School of Management, Economics, Law, Social Sciences, and International Affairs

Programming with Advanced Computer Languages

Python Group Project: Analysis of a Dataset of Used Cars

Authors

Casale Alice (20-993-465)

Bucci E l i a (21-616-065)

Bortoletti Aurora (19-994-565)

Lecturer

Silic Mario

May, 2024

Assignment Overview

The goal of this project is to use Python and Jupyter notebook to explore, analyze and visualize the Used Cars dataset. To solve the assignment we applied the knowledge gained from the theoretical and practical exercises. We used Seaborn, Matplotlib, and Pandas visualization libraries.

Assignment Deliverable

The deliverable for this assignment is the following file:

Assignment Background

The goal of our project is to analyse the availability of used cars in the market.

We want to help the user to understand the following informations:

- Analysing whether the most convenient are diesel or benzine-used cars.
- Compare the range of prices for different brand.
- Identify the most frequent models for each brand.
- Analysing the distribution of used cars based on the registration year and calculating the numbers of vehicles available for each range.

Steps to Do Before Running the Code

1. Install the required libraries: numpy, csv, matplotlib, seaborn and pandas
2. Access your Phyton Console

DATA SET DESCRIPTION:

This dataset is scraped from Ebay. The content of the dataset is in German, but it should not impose critical issues in understanding the data. The fields included in the dataset are as following:

Column name	**Description**
dateCrawle	when this ad was first crawled, all field-values are taken from this date
name	”name” of the car
seller	private or dealer
offerType	'Angebot', 'Gesuch'
price	at which year the car was first registered
abtest	'test', 'control'
vehicleType	type of vehicle ('coupe', 'suv', 'kleinwagen', 'limousine', 'cabrio', 'bus','kombi', 'andere')
yearOfRegistration	at which year the car was first registered
gearbox	'manuell', 'automatik’
powerPS	power of the car in PS
model	model of the vehicle
kilometer	how many kilometers the car has driven
monthOfRegistration	at which month the car was first registere
fuelType	vehicle fuel type
brand	vehicle brand
notRepairedDamage	if the car has a damage which is not repaired yet
dateCreated	the date for which the ad at ebay was created
nrOfPictures	number of pictures in the ad
postalCode	nostal code of the city where the vehicle is sold
lastSeenOnline	when the crawler saw this ad last online

STEP 0: IMPORT PHYTON LIBRARIES

```
import csv
import numpy as np
import matplotlib.pyplot as plt
```

```
import pandas as pd
import seaborn as sns
```

STEP 1: DATA CLEANING

In the Used Cars dataset, we identified the missing and invalid values for the columns: `vehicle type`, `price`, `brand`, and `month of registration`. We standardized the information and converted it to a unique value. We identified for each column the number of missing or invalid instances. The prices are in euros and the range of accepted prices is between €1000 and €100000.

Once we identified missing/invalid values for the given columns, we removed all rows where one or more columns had invalid/missing data.

1.1 Definition of function "delete_multiple_element"

```
# Function that deletes multiple elements in a list
def delete_multiple_element(list_object, indices):
    indices = sorted(indices, reverse=True)
    for idx in indices:
        if idx < len(list_object):
            list_object.pop(idx)
```

Indexes are sorted in descending order. This is important because when elements are deleted from a list the indexes are altered. It is therefore necessary to sort them in descending order to avoid making mistakes such as deleting wrong elements.

1.2 Reading the CSV file

```
with open("./used_cars_dataset.csv", encoding = "ISO-8859-1") as file:
    # Read initial file
    csv_reader = csv.reader(file)
    # Retrieve headers of the columns
    header = next(csv_reader)
    # Retrieve rows of the csv file
    rows = []
    for row in csv_reader:
        rows.append(row)
```

We now read the CSV file and then extract the name of the CSV file columns and all existing rows within the dataframe.

1.3 Localization of Columns Indexes

```
# Retrieve headers indexes
index_VT = header.index('vehicleType')
index_price = header.index('price')
index_brand = header.index('brand')
index_month_registration = header.index('monthOfRegistration')
```

Through the following lines of code we get the location of the columns we are interested in (whether they are the first, second, and so on).

1.4 Filtering of Missing/Invalid Data

```
# Filter all missing/invalid data
to_be_deleted = []
for idx, r in enumerate(rows):
    if r[index_brand] == '' or r[index_VT] == '' or (int(r[index_price]) <
   1000 or int(r[index_price]) > 100000) or (int(r[index_month_registration]) < 1 and
                                            int(r[index_month_registration]) > 12):
        to_be_deleted.append(idx)
```

We create an empty list called to_be_deleted and run a for loop that will repeat for the length of the dataframe. Its goal is to select all rows that have invalid data in the 4 columns we are interested in (for example if the month of record is higher than 12 or lower than one it is deleted as it does not exist).

1.5 Invalid Data Deletion

```
# Remove invalid data
delete_multiple_element(rows, to_be_deleted)
```

Having identified the invalid data indices we now remove them from the row list using the above defined function.

1.6 Writing a New CSV File

```
# Save to a new csv file
with open('./used_cars_filtered.csv', 'w') as f:
    write = csv.writer(f)
    write.writerow(header)
    write.writerows(rows)
```

Here we create a new file called "used_cars_filtered.csv" where we report the column names and all the rows we have already selected and cleaned up.

STEP 2: DATA ANALYSIS AND VISUALIZATION

We wanted to analyze for a given type of vehicle whether the price of diesel is greater than the one of benzine. We created a histogram that provides a representation of the average price of the used cars based on fuel types.

We can see on the plot that the price of diesel cars is higher than that of benzine.

2.1 Initialization of Dictionaries for Data Visualization

```
benzine_vehicles = {}
diesel_vehicles = {}
for r in rows:
    if r[index_VT] not in list(benzine_vehicles.keys()) and r[index_VT] not in list(diesel_vehicles.keys())
        benzine_vehicles[r[index_VT]] = [0, 0]
        diesel_vehicles[r[index_VT]] = [0, 0]
    if r[index_fuel_type] == "diesel":
        diesel_vehicles[r[index_VT]] = [diesel_vehicles[r[index_VT]][0] + 1,
                                        diesel_vehicles[r[index_VT]][1] + int(r[index_price])]
    elif r[index_fuel_type] == "benzin":
        benzine_vehicles[r[index_VT]] = [benzine_vehicles[r[index_VT]][0] + 1,
                                         benzine_vehicles[r[index_VT]][1] + int(r[index_price])]
vehicle_types = list(benzine_vehicles.keys())
```

We create two empty dictionaries one for diesel vehicles and one for gasoline vehicles. If the vehicle type is not already present in the dictionaries we initialize the count to [0,0] where the first element represents the number of vehicles and the second element represents the total price. This is done by dividing into two groups based on the fuel used.

2.2 Calculation of the Average Price

```
benzine_average = []
for k, v in benzine_vehicles.items():
    benzine_average.append(v[1]/v[0])

diesel_average = []
for k, v in diesel_vehicles.items():
    diesel_average.append(v[1]/v[0])
```

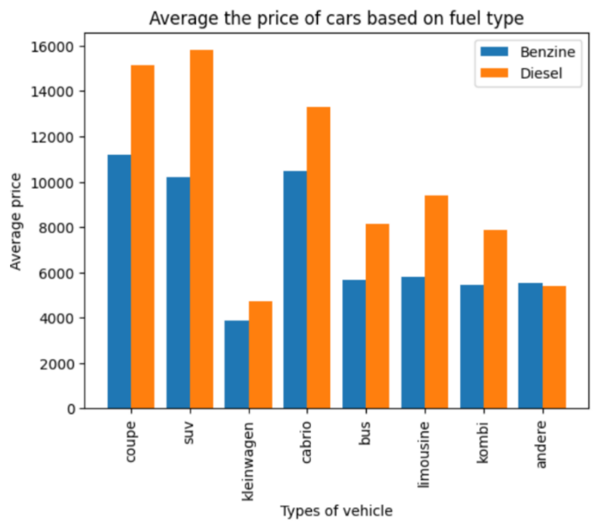
We average the cost of gasoline and diesel vehicles for each type of car. We want to answer the question: how much more expensive on average is a SUV that runs on gasoline than one that runs on diesel?

2.3 Barplot

```
X_axis = np.arange(len(vehicle_types))

plt.bar(X_axis - 0.2, benzine_average, 0.4, label = 'Benzine')
plt.bar(X_axis + 0.2, diesel_average, 0.4, label = 'Diesel')

plt.xticks(X_axis, vehicle_types)
plt.xlabel("Types of vehicle")
plt.xticks(rotation=90)
plt.ylabel("Average price")
plt.title("Average the price of cars based on fuel type")
plt.legend()
plt.show()
```



Here we create two side-by-side bar graphs. The blue bars are for benzine vehicles while the orange bars are for diesel vehicles.

STEP 3: PRICE ANALYSIS

We created a box plot to represent the distribution of prices for the following brands: mercedes_benz, fiat, volvo, alfa_romeo and lancia. The box plot showed that the price of the Mercedes is the highest followed by Volvo, Alfa Romeo, Lancia and Fiat.

3.1 Data Initialization

```python
import seaborn as sns
brands = ["mercedes_benz", "fiat", "volvo", "alfa_romeo", "lancia"]
all_data = []

with open("./used_cars_filtered.csv", encoding = "ISO-8859-1") as file:
    csv_reader = csv.reader(file)
    header = next(csv_reader)
    rows = []
    for row in csv_reader:
        rows.append(row)
    index_brand = header.index('brand')
    index_price = header.index('price')
```

Here we define a list of brands of interest calleda "brands" and then initialize a list with the goal of containing the iltracted data inside it. We continue by opening the CSV file " used_cars_filtered.csv" and reading the data inside it through the "for" loop. Finally we extract the indices of the columns of interest namely "brand" and "price".

3.2 Filtering data for each brand and collecting prices

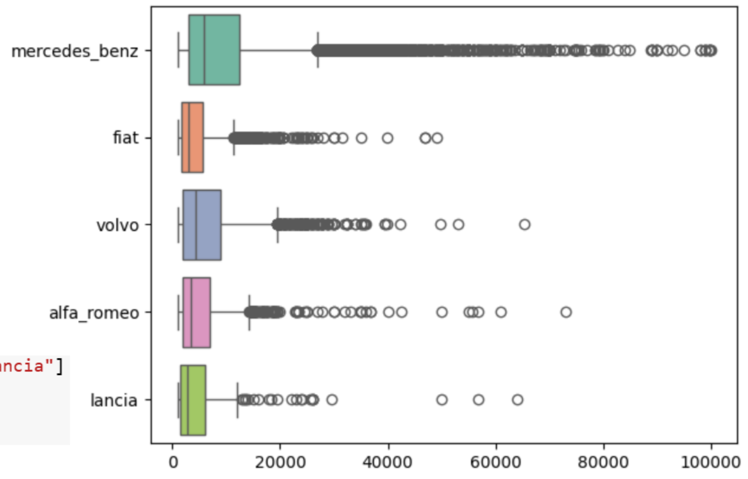
```python
for b in brands:
    tmp = []
    for r in rows:
        if r[index_brand] == b:
            tmp.append(int(r[index_price]))
    all_data.append(tmp)
```

This "for" loop begins by iterating through each of the brands in the "brands" list. Then we create an empty list needed to contain the car prices for a given brand. With the second "for" loop, on the other hand, we analyze each row of car data by checking the brands. In conclusion we add the pretto our "tmp" list and then add it to the larger list called "all data".

3.3 Boxplot

Through the command "sns.boxplot" we create a boxplot in horizontal "h" and name the x- and y-axes. The boxplot shows the distribution of used car prices for five different brands: Mercedes-Benz, Fiat, Volvo, Alfa Romeo and Lancia. Each boxplot represents a brand and provides a visualization of the key statistics of car prices for that brand. This visualization is useful for understanding the range of prices and the presence of any outliers among different brands of used cars.

```python
brands = ["mercedes_benz", "fiat", "volvo", "alfa_romeo", "lancia"]
ax = sns.boxplot(data=all_data, orient="h", palette="Set2")
ax.set_yticklabels(brands)
plt.show()
```



STEP 4: Additional Data Analysis

We created a tabular representation of price ranges, per brand, year of registration (only consider the interval 1960-2020 and create bins), and powerful/not powerful (above/below median powerPS of the entire dataset). The table shows the different brands as rows, and powerful/not powerful, years of registration as hierarchical columns. The value of a cell represents the range of prices for the cars with the brand identified by the row, interval of registration year, and powerful/not powerful identified by the column. Then we created a dataframe with the 3 most frequent models for each brand.

```python
import pandas as pd

df_cars = pd.read_csv('used_cars_filtered.csv')
median = df_cars['powerPS'].median()

df_cars['is_powerful'] = df_cars['powerPS'] > median
all_cars = df_cars[(df_cars.yearOfRegistration > 1960) & (df_cars.yearOfRegistration < 2020)]
bins = pd.cut(all_cars.yearOfRegistration, range(1960, 2021, 15))
all_cars.pivot_table(index='brand', columns=['is_powerful', bins], values=['price'], aggfunc=lambda a: (a.min(), a.max()))
```

The first step in creating the table is to calculate the median of the "powerPS" column in the "median" variable representing engine power. Next, a new column is created to identify the most powerful cars, through boolean values "True" or "False." The value "True" results if the power of a car is greater than the median calculated in the previous line of code. Continuing on, we selected cars registered in the years between 1960 and 2020 and then divide the data into groups comprising a range of 15 years of registration. Having obtained all the necessary data we can now create a pivot table that within it shows the maximum and minimum price of cars for each make divided by more and less powerful cars and by 15-year intervals. The objective is to be able to see how prices vary by engine power and registration period.

STEP 5: Visualization

For the price range 0 – 100’000 and the year of registration 1960 – 2020, we splitted the price range into 10 bins and for each bin we showed the distribution of the year of registration with a box plot.

```python
import seaborn as sns

all_cars_price = df_cars[(df_cars.price > 0) & (df_cars.price <= 100000) & (df_cars.yearOfRegistration > 1960)
                         & (df_cars.yearOfRegistration < 2020)]
all_cars_price = all_cars_price.assign(bins_price = pd.cut(all_cars_price.price, 10))

ax = sns.boxplot(data=all_cars_price, orient="h", palette="Set2", y="bins_price", x="yearOfRegistration" )
```

We further filter the dataframe into a new dataframe called "all_cars_price" in which we want only cars with a price between 0 and 100'000 and registered between 1960 and 2020 excluded. Then we add a new column to split the prices into 10 equal groups (bins). At this point we can create our boxplot, again horizontally, showing the distribution of car registration years within different price ranges. In this way we can visually see how the registration years of the cars vary between the different price ranges.

STEP 6: Aggregation table

```python
bins_kilometer = pd.cut(df_cars.kilometer, 6)

df_cars.groupby(["vehicleType", bins_kilometer]).size().unstack()
```

We split the "kilometer" column of the DataFrame "df_cars" into 6 intervals and assign each car to one of these intervals based on the kilometers driven. Now we group the data ("groupby") according to the type of vehicle and what kilometer interval they are in, this is done with the goal of creating a table that represents the distribution of the number of cars for each type of vehicle and kilometer interval so that we can see what type of vehicles fall into what specific kilometer intervals. A cell represents the number of cars with the kilometer in the range identified by the column, and the type of vehicle identified by the row. The use of the table is to be able to analyze the distribution of miles driven. For example, we see that vehicles that compared within each category are usually sold on ebay with much higher mileage.