ASSIGNMENT – 2

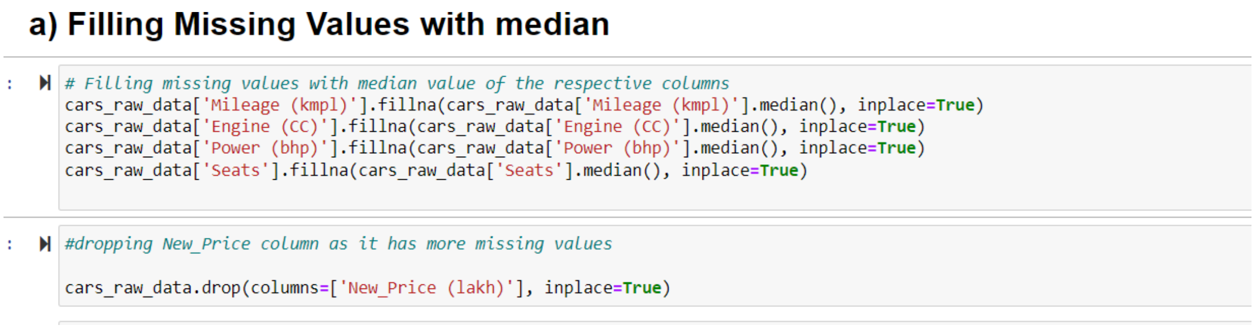
Name: Teja Praveen Kumar Kondaveeti

Student ID: 18250776

Answers:

a)

In this task, identified missing values in columns Mileage (2), Engine (36), Power (36), Seats (38) and New_Price (5032). Here New_Price has more missing values that's why dropped New_Price column from the data frame. As Mileage, Engine, Power, and Seats has less missing values count imputed them with "median" value. Imputing missing values with median is a robust approach that works well for various types of data distributions and is less sensitive to outliers compared to imputation with mean.



a) Filling Missing Values with median

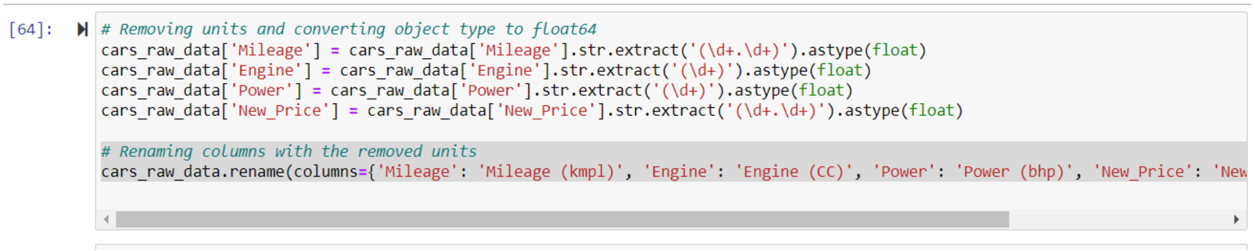
```
# Filling missing values with median value of the respective columns
cars_raw_data['Mileage (kmpl)'].fillna(cars_raw_data['Mileage (kmpl)'].median(), inplace=True)
cars_raw_data['Engine (CC)'].fillna(cars_raw_data['Engine (CC)'].median(), inplace=True)
cars_raw_data['Power (bhp)'].fillna(cars_raw_data['Power (bhp)'].median(), inplace=True)
cars_raw_data['Seats'].fillna(cars_raw_data['Seats'].median(), inplace=True)
```

```
#dropping New_Price column as it has more missing values

cars_raw_data.drop(columns=['New_Price (lakh)'], inplace=True)
```

b)

In this task, removed the units kmpl, CC, bhp, and lakh from the respective attributes Mileage, Engine, Power, and New_Price. And renamed columns with the removed units.

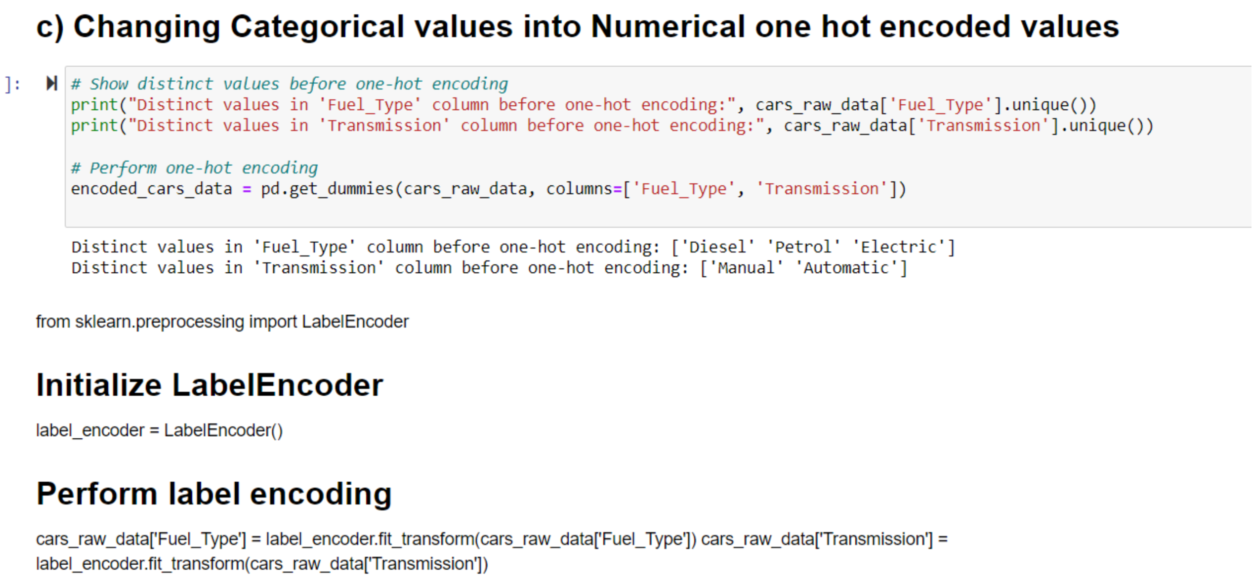


```
[64]:
# Removing units and converting object type to float64
cars_raw_data['Mileage'] = cars_raw_data['Mileage'].str.extract('(\d+.\d+)').astype(float)
cars_raw_data['Engine'] = cars_raw_data['Engine'].str.extract('(\d+)').astype(float)
cars_raw_data['Power'] = cars_raw_data['Power'].str.extract('(\d+)').astype(float)
cars_raw_data['New_Price'] = cars_raw_data['New_Price'].str.extract('(\d+.\d+)').astype(float)

# Renaming columns with the removed units
cars_raw_data.rename(columns={'Mileage': 'Mileage (kmpl)', 'Engine': 'Engine (CC)', 'Power': 'Power (bhp)', 'New_Price': 'New
```

c)

In this task, changed the categorical variables Fuel Type and Transmission into numerical using One-hot encoding.



c) Changing Categorical values into Numerical one hot encoded values

```
# Show distinct values before one-hot encoding
print("Distinct values in 'Fuel_Type' column before one-hot encoding:", cars_raw_data['Fuel_Type'].unique())
print("Distinct values in 'Transmission' column before one-hot encoding:", cars_raw_data['Transmission'].unique())

# Perform one-hot encoding
encoded_cars_data = pd.get_dummies(cars_raw_data, columns=['Fuel_Type', 'Transmission'])
```

```
Distinct values in 'Fuel_Type' column before one-hot encoding: ['Diesel' 'Petrol' 'Electric']
Distinct values in 'Transmission' column before one-hot encoding: ['Manual' 'Automatic']
```

from sklearn.preprocessing import LabelEncoder

Initialize LabelEncoder

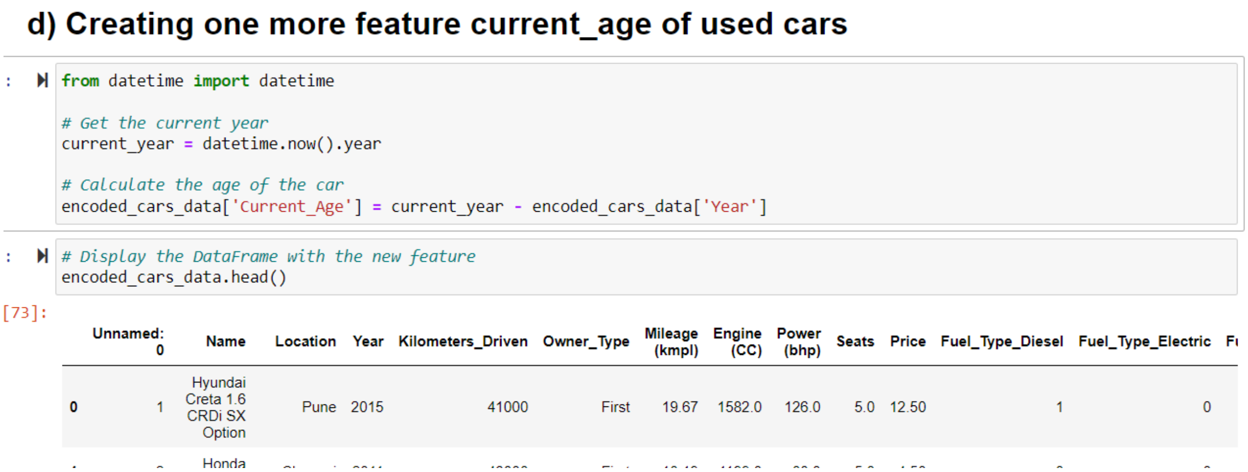
label_encoder = LabelEncoder()

Perform label encoding

cars_raw_data['Fuel_Type'] = label_encoder.fit_transform(cars_raw_data['Fuel_Type']) cars_raw_data['Transmission'] = label_encoder.fit_transform(cars_raw_data['Transmission'])

d)

Here, created one more feature Current_Age using current_age = current_year – Year and added to the dataset.



d) Creating one more feature current_age of used cars

```
from datetime import datetime

# Get the current year
current_year = datetime.now().year

# Calculate the age of the car
encoded_cars_data['Current_Age'] = current_year - encoded_cars_data['Year']
```

```
# Display the DataFrame with the new feature
encoded_cars_data.head()
```

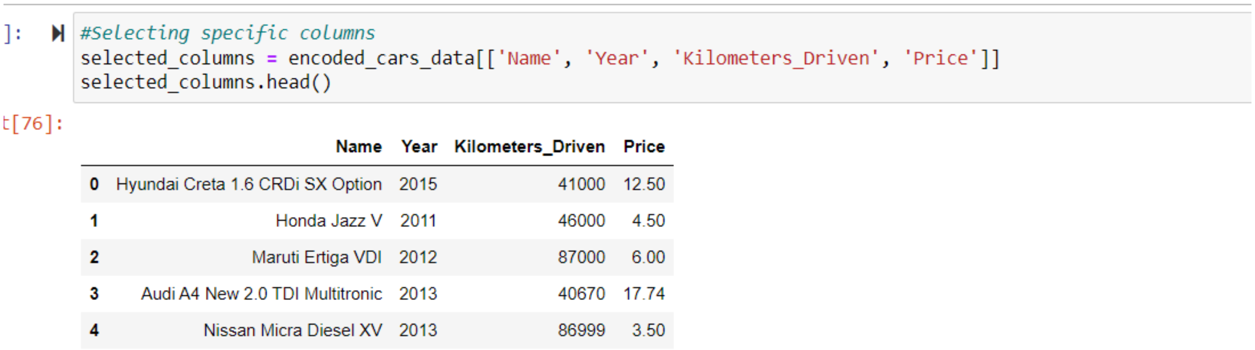
[73]:

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Owner_Type	Mileage (kmpl)	Engine (CC)	Power (bhp)	Seats	Price	Fuel_Type_Diesel	Fuel_Type_Electric	Fu
0	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	First	19.67	1582.0	126.0	5.0	12.50	1	0	
[illegible]	[illegible]	Honda	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	

e)

Performed select, filter, rename, mutate, arrange and summarize with group by on the dataset.

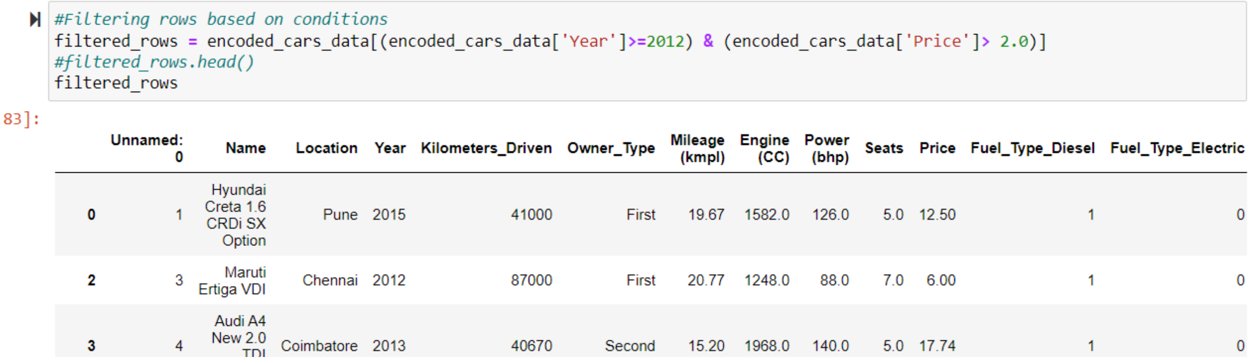
e) Performing select, filter, rename, mutate, arrange and summarize operations



```
]:  #Selecting specific columns
    selected_columns = encoded_cars_data[['Name', 'Year', 'Kilometers_Driven', 'Price']]
    selected_columns.head()
```

t[76]:

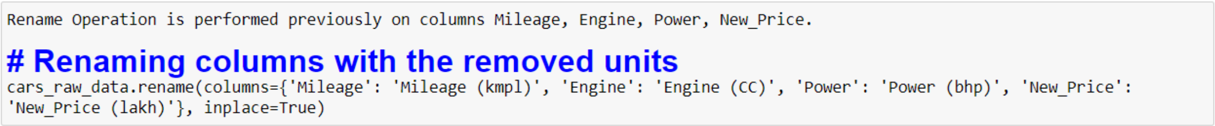
	Name	Year	Kilometers_Driven	Price
0	Hyundai Creta 1.6 CRDi SX Option	2015	41000	12.50
1	Honda Jazz V	2011	46000	4.50
2	Maruti Ertiga VDI	2012	87000	6.00
3	Audi A4 New 2.0 TDI Multitronic	2013	40670	17.74
4	Nissan Micra Diesel XV	2013	86999	3.50



```
#Filtering rows based on conditions
filtered_rows = encoded_cars_data[(encoded_cars_data['Year']>=2012) & (encoded_cars_data['Price']> 2.0)]
#filtered_rows.head()
filtered_rows
```

83]:

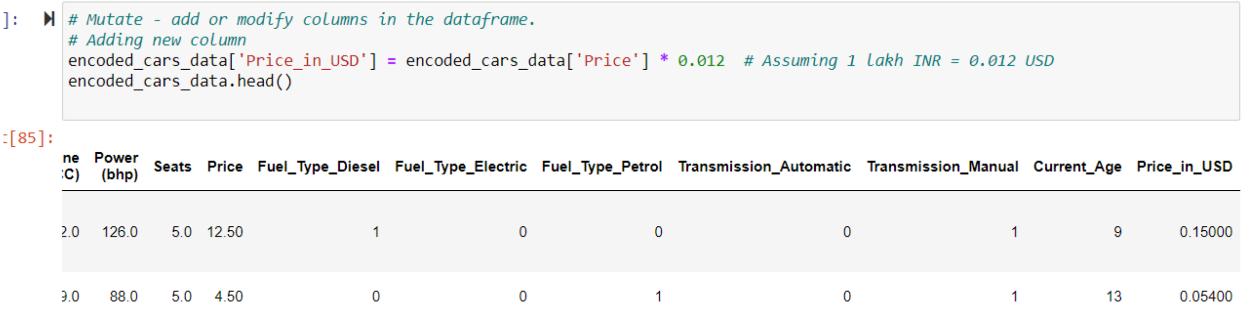
	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Owner_Type	Mileage (kmpl)	Engine (CC)	Power (bhp)	Seats	Price	Fuel_Type_Diesel	Fuel_Type_Electric
0	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	First	19.67	1582.0	126.0	5.0	12.50	1	0
2	3	Maruti Ertiga VDI	Chennai	2012	87000	First	20.77	1248.0	88.0	7.0	6.00	1	0
3	4	Audi A4 New 2.0 TDI	Coimbatore	2013	40670	Second	15.20	1968.0	140.0	5.0	17.74	1	0



Rename Operation is performed previously on columns Mileage, Engine, Power, New_Price.

Renaming columns with the removed units

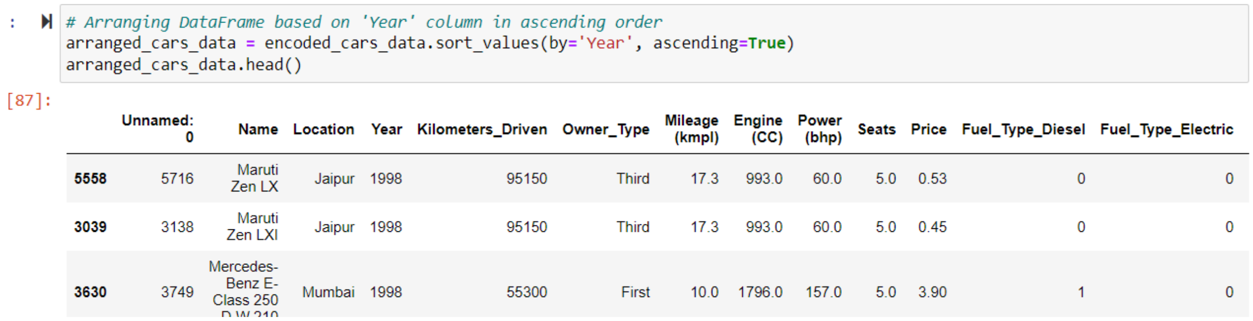
```
cars_raw_data.rename(columns={'Mileage': 'Mileage (kmpl)', 'Engine': 'Engine (CC)', 'Power': 'Power (bhp)', 'New_Price':
'New_Price (lakh)'}, inplace=True)
```



```
]:  # Mutate - add or modify columns in the dataframe.
    # Adding new column
    encoded_cars_data['Price_in_USD'] = encoded_cars_data['Price'] * 0.012  # Assuming 1 lakh INR = 0.012 USD
    encoded_cars_data.head()
```

:[85]:

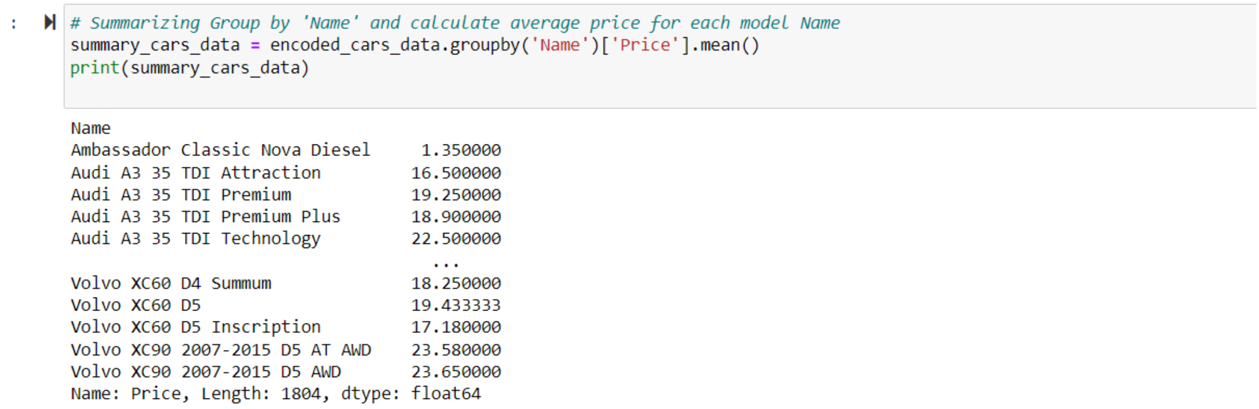
ne C)	Power (bhp)	Seats	Price	Fuel_Type_Diesel	Fuel_Type_Electric	Fuel_Type_Petrol	Transmission_Automatic	Transmission_Manual	Current_Age	Price_in_USD
2.0	126.0	5.0	12.50	1	0	0	0	1	9	0.15000
9.0	88.0	5.0	4.50	0	0	1	0	1	13	0.05400



```
# Arranging DataFrame based on 'Year' column in ascending order
arranged_cars_data = encoded_cars_data.sort_values(by='Year', ascending=True)
arranged_cars_data.head()
```

[87]:

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Owner_Type	Mileage (kmpl)	Engine (CC)	Power (bhp)	Seats	Price	Fuel_Type_Diesel	Fuel_Type_Electric
5558	5716	Maruti Zen LX	Jaipur	1998	95150	Third	17.3	993.0	60.0	5.0	0.53	0	0
3039	3138	Maruti Zen LXI	Jaipur	1998	95150	Third	17.3	993.0	60.0	5.0	0.45	0	0
3630	3749	Mercedes-Benz E-Class 250 D W 210	Mumbai	1998	55300	First	10.0	1796.0	157.0	5.0	3.90	1	0



```
# Summarizing Group by 'Name' and calculate average price for each model Name
summary_cars_data = encoded_cars_data.groupby('Name')['Price'].mean()
print(summary_cars_data)
```

```
Name
Ambassador Classic Nova Diesel     1.350000
Audi A3 35 TDI Attraction         16.500000
Audi A3 35 TDI Premium            19.250000
Audi A3 35 TDI Premium Plus       18.900000
Audi A3 35 TDI Technology         22.500000
                                    ...
Volvo XC60 D4 Summum              18.250000
Volvo XC60 D5                     19.433333
Volvo XC60 D5 Inscription         17.180000
Volvo XC90 2007-2015 D5 AT AWD    23.580000
Volvo XC90 2007-2015 D5 AWD       23.650000
Name: Price, Length: 1804, dtype: float64
```