DSBDAL

Assignment - 1

Name:- Vinayaki Dalvi

Roll No.:- 31413

Date:- 02/02/2022

Title:- Data Wrangling

Problem Statement:- Perform the following operations using Python on any open source dataset (e.g.,data.csv)1.Import all the required Python Libraries.2.Locate an open source data from the web (e.g.https://www.kaggle.com). Provide a clear description of the data and its source (i.e.,URL of the web site).3.Load the Dataset into pandas dataframe.4.Data Preprocessing: check for missing values in the data using pandas insult(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.5.Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions.6.Turn categorical variables into quantitative variables in Python.

Learning Objectives:-

Objective is to understand and implement:-

1. Installation of Jupyter Notebook
2. Reading csv files and performing operations on data frames
3. Data pre-processing
4. Handling categorical values

Learning Outcomes:-

Students will be able to understand and implement:-

5. Installation of Jupyter Notebook
6. Reading csv files and performing operations on data frames
7. Handling categorical values

S/W and H/W requirements:-

Windows/ Linux OS -64bit
Ubuntu 20.04 LTS Server
Python IDE

Theory

A. Jupyter Notebook Installation on machines

1. Install Python3
   ```
   sudo apt install python3
   ```
2. Install pip3
   ```
   sudo apt install python3-pip
   ```
3. Install Jupyter Notebook
   ```
   pip3 install notebook
   ```
4. Check version
   ```
   jupyter --version
   ```
5. Launch Jupyter Notebook
   ```
   Jupyter Notebook
   ```

```
Create a python file.
```

B. CSV file

CSV Files

Files with .csv (Comma Separated Values) extension represent plain text files that contain records of data with comma separated values. Each line in a CSV file is a new record from the set of records contained in the file. Such files are generated when data transfer is intended from one storage system to another. Since all applications can

recognize records separated by comma, import of such data files to database is done very conveniently.

1. Download any DataSet from https://www.kaggle.com.

2. Add the downloaded csv file in folder in which we are working.

3. Import pandas and numpy in python file and create their object (pd,np).

4. Read csv file in the python file using:
   The read_csv is a Pandas method that allows a user to create a Pandas Dataframe from a local CSV.
   Var_name = pd.read_csv("Name_Of_CSV_File")

5. Print data by using print(Var_name)

6. To get information about read csv file use:-
   The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values).
   Var_name.info()

7. To get all the statistical values of the dataset.
   The describe() method is used for calculating some statistical data like percentile, mean and std of the numerical values of the Series or DataFrame. It analyzes both numeric and object series and also the DataFrame column sets of mixed data types.
   Var_name.describe()

8. To get number of null values in data set:-
   The isna() method returns a DataFrame object where all the values are replaced with a Boolean value True for NA (not-a -number) values, and otherwise False.
   Var_name.isna()

C. Type Conversion

The astype() method returns a new DataFrame where the data types has been changed to the specified type.

```
Var_name["Col_name"]=Var_name["Col_name"].astype("data_type")
```

Returns a Pandas DataFrame with the changes set according to the specified dtype(s).

D. Categorical Values and their conversion

One-hot Encoding is a type of vector representation in which *all of the elements* in a vector are 0, except for one, which has 1 as its value, where 1 represents a boolean specifying a category of the element.

For example, Column contains categorical ratings for the product and the values are: Bad, Average, Good.

Then one-hot-encoded vector for these values will be 001,010,100.

For getting one-hot-encoded vector for a given column of categorical values there is a function in pandas called get_dummies().

one_Hot_encoded_data=pd.get_dummies(Var_name,columns=['Name of column with categorical values])

Code

jupyter Assignment 1 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
CarSheet = pd.read_csv("Car_sales.csv")
```

In [4]:
```
CarSheet
```

Out[4]:

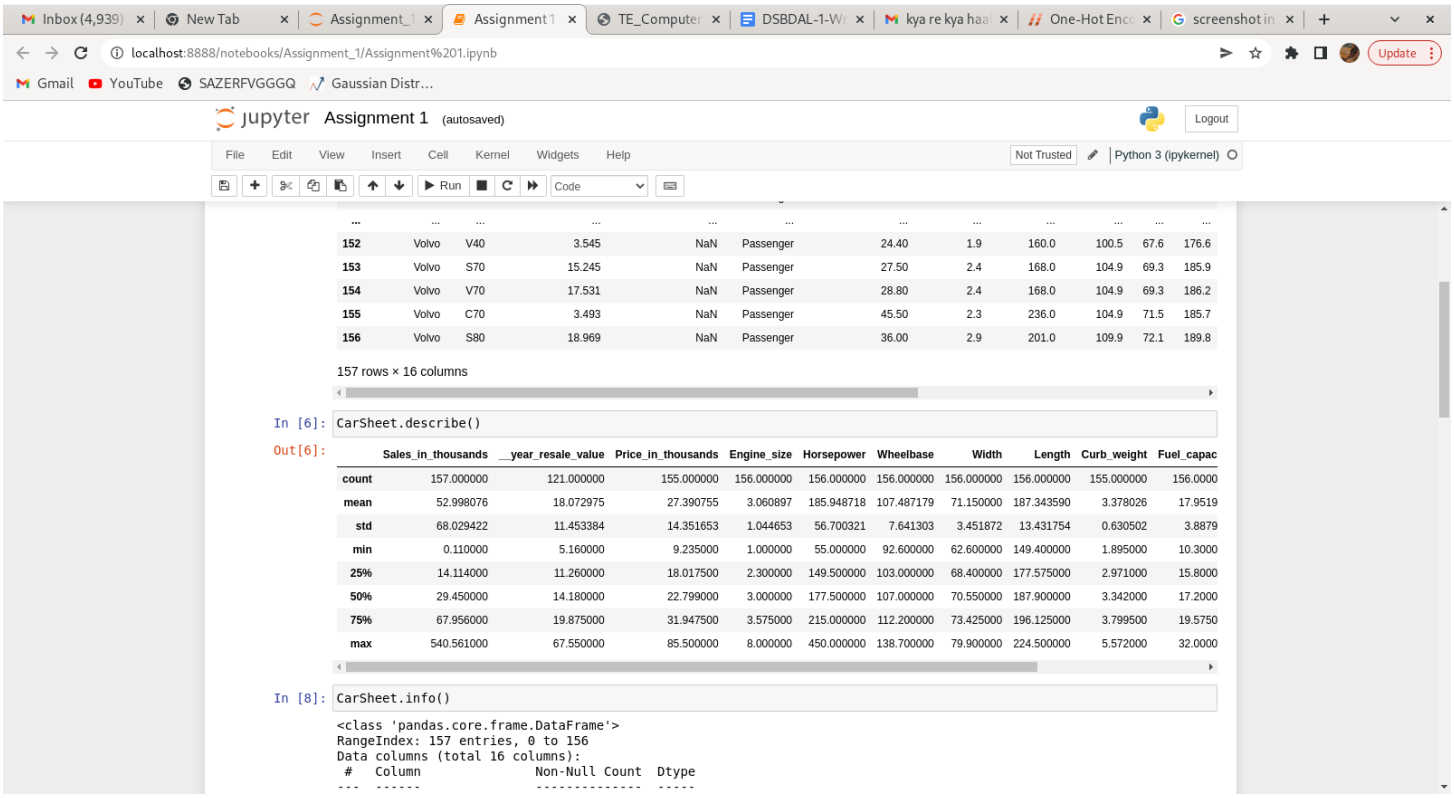
	Manufacturer	Model	Sales_in_thousands	__year_resale_value	Vehicle_type	Price_in_thousands	Engine_size	Horsepower	Wheelbase	Width	Length
0	Acura	Integra	16.919	16.360	Passenger	21.50	1.8	140.0	101.2	67.3	172.4
1	Acura	TL	39.384	19.875	Passenger	28.40	3.2	225.0	108.1	70.3	192.9
2	Acura	CL	14.114	18.225	Passenger	NaN	3.2	225.0	106.9	70.6	192.0
3	Acura	RL	8.588	29.725	Passenger	42.00	3.5	210.0	114.6	71.4	196.6
4	Audi	A4	20.397	22.255	Passenger	23.99	1.8	150.0	102.6	68.2	178.0
...	...	...	...	...	...	...	...	...	...	...	...
152	Volvo	V40	3.545	NaN	Passenger	24.40	1.9	160.0	100.5	67.6	176.6
153	Volvo	S70	15.245	NaN	Passenger	27.50	2.4	168.0	104.9	69.3	185.9
154	Volvo	V70	17.531	NaN	Passenger	28.80	2.4	168.0	104.9	69.3	186.2
155	Volvo	C70	3.493	NaN	Passenger	45.50	2.3	236.0	104.9	71.5	185.7
156	Volvo	S80	18.969	NaN	Passenger	36.00	2.9	201.0	109.9	72.1	189.8

157 rows × 16 columns

In [6]:
```
CarSheet.describe()
```

Out[6]:

	Sales_in_thousands	__year_resale_value	Price_in_thousands	Engine_size	Horsepower	Wheelbase	Width	Length	Curb_weight	Fuel_capac
count	157.000000	121.000000	155.000000	156.000000	156.000000	156.000000	156.000000	156.000000	155.000000	156.0000



...	...	...	...	...	...	...	...	...	...	...	...
152	Volvo	V40	3.545	NaN	Passenger	24.40	1.9	160.0	100.5	67.6	176.6
153	Volvo	S70	15.245	NaN	Passenger	27.50	2.4	168.0	104.9	69.3	185.9
154	Volvo	V70	17.531	NaN	Passenger	28.80	2.4	168.0	104.9	69.3	186.2
155	Volvo	C70	3.493	NaN	Passenger	45.50	2.3	236.0	104.9	71.5	185.7
156	Volvo	S80	18.969	NaN	Passenger	36.00	2.9	201.0	109.9	72.1	189.8

157 rows × 16 columns

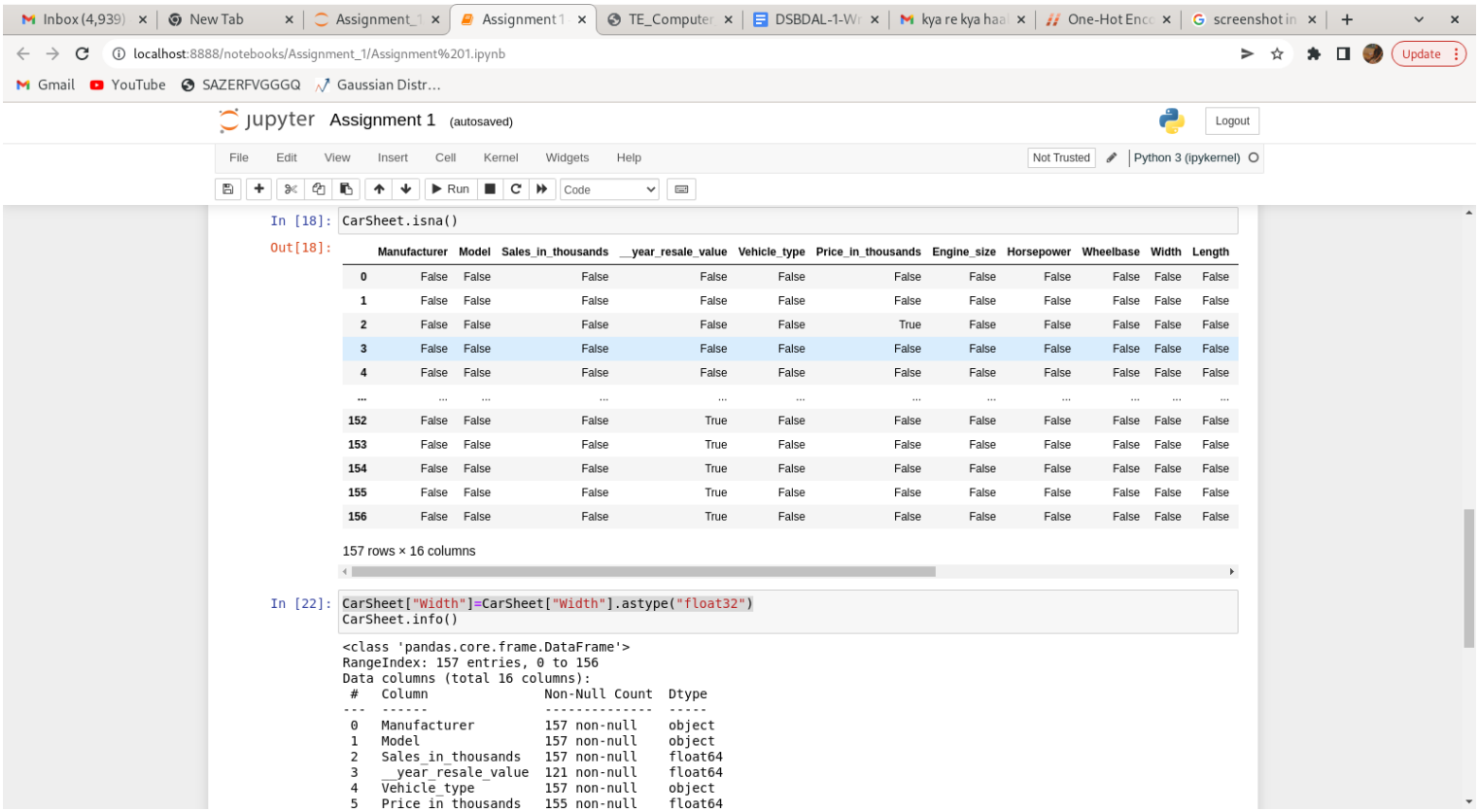
In [6]:
```
CarSheet.describe()
```

Out[6]:

	Sales_in_thousands	__year_resale_value	Price_in_thousands	Engine_size	Horsepower	Wheelbase	Width	Length	Curb_weight	Fuel_capac
count	157.000000	121.000000	155.000000	156.000000	156.000000	156.000000	156.000000	156.000000	155.000000	156.0000
mean	52.998076	18.072975	27.390755	3.060897	185.948718	107.487179	71.150000	187.343590	3.378026	17.9519
std	68.029422	11.453384	14.351653	1.044653	56.700321	7.641303	3.451872	13.431754	0.630502	3.8879
min	0.110000	5.160000	9.235000	1.000000	55.000000	92.600000	62.600000	149.400000	1.895000	10.3000
25%	14.114000	11.260000	18.017500	2.300000	149.500000	103.000000	68.400000	177.575000	2.971000	15.8000
50%	29.450000	14.180000	22.799000	3.000000	177.500000	107.000000	70.550000	187.900000	3.342000	17.2000
75%	67.956000	19.875000	31.947500	3.575000	215.000000	112.200000	73.425000	196.125000	3.799500	19.5750
max	540.561000	67.550000	85.500000	8.000000	450.000000	138.700000	79.900000	224.500000	5.572000	32.0000

In [8]:
```
CarSheet.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 157 entries, 0 to 156
Data columns (total 16 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
```



In [18]:
```
CarSheet.isna()
```

Out[18]:

	Manufacturer	Model	Sales_in_thousands	__year_resale_value	Vehicle_type	Price_in_thousands	Engine_size	Horsepower	Wheelbase	Width	Length
0	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	True	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...
152	False	False	False	True	False	False	False	False	False	False	False
153	False	False	False	True	False	False	False	False	False	False	False
154	False	False	False	True	False	False	False	False	False	False	False
155	False	False	False	True	False	False	False	False	False	False	False
156	False	False	False	True	False	False	False	False	False	False	False

157 rows × 16 columns

In [22]:
```
CarSheet["Width"]=CarSheet["Width"].astype("float32")
CarSheet.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 157 entries, 0 to 156
Data columns (total 16 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Manufacturer        157 non-null    object
 1   Model               157 non-null    object
 2   Sales_in_thousands  157 non-null    float64
 3   __year_resale_value  121 non-null    float64
 4   Vehicle_type        157 non-null    object
 5   Price_in_thousands  155 non-null    float64
```

jupyter Assignment 1 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

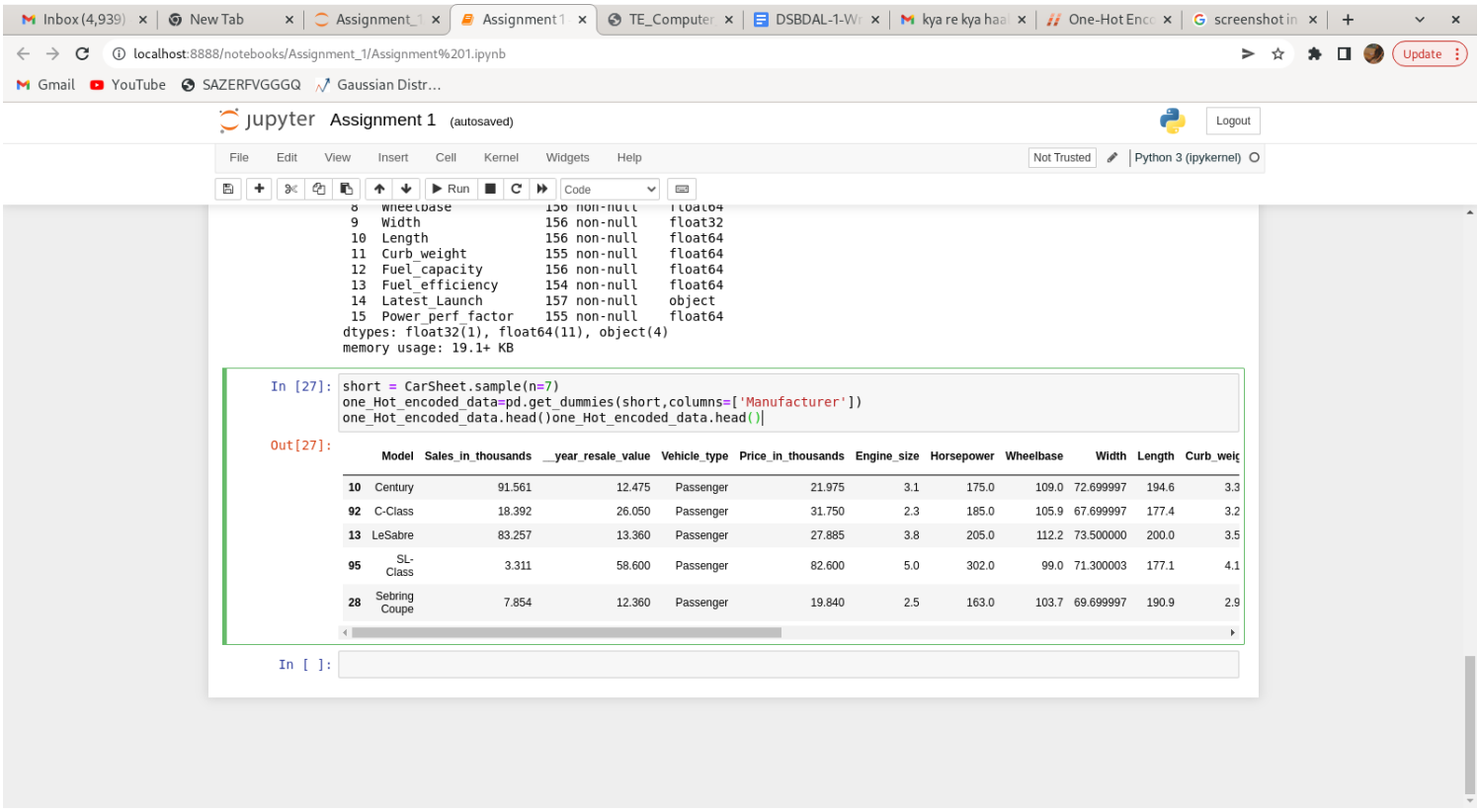
Run | Code

In [8]:
```
CarSheet.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 157 entries, 0 to 156
Data columns (total 16 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Manufacturer         157 non-null    object
 1   Model                157 non-null    object
 2   Sales_in_thousands   157 non-null    float64
 3   __year_resale_value  121 non-null    float64
 4   Vehicle_type         157 non-null    object
 5   Price_in_thousands   155 non-null    float64
 6   Engine_size          156 non-null    float64
 7   Horsepower           156 non-null    float64
 8   Wheelbase            156 non-null    float64
 9   Width                156 non-null    float64
 10  Length               156 non-null    float64
 11  Curb_weight          155 non-null    float64
 12  Fuel_capacity        156 non-null    float64
 13  Fuel_efficiency      154 non-null    float64
 14  Latest_Launch        157 non-null    object
 15  Power_perf_factor    155 non-null    float64
dtypes: float64(12), object(4)
memory usage: 19.8+ KB
```

In [12]:
```
CarSheet.shape
```

Out[12]: (157, 16)



Conclusion

We have implemented data pre-processing on Car_sales.csv file. We have converted the categorical value Manufacturer into numerical value using one hot encoding scheme.