

Name :Shreyash Kamat

Div : D15C

Roll No : 22

Experiment No. - 1 :

Aim :

- Load data in Pandas.
- Description of the dataset.
- Drop columns that aren't useful.
- Drop rows with maximum missing values.
- Take care of missing data.
- Create dummy variables.
- Find out outliers (manually)
- standardization and normalization of columns

Problem Statement : Introduction to Data science and Data preparation using Pandas steps.

Introduction :

Q.What is Data Science and Data Preparation ?

1. Data Science

Data Science is the process of extracting insights from data using statistical and computational techniques. It involves:

- **Data Processing** – Collecting, cleaning, and organizing raw data.
- **Analysis & Modeling** – Applying machine learning and statistical methods to identify patterns.
- **Decision Making** – Using data-driven insights to solve real-world problems.

2. Data Preparation

Data Preparation ensures data quality for analysis and modeling by refining raw data. It includes:

- **Cleaning** – Handling missing values, duplicates, and inconsistencies.
- **Transformation** – Normalizing, scaling, and encoding data for better model performance.
- **Feature Selection** – Choosing relevant data attributes to improve accuracy.

Dataset Used : Car features and their corresponding MSRP.

The dataset titled "Car Features and MSRP" provides detailed information on various car attributes and their corresponding Manufacturer's Suggested Retail Prices (MSRP). This dataset is valuable for analyzing how different features influence car pricing

Key Features of the Dataset:

- **Make and Model:** Identifies the manufacturer and specific model of each car.
- **Year:** Indicates the production year of the vehicle.
- **Engine Type:** Details about the engine, such as displacement and configuration.
- **Fuel Type:** Indicates the kind of fuel the car uses, such as gasoline, diesel, or electric.
- **MSRP:** Lists the Manufacturer's Suggested Retail Price for each vehicle.

This dataset is structured to facilitate analysis of how these features correlate with car pricing, making it a valuable resource for studies in automotive market trends and pricing strategies.

1.Loading Data into Pandas

```
import pandas as pd
df = pd.read_csv('Car_Features.csv')
df.info()
df.describe()
```

	Year	Engine HP	Engine Cylinders	Number of Doors	highway MPG	city mpg	Popularity	MSRP
count	11914.000000	11845.00000	11884.000000	11908.000000	11914.000000	11914.000000	11914.000000	1.191400e+04
mean	2010.384338	249.38607	5.628829	3.436093	26.637485	19.733255	1554.911197	4.059474e+04
std	7.579740	109.19187	1.780559	0.881315	8.863001	8.987798	1441.855347	6.010910e+04
min	1990.000000	55.00000	0.000000	2.000000	12.000000	7.000000	2.000000	2.000000e+03
25%	2007.000000	170.00000	4.000000	2.000000	22.000000	16.000000	549.000000	2.100000e+04
50%	2015.000000	227.00000	6.000000	4.000000	26.000000	18.000000	1385.000000	2.999500e+04
75%	2016.000000	300.00000	6.000000	4.000000	30.000000	22.000000	2009.000000	4.223125e+04
max	2017.000000	1001.00000	16.000000	4.000000	354.000000	137.000000	5657.000000	2.065902e+06

All of the data from the dataset file of 'Car_Features.csv' was loaded onto pandas and the successful loading of the file was verified by using the df.describe() command that displays the data within the file.

2.Description of the Dataset

```
import pandas as pd
df = pd.read_csv('Car_Features.csv')
df.info()
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Make                  11914 non-null  object
1   Model                 11914 non-null  object
2   Year                  11914 non-null  int64
3   Engine Fuel Type     11911 non-null  object
4   Engine HP             11845 non-null  float64
5   Engine Cylinders      11884 non-null  float64
6   Transmission Type     11914 non-null  object
7   Driven_Wheels         11914 non-null  object
8   Number of Doors       11908 non-null  float64
9   Market Category       8172 non-null   object
10  Vehicle Size          11914 non-null  object
11  Vehicle Style         11914 non-null  object
12  highway MPG           11914 non-null  int64
13  city mpg              11914 non-null  int64
14  Popularity            11914 non-null  int64
15  MSRP                  11914 non-null  int64
dtypes: float64(3), int64(5), object(8)
memory usage: 1.5+ MB
```

The `df.describe()` command is used to obtain a description of the data inside of the dataset.

3.Drop columns that are not useful.(Dropping Column "Popularity")

Dropping Column "Popularity"

```
[ ] import pandas as pd
df = pd.read_csv('Car_Features.csv')
df = df.drop('Popularity', axis=1)
df.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 15 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Make                  11914 non-null  object
1   Model                 11914 non-null  object
2   Year                  11914 non-null  int64
3   Engine Fuel Type      11911 non-null  object
4   Engine HP             11845 non-null  float64
5   Engine Cylinders      11884 non-null  float64
6   Transmission Type     11914 non-null  object
7   Driven_Wheels         11914 non-null  object
8   Number of Doors       11908 non-null  float64
9   Market Category       8172 non-null   object
10  Vehicle Size          11914 non-null  object
11  Vehicle Style         11914 non-null  object
12  highway MPG           11914 non-null  int64
13  city mpg              11914 non-null  int64
14  MSRP                  11914 non-null  int64
dtypes: float64(3), int64(4), object(8)
memory usage: 1.4+ MB
```

The column of 'Popularity' which is not really all that useful from the perspective of analysis of the data is removed from the dataset as a part of its processing phase.

4. Dropping rows with missing values

Dropping rows with Missing values.

```
import pandas as pd
df = pd.read_csv('Car_Features.csv')
df = df.dropna()
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 8084 entries, 0 to 11913
Data columns (total 16 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Make                                  8084 non-null   object
1   Model                                8084 non-null   object
2   Year                                  8084 non-null   int64
3   Engine Fuel Type                      8084 non-null   object
4   Engine HP                             8084 non-null   float64
5   Engine Cylinders                      8084 non-null   float64
6   Transmission Type                    8084 non-null   object
7   Driven_Wheels                        8084 non-null   object
8   Number of Doors                      8084 non-null   float64
9   Market Category                      8084 non-null   object
10  Vehicle Size                         8084 non-null   object
11  Vehicle Style                        8084 non-null   object
12  highway MPG                          8084 non-null   int64
13  city mpg                             8084 non-null   int64
14  Popularity                           8084 non-null   int64
15  MSRP                                 8084 non-null   int64
dtypes: float64(3), int64(5), object(8)
memory usage: 1.0+ MB
None
```

`dropna()` removes rows or columns containing missing (NaN) values cleaning the dataset of all of the missing values that do not exist which provides us with more consistent data values and accurate analysis.

5. Taking care of missing values by replacing it with Mean

Taking care of missing values by putting Mean

```
import pandas as pd
df = pd.read_csv('Car_Features.csv')
df.fillna(df.mean(numeric_only=True), inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Make                  11914 non-null  object
1   Model                 11914 non-null  object
2   Year                  11914 non-null  int64
3   Engine Fuel Type      11911 non-null  object
4   Engine HP              11914 non-null  float64
5   Engine Cylinders      11914 non-null  float64
6   Transmission Type     11914 non-null  object
7   Driven_Wheels         11914 non-null  object
8   Number of Doors       11914 non-null  float64
9   Market Category       8172 non-null   object
10  Vehicle Size           11914 non-null  object
11  Vehicle Style          11914 non-null  object
12  highway MPG            11914 non-null  int64
13  city mpg               11914 non-null  int64
14  Popularity             11914 non-null  int64
15  MSRP                   11914 non-null  int64
dtypes: float64(3), int64(5), object(8)
memory usage: 1.5+ MB
```

All of the missing values are replaced by the mean of that corresponding column to get more accurate analysis and make sure that the data is consistent.

6.Creating Dummy variables for the Transmission type

```
import pandas as pd
df = pd.read_csv('Car_Features.csv')
transmission_dummies = pd.get_dummies(df['Transmission Type'])
df_with_dummies = pd.concat([df, transmission_dummies], axis=1)
df_with_dummies.info()
df_with_dummies.head(10)
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	...	Vehicle Style	highway MPG	city mpg	Popularity	MSRP	AUTOMATED_MANUAL	AUTOMATIC	DIRECT_DRIVE	MANUAL
0	BMW	Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	...	Coupe	26	19	3916	46135	False	False	False	True
1	BMW	Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	...	Convertible	28	19	3916	40650	False	False	False	True
2	BMW	Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	...	Coupe	28	20	3916	36350	False	False	False	True
3	BMW	Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	...	Coupe	28	18	3916	29450	False	False	False	True
4	BMW	Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	...	Convertible	28	18	3916	34500	False	False	False	True
5	BMW	Series	2012	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	...	Coupe	28	18	3916	31200	False	False	False	True

Creating dummy variables for the transmission type converts categorical data into a numeric format for machine learning models. Using `pd.get_dummies(df['Transmission'])`, each unique transmission type (e.g., Automatic, Manual) becomes a separate column with binary values (0 or 1), allowing models to interpret the categorical feature effectively without introducing ordering bias.

7. Find out outliers

```
import pandas as pd
df = pd.read_csv('Car_Features.csv')
column_to_check = 'MSRP'

Q1 = df[column_to_check].quantile(0.25)
Q3 = df[column_to_check].quantile(0.75)

IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = df[(df[column_to_check] < lower_bound) | (df[column_to_check] > upper_bound)]
print(f"Outliers in {column_to_check}:\n", outliers.head(10))
print("\nNumber of outliers:", outliers.shape[0])
```



Outliers in MSRP:

	Make	Model	Year	Engine	Fuel Type	Engine HP	\
294	Ferrari	360	2002	premium	unleaded (required)	400.0	
295	Ferrari	360	2002	premium	unleaded (required)	400.0	
296	Ferrari	360	2002	premium	unleaded (required)	400.0	
297	Ferrari	360	2002	premium	unleaded (required)	400.0	
298	Ferrari	360	2003	premium	unleaded (required)	400.0	

	Engine	Cylinders	Transmission	Type	Driven_Wheels	Number of Doors	\
294		8.0		MANUAL	rear wheel drive	2.0	
295		8.0		MANUAL	rear wheel drive	2.0	
296		8.0	AUTOMATED_MANUAL		rear wheel drive	2.0	
297		8.0	AUTOMATED_MANUAL		rear wheel drive	2.0	
298		8.0		MANUAL	rear wheel drive	2.0	

	Market Category	Vehicle Size	Vehicle Style	highway MPG	\
294	Exotic,High-Performance	Compact	Convertible	15	
295	Exotic,High-Performance	Compact	Coupe	15	
296	Exotic,High-Performance	Compact	Coupe	15	
297	Exotic,High-Performance	Compact	Convertible	15	
298	Exotic,High-Performance	Compact	Convertible	15	

	city mpg	Popularity	MSRP
294	10	2774	160829
295	10	2774	140615
296	10	2774	150694
297	10	2774	170829
298	10	2774	165986

Number of outliers: 996

8. Standardization and normalization of columns

```
import pandas as pd
from sklearn.preprocessing import StandardScaler

df = pd.read_csv('Car_Features.csv')
column_to_standardize = "MSRP"
scaler = StandardScaler()
df[column_to_standardize + " Standardized"] = scaler.fit_transform(df[[column_to_standardize]])
print(df.head(10).to_string())
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible
5	BMW	1 Series	2012	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe
6	BMW	1 Series	2012	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible
7	BMW	1 Series	2012	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe
8	BMW	1 Series	2012	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible
9	BMW	1 Series	2013	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler

df = pd.read_csv('Car_Features.csv')
column_to_normalize = "MSRP"
scaler = MinMaxScaler()
df[column_to_normalize + " Normalized"] = scaler.fit_transform(df[[column_to_normalize]])
print(df.head(10).to_string())
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible
5	BMW	1 Series	2012	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe
6	BMW	1 Series	2012	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible
7	BMW	1 Series	2012	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe
8	BMW	1 Series	2012	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible
9	BMW	1 Series	2013	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible

Standardization – This process transforms numerical features to have a **mean of 0** and a **standard deviation of 1** using the **Z-score formula**:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma}$$

where **XXX** is the original value, μ is the mean, and σ is the standard deviation. This method ensures that features with different units are comparable, making it useful for models like linear regression and SVM.

Normalization : his scales values between a fixed range, typically **[0,1]**, using **Min-Max scaling**:

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

where X_{\min} and X_{\max} are the minimum and maximum values of the feature. This helps models like neural networks that require inputs within a specific range.

Conclusion : Thus we have successfully applied all of the basic commands on our chosen dataset of Car Features and MSRP and have learned the basic process of modifying the data, cleaning it and preparing it for processing.