SVKM's NMIMS
Mukesh Patel School of Technology Management & Engineering
Program: B.Tech\MBA.Tech\MBA.Tech AI

Course: Machine Learning

Experiment No.02

PART A

(PART A : TO BE REFFERED BY STUDENTS)

A.1 Aim: **To understand and implement d**ata exploration techniques using Pandas Library.

Task 1: Perform Exploratory data analysis on Car dataset and write the inferences for each question.

i. Read the Toyota.csv file into a DataFrame.
ii. Explore size, shape, data types of each column in the dataset.
iii. List down the columns of dataset
iv. Find out 'Fuel Type' for the 4^{th} row.
v. Find out value for second column for the 4^{th} row.
vi. Select all rows for column "Fuel Type"
vii. Select all rows for columns "KM", "HP" and "Automatic"
viii. Display 1 to 5 rows for columns 2 to 4 (excluding row 5 and column 4)
ix. Display the info of dataset and state your observations
x. Identify unique values for columns "KM", "HP" and "Doors"
xi. Create a new data frame, by replacing "??" with NAN
xii. Replace the categorical values in the "Doors" column with its corresponding numeric value
xiii. Convert data types of columns "Doors", "MetColor" and "Automatic" to int, and object
xiv. Identify the total number of null values in each column of the data set
xv. Drop rows with null values
xvi. Identify total number of cars that runs on "Petrol", "Diesel" or "CNG"
xvii. Identify mean of "KM" for the cars that runs on "Diesel"

Task 2:
Perform one hot and label encoding on relationship column of "adults" dataset

A.2 Prerequisite:

Python Programming, Pandas library

A.3 Outcome:

After successful completion of this experiment students will be able to:

i. Read different types of data files(csv, excel, text file etc.)
ii. Obtain metadata of given dataset
iii. Understand finding of null values and replacing null values
iv. Understand and implement class label encoding
v. Understand and implement one hot encoding

A.4 Theory:

Exploratory Data Analysis:

Exploratory Data Analysis (EDA) is an open-ended process where we calculate statistics and make figures to find trends, anomalies, patterns, or relationships within the data. The goal of EDA is to learn what our data can tell us. It generally starts out with a high level overview, then narrows in to specific areas as we find intriguing areas of the data. The findings may be interesting in their own right, or they can be used to inform our modeling choices, such as by helping us decide which features to use.

Pandas Library:

DataFrame is a 2-dimensional labeled data structure with columns of potentially different types. You can think of it like a spreadsheet or SQL table, or a dict of Series objects. It is generally the most commonly used pandas object. Like Series, DataFrame accepts many different kinds of input:

- Dict of 1D ndarrays, lists, dicts, or Series
- 2-D numpy.ndarray
- Structured or record ndarray
- A Series
- Another DataFrame

Encoding:

One hot encoding:

One-hot encoding converts the categorical data into numeric data by splitting the column into multiple columns. The numbers are replaced by 1s and 0s, depending on which column has what value

Label encoding:

This approach is very simple and it involves converting each value in a column into a number.

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical.)

Roll No. I066	Name: Srihari Thyagarajan
Class: B Tech Artificial Intelligence	Batch: B2
Date of Experiment: 22/12/2022	Date of Submission:
Grade:	

B.1 Task1

```
# ML Practical Experiment 2
```

```
# import libraries
import pandas as pd
import numpy as np
import statistics as statistics
from sklearn.preprocessing import OneHotEncoder # For Task 2
```

Task 1

```
df = pd.read_excel("/content/Toyota.csv")
```

```
df
```

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986	Diesel	90	1.0	0	2000	three	1165
1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	13950	24.0	41711	Diesel	90	NaN	0	2000	3	1165
3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	NaN	20544	Petrol	86	1.0	0	1300	3	1025
1432	10845	72.0	??	Petrol	86	0.0	0	1300	3	1015
1433	8500	NaN	17016	Petrol	86	0.0	0	1300	3	1015
1434	7250	70.0	??	NaN	86	1.0	0	1300	3	1015
1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 10 columns

```
[66] # Size
     df.size

     14360
```

```
[67] # Shape
     df.shape

     (1436, 10)
```

```
# Data Types
df.dtypes

Price          int64
Age          float64
KM            object
FuelType      object
HP            object
MetColor      object
Automatic     object
CC             int64
Doors          int64
Weight         int64
dtype: object
```

```
[69] # Columns of a Dataset
     for column in df.columns:
       print(column)

     Price
     Age
     KM
     FuelType
     HP
     MetColor
     Automatic
     CC
     Doors
     Weight
```

```
[70] # Fuel Type of the 4th row
     df['FuelType'][3]

     'Diesel'
```

```
[85] # Value for second column for the 4th row
     df.iloc[:, 2][4]

     38500
```

38500

```python
df['FuelType']
```

```
0       Diesel
1       Diesel
2       Diesel
3       Diesel
4       Diesel
         ...
1431    Petrol
1432    Petrol
1433    Petrol
1434         0
1435    Petrol
Name: FuelType, Length: 1436, dtype: object
```

```python
df[["FuelType", "KM", "HP"]]
```

	FuelType	KM	HP
0	Diesel	46986	90
1	Diesel	72937	90
2	Diesel	41711	90
3	Diesel	48000	90
4	Diesel	38500	90
...	...	...	...
1431	Petrol	20544	86
1432	Petrol	NaN	86
1433	Petrol	17016	86
1434	0	NaN	86
1435	Petrol	1	110

1436 rows × 3 columns

[130]
```python
# Value for 1-5 rows and 2-4 columns exluding the 5th row and 4th column.
df.iloc[1: 5, 2 : 4]
```

	fnlwgt	education
1	89814	HS-grad
2	336951	Assoc-acdm
3	160323	Some-college
4	103497	Some-college

```
# Info of dataset:
df
```

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986	Diesel	90	1.0	0	2000	3	1165
1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	13950	24.0	41711	Diesel	90	0.0	0	2000	3	1165
3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	0.0	20544	Petrol	86	1.0	0	1300	3	1025
1432	10845	72.0	NaN	Petrol	86	0.0	0	1300	3	1015
1433	8500	0.0	17016	Petrol	86	0.0	0	1300	3	1015
1434	7250	70.0	NaN	0	86	1.0	0	1300	3	1015
1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 10 columns

Observations from the Dataset:

From the Dataset we observe that:

1. The columns KM -> Kilometres and Doors should have the Integer datatype. However from the dataframe we observe that some values in these columns have non-integer values.
2. The datatypes of these 2 columns have the "object" datatype.

```
df["KM"].unique()
```

```
array([46986, 72937, 41711, ..., 30964, 20544, 17016], dtype=object)
```

```
[77] df["HP"].unique()
```

```
array([90, '????', 192, 110, 97, 71, 116, 98, 69, 86, 72, 107, 73],
      dtype=object)
```

```
[78] df["Doors"].unique()
```

```
array(['three', 3, 5, 4, 'four', 'five', 2], dtype=object)
```

```
[133] df[["KM","HP","Doors"]].nunique()
```

```
KM       1256
HP         13
Doors       7
dtype: int64
```

```
df = df.fillna(0)
df.replace('??', 'NaN', inplace = True)

# Forming a new dataframe
newdf = df.replace(to_replace = ["??","????"], value = "NAN")
newdf


# df_1 = df.fillna(0)
# df_1.replace('??', 'NaN', inplace = True)
```

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986	Diesel	90	1.0	0	2000	three	1165
1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	13950	24.0	41711	Diesel	90	0.0	0	2000	3	1165
3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	0.0	20544	Petrol	86	1.0	0	1300	3	1025
1432	10845	72.0	NaN	Petrol	86	0.0	0	1300	3	1015
1433	8500	0.0	17016	Petrol	86	0.0	0	1300	3	1015
1434	7250	70.0	NaN	0	86	1.0	0	1300	3	1015
1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 10 columns

```
[211] # New dataframe containing ?? replaced with NaN
newdf
```

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986	Diesel	90	1.0	0	2000	three	1165
1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	13950	24.0	41711	Diesel	90	0.0	0	2000	3	1165
3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	0.0	20544	Petrol	86	1.0	0	1300	3	1025
1432	10845	72.0	NaN	Petrol	86	0.0	0	1300	3	1015
1433	8500	0.0	17016	Petrol	86	0.0	0	1300	3	1015
1434	7250	70.0	NaN	0	86	1.0	0	1300	3	1015
1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 10 columns

```
[220] # Categorical
newdf["Doors"].replace(["three", "four", "five"], [3, 4, 5], inplace = True)
```

```
[221] newdf
```

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986	Diesel	90	1.0	0	2000	3	1165
1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	13950	24.0	41711	Diesel	90	0.0	0	2000	3	1165
3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	0.0	20544	Petrol	86	1.0	0	1300	3	1025
1432	10845	72.0	NaN	Petrol	86	0.0	0	1300	3	1015
1433	8500	0.0	17016	Petrol	86	0.0	0	1300	3	1015
1434	7250	70.0	NaN	0	86	1.0	0	1300	3	1015
1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 10 columns

```
newdf["Doors"] = newdf["Doors"].astype(int)
newdf["MetColor"] = newdf["MetColor"].astype(object)
newdf["Automatic"] = newdf["Automatic"].astype(object)
```

```
[223] df_1 = pd.read_excel("/content/Toyota.csv")
      df_1.isnull().sum()
```

```
Price          0
Age          100
KM             0
FuelType     100
HP             0
MetColor     150
Automatic      0
CC             0
Doors          0
Weight         0
dtype: int64
```

```
newdf.dropna()
```

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986	Diesel	90	1.0	0	2000	3	1165
1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	13950	24.0	41711	Diesel	90	0.0	0	2000	3	1165
3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...
1431	7500	0.0	20544	Petrol	86	1.0	0	1300	3	1025
1432	10845	72.0	NaN	Petrol	86	0.0	0	1300	3	1015
1433	8500	0.0	17016	Petrol	86	0.0	0	1300	3	1015
1434	7250	70.0	NaN	0	86	1.0	0	1300	3	1015
1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 10 columns

```
[225] newdf["FuelType"].value_counts()

Petrol    1177
Diesel     144
0          100
CNG         15
Name: FuelType, dtype: int64
```

```
[218] newdf

[226] l = []

      for i in range(len(newdf['FuelType'])):
        if newdf['FuelType'][i] == 'Diesel':
          if newdf['KM'][i] != 'NaN':
            l.append(int(newdf['KM'][i]))

      np.mean(l)

      114927.87857142858
```

B.2 Task 2

Task 2

```
[120] df = pd.read_excel("/content/adult.csv")
      df_1 = pd.read_excel("/content/adult.csv")
```

```
[123] # Checking for the labels in the categorical parameters
      df_1["relationship"].unique()
```

```
array(['Own-child', 'Husband', 'Not-in-family', 'Unmarried', 'Wife',
       'Other-relative'], dtype=object)
```

```
# Checking for the label counts in the categorical parameters
df_1["relationship"].value_counts()
```

```
Husband           19716
Not-in-family     12583
Own-child          7581
Unmarried          5125
Wife               2331
Other-relative     1506
Name: relationship, dtype: int64
```

Method 1:

One Hot Encoding using Sci-kit learn Library:

```
[110] # Creating aninstance of the one-hot-encoder
      encoder = OneHotEncoder(handle_unknown='ignore')

      # Perform one-hot encoding on 'relationship' column
      encoder_df = pd.DataFrame(encoder.fit_transform(df[['relationship']]).toarray())
```

```
[111] # Merging one-hot encoded columns back with original DataFrame df.
      final_df = df.join(encoder_df)
```

```
final_df
```

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	race	gender	...	capital-loss	hours-per-week	native-country	income	0	1	2	3	4	
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	...	0	40	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	...	0	50	United-States	<=50K	1.0	0.0	0.0	0.0	0.0	0.
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	...	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	...	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.
4	18	?	103497	Some-college	10	Never-married	?	Own-child	White	Female	...	0	30	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
48837	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White	Female	...	0	38	United-States	<=50K	0.0	0.0	0.0	0.0	0.0	1.
48838	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	...	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.
48839	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Female	...	0	40	United-States	<=50K	0.0	0.0	0.0	0.0	1.0	0.
48840	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Male	...	0	20	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.
48841	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Female	...	0	40	United-States	>50K	0.0	0.0	0.0	0.0	0.0	1.

48842 rows × 21 columns

48842 rows × 21 columns

```
[113] # Dropping the original relationship column from the dataframe as we will be refering only to the Numerical values
      # which are to be generated

      final_df.drop('relationship', axis=1, inplace=True)
```

```
final_df
```

```
final_df
```

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income	0	1	2	3	4	5
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Black	Male	0	0	40	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.0
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	White	Male	0	0	50	United-States	<=50K	1.0	0.0	0.0	0.0	0.0	0.0
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	White	Male	0	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.0
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Black	Male	7688	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.0
4	18	?	103497	Some-college	10	Never-married	?	White	Female	0	0	30	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
48837	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	White	Female	0	0	38	United-States	<=50K	0.0	0.0	0.0	0.0	0.0	1.0
48838	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	White	Male	0	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.0
48839	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	White	Female	0	0	40	United-States	<=50K	0.0	0.0	0.0	0.0	1.0	0.0
48840	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	White	Male	0	0	20	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.0
48841	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	White	Female	15024	0	40	United-States	>50K	0.0	0.0	0.0	0.0	0.0	1.0

48842 rows × 20 columns

```
[126] final_df.columns = ['age',        'workclass',          'fnlwgt',
                 'education', 'educational-num',  'marital-status',
                'occupation',            'race',          'gender',
              'capital-gain',    'capital-loss',  'hours-per-week',
            'native-country',          'income', 'Own-child', 'Husband', 'Not-in-family', 'Unmarried', 'Wife', 'Other-relative']
```

```
final_df
```

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income	Own-child	Husband	Not-in-family	Unmarried	Wife	Other-relative
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Black	Male	0	0	40	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.0
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	White	Male	0	0	50	United-States	<=50K	1.0	0.0	0.0	0.0	0.0	0.0
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	White	Male	0	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.0
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Black	Male	7688	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.0
4	18	?	103497	Some-college	10	Never-married	?	White	Female	0	0	30	United-States	<=50K	0.0	0.0	0.0	1.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
48837	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	White	Female	0	0	38	United-States	<=50K	0.0	0.0	0.0	0.0	0.0	1.0
48838	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	White	Male	0	0	40	United-States	>50K	1.0	0.0	0.0	0.0	0.0	0.0
48839	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	White	Female	0	0	40	United-States	<=50K	0.0	0.0	0.0	0.0	1.0	0.0

Method 2

One-Hot encoding the categorical parameters using get_dummies()

```
[128] one_hot_encoded_data = pd.get_dummies(df_1, columns = ['relationship'])
```

```
one_hot_encoded_data
```

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income	relationship_Husband	relationship_Not-in-family	relatio
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Black	Male	0	0	40	United-States	<=50K	0	0	
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	White	Male	0	0	50	United-States	<=50K	1	0	
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	White	Male	0	0	40	United-States	>50K	1	0	
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Black	Male	7688	0	40	United-States	>50K	1	0	
4	18	?	103497	Some-college	10	Never-married	?	White	Female	0	0	30	United-States	<=50K	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
48837	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	White	Female	0	0	38	United-States	<=50K	0	0	
48838	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	White	Male	0	0	40	United-States	>50K	1	0	
48839	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	White	Female	0	0	40	United-States	<=50K	0	0	
48840	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	White	Male	0	0	20	United-States	<=50K	0	0	
48841	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	White	Female	15024	0	40	United-States	>50K	0	0	

48842 rows × 20 columns

Warning: total number of rows (48842) exceeds max_rows (20000). Limiting to first (20000) rows.

B.4 Conclusion:

From the above experiment, I learnt the following:

- Read different types of data files (csv, excel, text file etc.).
- Obtain metadata of given dataset.
- Understand finding of null values and replacing null values.
- Understand and implement class label encoding.
- Understand and implement one hot encoding.