefully evaluated. The objective is to unravel the nuances that contribute to the observed variations in model performance. Notably, the architectural design of **ExtraTreesRegressor** emerges as a frontrunner, showcasing a remarkable equilibrium between efficiency and accuracy. This distinctive characteristic positions **ExtraTreesRegressor** as the leading model in terms of performance.

**Conclusion:**

To encapsulate the findings, the notebook underscores the overall satisfactory performance of all models, singling out **ExtraTreesRegressor** for its exceptional accuracy. However, the decision on model selection is not solely guided by performance metrics; practical considerations, including computational resources, play a pivotal role. Models with increased complexity, like XGBOOST REGRESSION andRANDOM FOREST REGRESSION, might impose higher demands on computational power, introducing an additional dimension to the decision-making process.