

Results

A) Look for the missing values in all the columns and either impute them (replace with mean, median, or mode) or drop them. Justify your action for this task.

1. ****Verify Any Missing Data:****

Each column's missing values are counted using the formula `{missing_values = car_data.isnull().sum()}`.

- The count of missing values for each column is shown by using `{print(missing_values)}`.

2. ****Find Missing Values in Columns:****

- The list "missing_columns" is generated empty.
- It adds the missing values to the list by iteratively going over the columns.

3. ****Impute Missing Values:**** 'missing_columns' is iterated through.

- Uses the mode to fill in the missing values for object-type columns.
- Uses the mean to fill in any missing values in non-object columns.

4. ****Display Details about the Dataset:**** - Post-imputation dataset details (non-null counts, data types, memory utilization) are displayed using ``car_data.info()`.

Essentially, this function provides insights into the updated dataset and assures completeness by substituting acceptable measures for missing values.

Files | PDS_Assignment-2/ at main | colab.google | PDS_Ass-2 - Colaboratory

colab.research.google.com/drive/1Hj1cuY70OebtV0Y9spvSqVYoBT3qSJZ6#scrollTo=82437aa0

PDS_Ass-2

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Files

- sample_data
- clean_cars.csv
- train.csv

Unamed: 0

		Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Se
0	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67 kmpl	1582 CC	126.2 bhp	
1	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	13 km/kg	1199 CC	88.7 bhp	
2	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77 kmpl	1248 CC	88.76 bhp	
3	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC	140.8 bhp	
4	6	Nissan Micra Diesel XV	Jaipur	2013	86999	Diesel	Manual	First	23.08 kmpl	1461 CC	63.1 bhp	
...
5842	6014	Maruti Swift VDI	Delhi	2014	27365	Diesel	Manual	First	28.4 kmpl	1248 CC	74 bhp	
5843	6015	Hyundai Xcent 1.1 CRDi S	Jaipur	2015	100000	Diesel	Manual	First	24.4 kmpl	1120 CC	71 bhp	
5844	6016	Mahindra Xylo D4 BSIV	Jaipur	2012	55000	Diesel	Manual	Second	14.0 kmpl	2498 CC	112 bhp	
5845	6017	Maruti Wagon R VXI	Kolkata	2013	46000	Petrol	Manual	First	18.9 kmpl	998 CC	67.1 bhp	

Disk 81.40 GB available

completed at 8:08 PM

Files | PDS_Assignment-2/ at main | colab.google | PDS_Ass-2 - Colaboratory

colab.research.google.com/drive/1Hj1cuY70OebtV0Y9spvSqVYoBT3qSJZ6#scrollTo=82437aa0

PDS_Ass-2

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Files

- sample_data
- clean_cars.csv
- train.csv

```
print(columns_with_missing_values, missing_columns)

Unamed: 0
Name
Location
Year
Kilometers_Driven
Fuel_Type
Transmission
Owner_Type
Mileage
Engine
Power
Seats
New_Price
Price
dtype: int64
Columns with missing values: ['Mileage', 'Engine', 'Power', 'Seats', 'New_Price']

[9] for col in missing_columns:
    if car_data[col].dtype == 'object':
        car_data[col].fillna(car_data[col].mode()[0], inplace=True)
    else:
        car_data[col].fillna(car_data[col].mean(), inplace=True)

[10] car_data.info
```

pandas.core.frame.DataFrame.info

def info(verbose: bool | None=None, buf: WriteBuffer[str] | None=None, max_cols: int | None=None, memory_usage: bool | str | None=None, show_counts: bool | None=None, null_counts: bool | None=None) -> None

</usr/local/lib/python3.10/dist-packages/pandas/core/frame.py>

Print a concise summary of a DataFrame.

This method prints information about a DataFrame including the index dtype and columns, non-null values and memory usage.

Disk 81.40 GB available

completed at 8:08 PM

B) Remove the units from some of the attributes and only keep the numerical values (for example remove kmpl from “Mileage”, CC from “Engine”, bhp from “Power”, and lakh from “New_price”).

1. ****Exploration of Data Types****

- Each column's data types are captured by ``column_data_typ = car_data.dtypes``.
- The data types are displayed via ``print(column_data_typ)``.

2. ****Conversion of Data Type:**** - Using ``astype(str)``, the selected columns ('Mileage, Engine, Power, and New Price') are transformed to string type.

3. ****Extract Numerical Values:**** - Using a regular expression, numerical values are taken out of the strings and transformed to float for the 'Mileage' column.

4. ****Clean "Engine" Column:**** - "CC" is removed and "Engine" values are converted to numeric values. Erroneous inputs are forced to NaN.

5. ****Handle Missing 'Engine' Values:**** - If a value is missing in the 'Engine' column, it is changed to integer type and supplied with a default value (0).

6. ****Extract Numerical Values for 'Power', 'New_Price':**** - A regular expression is used to extract numerical values from the 'Power' and 'New_Price' columns, which are then converted to float.

7. ****Final DataFrame:**** - The updated 'car_data' DataFrame is shown, displaying the value extractions and data type conversions that were made.

Files | PDS_Assignment-2/ at main | colab.google | PDS_Ass-2 - Colaboratory

colab.research.google.com/drive/1Hj1cuY70OebtV0Y9spvSqVYoBT3qSJZ6#scrollTo=4aa250ea

PDS_Ass-2

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Colab AI

Files

- sample_data
- clean_cars.csv
- train.csv

+ Code + Text

```
[11] Unnamed: 0      int64
      Name        object
      Location    object
      Year        int64
      Kilometers_Driven int64
      Fuel_Type    object
      Transmission object
      Owner_Type   object
      Mileage      object
      Engine       object
      Power        object
      Seats        float64
      New_Price    object
      Price        float64
      dtype: object
```

car_data

Unnamed: 0		Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Se
0	1	Hyundai Creta 1.6 CRDI SX Option	Pune	2015	41000	Diesel	Manual	First	19.67 kmpl	1582 CC	126.2 bhp	
1	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	13 km/kg	1199 CC	88.7 bhp	
2	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77 kmpl	1248 CC	88.76 bhp	
3	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC	140.8 bhp	
		Nissan							23.08	1461	63.1	

0s completed at 8:08 PM

Files | PDS_Assignment-2/ at main | colab.google | PDS_Ass-2 - Colaboratory

colab.research.google.com/drive/1Hj1cuY70OebtV0Y9spvSqVYoBT3qSJZ6#scrollTo=4aa250ea

PDS_Ass-2

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Colab AI

Files

- sample_data
- clean_cars.csv
- train.csv

+ Code + Text

car_data

Unnamed: 0		Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Se
0	1	Hyundai Creta 1.6 CRDI SX Option	Pune	2015	41000	Diesel	Manual	First	19.67	1582	126.20	
1	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	NaN	1199	88.70	
2	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77	1248	88.76	
3	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.20	1968	140.80	
4	6	Nissan Micra Diesel XV	Jaipur	2013	86999	Diesel	Manual	First	23.08	1461	63.10	
...
5842	6014	Maruti Swift VDI	Delhi	2014	27365	Diesel	Manual	First	28.40	1248	NaN	
5843	6015	Hyundai Xcent 1.1 CRDI S	Jaipur	2015	100000	Diesel	Manual	First	24.40	1120	NaN	
5844	6016	Mahindra Xylo D4 BSIV	Jaipur	2012	55000	Diesel	Manual	Second	14.00	2498	112.00	
5845	6017	Maruti Wagon R	Kolkata	2013	46000	Petrol	Manual	First	18.90	998	67.10	

0s completed at 8:08 PM

C)Change the categorical variables (“Fuel_Type” and “Transmission”) into numerical one hot encoded value.

1. ****Encoding of Labels:****

Makes use of `LabelEncoder()` to convert the 'Transmission' and 'Fuel_Type' columns into numerical labels.

- Inserts the encoded labels into the newly created columns "Transmission_Label" and "Fuel_Type_Label."

2. ****Encoding One-Hot:****

- Uses `OneHotEncoder()` to create binary vectors from the label-encoded columns.

Makes two distinct DataFrames ('transmission_encoded_df' and 'fuel_type_encoded_df') for every new binary vector.

3. ****Concatenation:**** - Merges the one-hot encoded DataFrames with the original 'car_data' DataFrame.

4. **Eliminate Superfluous Columns:**

- Eliminates the label-encoded columns ('Fuel_Type_Label', 'Transmission_Label') and the original categorical columns ('Fuel_Type', 'Transmission').

5. ****Final DataFrame:**** - Shows the updated "car_data" DataFrame with one-hot encoded representations of "Transmission" and "Fuel Type."

Unnamed: 0	Name	Location	Year	Kilometers_Driven	Owner_Type	Mileage	Engine	Power	Seats	New_Price	Price	Fu
0	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	First	19.67	1582	126.20	5.0	4.78	12.50	
1	Honda Jazz V	Chennai	2011	46000	First	NaN	1199	88.70	5.0	8.61	4.50	
2	Maruti Eriga VDI	Chennai	2012	87000	First	20.77	1248	88.76	7.0	4.78	6.00	
3	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Second	15.20	1968	140.80	5.0	4.78	17.74	
4	Nissan Micra Diesel XV	Jaipur	2013	86999	First	23.08	1461	63.10	5.0	4.78	3.50	
...
5842	Maruti Swift VDI	Delhi	2014	27365	First	28.40	1248	NaN	5.0	7.88	4.75	
5843	Hyundai Xcent 1.1 CRDi S	Jaipur	2015	100000	First	24.40	1120	NaN	5.0	4.78	4.00	
5844	Mahindra Xylo D4 BSIV	Jaipur	2012	55000	Second	14.00	2498	112.00	8.0	4.78	2.90	
5845	Maruti Wagon R VXI	Kolkata	2013	46000	First	18.90	998	67.10	5.0	4.78	2.85	

D) Create one more feature and add this column to the dataset (you can use mutate function in R for this). For example, you can calculate the current age of the car by subtracting “Year” value from the current year.

1. ****Current Year Calculation:**** - Using the `datetime` module, `curr_year = datetime.now().year` retrieves the current year.

2. ****Determine Age of Car:****

Each car's age is calculated by deducting its production year (found in the 'Year' column) from the current year using the formula `{car_data['Current_Age'] = curr_year - car_data['Year']}`.

3. ****Result:**** - The age of each automobile is now displayed in a new column called "Current_Age" in the updated "car_data" DataFrame.

The screenshot shows a Google Colab notebook interface. The left sidebar displays the file explorer with a folder named 'sample_data' containing 'clean_cars.csv' and 'train.csv'. The main area shows a code cell with the following Python code:

```
curr_year = datetime.now().year
car_data['Current_Age'] = curr_year - car_data['Year']
car_data
```

The output of the code is a DataFrame with the following columns: Unnamed: 0, Name, Location, Year, Kilometers_Driven, Owner_Type, Mileage, Engine, Power, Seats, New_Price, Price, Fuel_Charge. The data includes rows for various car models like Hyundai Creta, Honda Jazz, Maruti Ertiga, Audi A4, Nissan Micra, Maruti Swift, Hyundai Xcent, and Mahindra Xylo.

E) Perform select, filter, rename, mutate, arrange and summarize with group by operations (or their equivalent operations in python) on this dataset.

1. **Choosing a Column:**

- {car_data = select_data[['Year, 'Name,' Price, 'Engine,' Location]]Name, Year, Price, Engine, and Location are the columns that are chosen, and they are then saved in a new DataFrame by {.

****Data Filtering:**** - The dataset is filtered to include only rows where the 'Location' is 'Chennai' using the formula {filt_data = car_data[car_data['Location'] == 'Chennai']}.}

3. **Renaming of Columns:**

- The 'New Price' column is renamed to 'NewPriceCAR' by running the command `car_data = car_data.rename(columns={'New_Price': 'NewPriceCAR'})`.

- {car_data['NewIncreseprice'] = car_data['NewPriceCAR'] 4 **Column
Operation:**Column 'NewPriceCAR' is multiplied by three to create a new column
'NewIncreseprice' using the formula *3{.

5. **Data Sorting:**

- The dataset is sorted in ascending order using the 'Location' column by
sort_car_data = car_data.sort_values(by='Location', ascending=True)}.

6. **Aggregation and Grouping:**

- {grouped_summary = car_data.groupby('Year').The data is grouped by 'Year'
and the average of the 'Mileage' for each group is computed using the formula
agg(Average_MPG=('Mileage','mean'))}.

7. **CSV Saving:**

The adjusted 'car_data' DataFrame is saved to a CSV file called 'clean_cars.csv'
with the command `car_data.to_csv('clean_cars.csv', index=False)`.

```

filt_data = car_data[car_data['Location'] == 'Chennai'] #filtered Data
print(filt_data)

```

	Unnamed: 0	Name	Location	Year
1	2	Honda Jazz V	Chennai	2011
2	3	Maruti Ertiga VDI	Chennai	2012
7	9	Tata Indica Vista Quadrajet LS	Chennai	2012
37	39	Volkswagen Polo Diesel Trendline 1.2L	Chennai	2013
52	54	Hyundai Grand i10 Sportz	Chennai	2015
...
5762	5930	Maruti Swift Dzire VXI Optional	Chennai	2015
5764	5932	Maruti Alto 800 2016-2019 LXI	Chennai	2016
5784	5954	Toyota Fortuner 4x4 AT	Chennai	2015
5786	5956	Toyota Innova 2.5 GX (Diesel) 7 Seater	Chennai	2012
5827	5999	Tata Bolt Revotron XT	Chennai	2016

	Kilometers_Driven	Owner_Type	Mileage	Engine	Power	Seats	New_Price
1	46000	First	NaN	1199	88.70	5.0	8.61
2	87000	First	20.77	1248	88.76	7.0	4.78
7	65932	Second	22.30	1248	NaN	5.0	4.78
37	62000	First	22.07	1199	73.90	5.0	4.78
52	54000	Third	18.90	1197	NaN	5.0	4.78
...
5762	49900	First	20.85	1197	83.14	5.0	4.78
5764	57000	First	24.70	796	47.30	5.0	4.78
5784	95000	First	12.55	2982	168.50	7.0	4.78
5786	96000	First	12.99	2494	100.00	7.0	4.78
5827	10000	First	17.57	1193	88.70	5.0	7.77

	Price	Fuel_Type_0	Fuel_Type_1	Fuel_Type_2	Transmission_0
1	4.50	0.0	0.0	1.0	0.0
2	6.00	1.0	0.0	0.0	0.0
7	1.95	1.0	0.0	0.0	0.0
37	5.00	1.0	0.0	0.0	0.0
52	3.60	0.0	0.0	1.0	0.0
...
5762	4.80	0.0	0.0	1.0	0.0
5764	2.80	0.0	0.0	1.0	0.0
...

0s completed at 8:08 PM

Files | PDS_Assignment-2/ at main | colab.google | PDS_Ass-2 - Collaboratory

colab.research.google.com/drive/1Hj1cuY70OebtV0Y9spvSqVYoBT3qSJZ6#scrollTo=ede5bace

PDS_Ass-2

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Colab AI

Files

- sample_data
- clean_cars.csv
- train.csv

+ Code + Text

```
sort_car_data = car_data.sort_values(by='Location', ascending=True)
sort_car_data
```

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Owner_Type	Mileage	Engine	Power	Seats	NewPriceCAR	Price
2265	2334	Toyota Fortuner 2.8 2WD AT	Ahmedabad	2016	88008	Second	12.90	2755	174.50	7.0	34.45	22.50
5127	5276	Mercedes-Benz CLA 200 D Sport Edition	Ahmedabad	2015	13500	First	17.90	2143	136.00	5.0	4.78	25.51
2430	2511	Hyundai Grand i10 CRDI Asta	Ahmedabad	2014	116905	First	24.00	1120	NaN	5.0	4.78	4.00
4011	4140	Hyundai EON D Lite Plus	Ahmedabad	2015	20000	First	21.10	814	55.20	5.0	4.78	2.66
936	969	Skoda Rapid 1.5 TDI AT Ambition	Ahmedabad	2015	60001	First	21.66	1498	103.52	5.0	4.78	7.25
...
4507	4642	Honda City 1.5 V MT	Pune	2011	102013	First	17.00	1497	118.00	5.0	4.78	3.75
4508	4643	Toyota Fortuner 4x2 AT	Pune	2015	36000	First	12.55	2982	168.50	7.0	4.78	22.00

0s completed at 8:08 PM

Files | PDS_Assignment-2/ at main | colab.google | PDS_Ass-2 - Collaboratory

colab.research.google.com/drive/1Hj1cuY70OebtV0Y9spvSqVYoBT3qSJZ6#scrollTo=ede5bace

PDS_Ass-2

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Colab AI

Files

- sample_data
- clean_cars.csv
- train.csv

+ Code + Text

Next steps: [Generate code with sort_car_data](#) [View recommended plots](#)

```
grouped_summary = car_data.groupby('Year').agg(Average_MPG=('Mileage', 'mean'))
print(grouped_summary)
```

Year	Average_MPG
1998	14.866667
1999	15.150000
2000	15.267500
2001	14.440000
2002	15.158333
2003	14.593750
2004	13.743704
2005	14.057955
2006	15.165942
2007	14.844811
2008	15.279264
2009	15.546528
2010	16.330818
2011	16.836608
2012	17.933244
2013	18.311125
2014	18.733929
2015	19.089891
2016	19.609114
2017	19.392990
2018	19.250816
2019	17.861584

```
car_data.to_csv('clean_cars.csv', index=False)
```

0s completed at 8:08 PM

