**Analysis-3 :**

1. **The memory usage of the data is around 6.1 mb.How can we reduce the memory usage of the data set?**

* datac.drop(["nrOfPictures"],axis=1,inplace=True)

datac.drop(["dateCrawled"],axis=1,inplace=True)

datac.drop(["dateCreated"],axis=1,inplace=True)

datac.drop(["postalCode"],axis=1,inplace=True)

datac.drop(["lastSeen"],axis=1,inplace=True)

datac["price"]=datac["price"].astype(np.int32)

datac["kilometer"]=datac["kilometer"].astype(np.int32)

datac["monthOfRegistration"]=datac["monthOfRegistration"].astype(np.int8)

datac.info()

datac.columns

* Am removing several columns from the "datac" dataset using the `drop()` function with the `inplace=True` parameter. The columns being dropped are "nrOfPictures", "dateCrawled", "dateCreated", "postalCode", and "lastSeen".
* After that, am changing the data types of the "price", "kilometer", and "monthOfRegistration" columns using `astype()` from the NumPy library. You are converting "price" and "kilometer" to `np.int32` and "monthOfRegistration" to `np.int8`. This helps optimize memory usage and ensures the data is stored in the appropriate format.
* Finally, checking the dataset's information with `datac.info()` to get an overview of its structure. You are also printing the column names using `datac.columns` to see the remaining columns after the ones dropped.

**2. What is the Average price of vehicle by fuel type and gearbox type.Give a plot.**

* Using the code :

index\_names = datac[datac['price']>2000000 ].index

datac=datac.drop(index\_names,inplace=True)

datac

avg\_price=np.round((datac.groupby(["fuelType","gearbox"])["price"].mean()),2)

avg\_price.plot(kind="bar")

plt.xlabel("fuelType and gearbox type")

plt.ylabel("average price")

plt.title("avg price of fuelType and gearbox")

plt.show()

* First, am creating a new variable called "index\_names" to store the indices of rows where the "price" column is greater than 2000000. Then, am dropping those rows from the "datac" dataset using the `drop()` function with `index\_names` as the argument.
* Afterwards, am calculating the average price of vehicles based on the "fuelType" and "gearbox" columns using the `groupby()` and `mean()` functions. It store the results in the "avg\_price" variable.
* Next, I plotted a bar chart using the "avg\_price" data. The x-axis represents the combination of "fuelType" and "gearbox" types, while the y-axis represents the average
* price. label the x-axis as "fuelType and gearbox type", the y-axis as "average price", and give the chart a title.

**3. What is the Average power of a vehicle by vehicle type and gearbox type.Give a plot.**

* Using this code:

avg\_power=datac.groupby(["vehicleType","gearbox"])["powerPS"].mean()

avg\_power.plot(kind="bar")

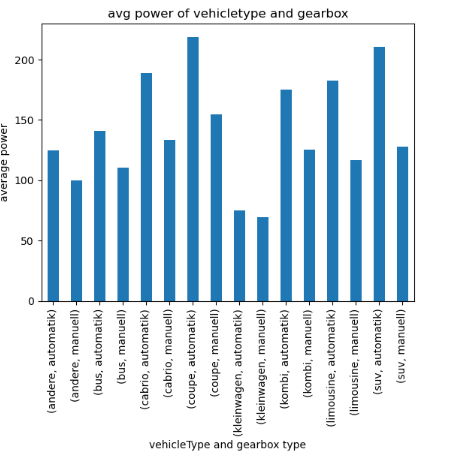
plt.xlabel("vehicleType and gearbox type")

plt.ylabel("average power")

plt.title("avg power of vehicletype and gearbox")

plt.show()

* First, am creating a new variable called "index\_names" to store the indices of rows where the "price" column is greater than 2000000. Then, am dropping those rows from the "datac" dataset using the `drop()` function with `index\_names` as the argument.
* Afterwards, am calculating the average price of vehicles based on the "fuelType" and "gearbox" columns using the `groupby()` and `mean()` functions. It store the results in the "avg\_price" variable.
* Next, I plotted a bar chart using the "avg\_price" data. The x-axis represents the combination of "fuelType" and "gearbox" types, while the y-axis represents the average price. label the x-axis as "fuelType and gearbox type", the y-axis as "average price", and give the chart a title.



**4. What is the Average price of a vehicle by brand as well as vehicle type.Use heatmap to explain this.**

* Using this code :
* datax=datac.groupby(["vehicleType","brand"])

datax["price"].mean()

avg\_price=pd.DataFrame(pd.crosstab(index=datac["brand"],columns=datac["vehicleType"],values=datac["price"],aggfunc=["mean"]))

avg\_price.fillna(0,inplace=True)

avg\_price

* In the code I have a DataFrame called "datac" which contains information about vehicles, including the vehicle type, brand, and price.
* The first line of code uses the "groupby" function to group the data based on the "vehicleType" and "brand" columns. This creates groups of data where each group represents a unique combination of vehicle type and brand.
* The second line of code calculates the mean (average) price for each group using the "mean" function. This gives the average price of vehicles for each combination of vehicle type and brand.
* The third line of code creates a new DataFrame called "avg\_price" using the "pd.DataFrame" function. It uses the "pd.crosstab" function to create a cross-tabulation of the "brand" and "vehicleType" columns.