Machine Learning
Ex: 02 Pandas, Numpy, Sub Plot

Name: Athithraja R
Reg.no: 2022503702

PANDAS

1. To read the content in excel file to the data frame…….

```
[ ] #1. To read the content of an Excel file into a DataFrame:
```

```
[ ] import pandas as pd
```

```
[ ] data_frame=pd.read_excel('a.xlsx')
    print(data_frame)
```

```
     Name  Age
0    athi   20
1     als   27
2   as;sq   22
3   qwdjq   10
4   qd;kj   90
5    athi   20
6     als   27
7   as;sq   22
8   qwdjq   10
9   qd;kj   90
10   athi   20
11    als   27
12  as;sq   22
13  qwdjq   10
14  qd;kj   90
```

2. To read the content in CSV file to the data frame……

```
[ ] #2. To read the content of a CSV file into a DataFrame:
```

```
data_csv=pd.read_csv('b.csv')
print(data_csv)
```

```
     Name   Age
0    Athi    20
1   sjksx    78
2    xsjb     8
3     jks    30
4     ksj    31
5    Athi    20
6   sjksx    78
7    xsjb     8
8     jks    30
9     ksj    31
10   Athi    20
11  sjksx    78
12   xsjb     8
13    jks    30
```

3. The code to print first five rows in the dataframe……..

```
[ ] #3. Print the first five rows:

print(data_csv.head())

    Name   Age
0   Athi    20
1  sjksx    78
2   xsjb     8
3    jks    30
4    ksj    31
```

4. The code to print last five rows in the dataframe…….

```
[ ] #4. Print the last five rows:

print(data_csv.tail())

     Name   Age
9     ksj    31
10   Athi    20
11  sjksx    78
12   xsjb     8
13    jks    30
```

5. The code to print first three rows in the dataframe……

```
[ ] #5. Print the first 3 rows:

[ ] print(data_csv.head(3))

    Name   Age
0   Athi    20
1  sjksx    78
2   xsjb     8
```

6. The code to print last three rows in the dataframe……

```
[ ] #6. Print the last 3 rows:

[ ] print(data_csv.tail(3))

     Name   Age
11  sjksx    78
12   xsjb     8
13    jks    30
```

7. The method to print the summary of data and what attributes will be displayed…….

```
[ ] #7. Print a summary of the data:
```

```
[ ] print(data_csv.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Name    14 non-null     object
 1    Age    14 non-null     int64
dtypes: int64(1), object(1)
memory usage: 356.0+ bytes
None
```

8. The attributes that will be printed using describe ()…….

```
[ ] #8. Attributes printed by describe():
```

```
[ ] print(data_csv.describe())
```

```
             Age
count  14.000000
mean   33.571429
std    25.551499
min     8.000000
25%    20.000000
50%    30.000000
75%    31.000000
max    78.000000
```

9. To print summary of categorical values……

```
+ Code  + Text
[ ] #9. Print summary of categorical values:
```

```
[ ] print(data_csv.describe(include='object'))
```

```
        Name
count     14
unique     5
top     Athi
freq       3
```

10. To print categorical column values alone using describe method…….

```
[ ] #10. Print categorical column values using describe():
```

```
[ ] print(data_frame['Name'].describe())
```

```
count       15
unique       5
top       athi
freq         3
Name: Name, dtype: object
```

11. To print the summary of single attribute using describe ()…….

```
[ ] #11. Summary of a single attribute using describe()
```

```
[ ] print(data_frame['Age'].describe())
```

```
count    15.000000
mean     33.800000
std      29.643597
min      10.000000
25%      20.000000
50%      22.000000
75%      27.000000
max      90.000000
Name: Age, dtype: float64
```

12. The method to print distinct observations for each attribute…….

```
[ ] #12. Print distinct observations for each attribute
```

```
[ ] print(data_frame.nunique())
```

```
Name    5
Age     5
dtype: int64
```

13. The method to print unique values of a column in ascending order……

```
[ ] #13. Print unique values of a column in ascending order:
```

```
[ ] print(sorted(data_frame['Age'].unique()))
```

```
[10, 20, 22, 27, 90]
```

14. To print the summary of a column by grouping the data…….

```
[ ] #14. Print summary of a column by grouping the data
```

```
[ ] print(data_frame.groupby('Name')['Age'].mean())
```

```
Name
als      27.0
as;sq    22.0
athi     20.0
qd;kj    90.0
qwdjq    10.0
Name: Age, dtype: float64
```

15. The code to print the number of groups created along with row number……..

```
[ ] #15. Print the number of groups created along with row count
```

```
print(data_frame.groupby('Age').size())
```

```
Age
10    3
20    3
22    3
27    3
90    3
dtype: int64
```

16. The method to print size of the group……

```
#16. Print size of the group:
```

```
[ ] print(data_frame.groupby('Name').size())
```

```
Name
als      3
as;sq    3
athi     3
qd;kj    3
qwdjq    3
dtype: int64
```

17. The code to print particular group values……..

```
#17. Print particular group values:
```

```
[ ] print(data_frame.groupby('Name').get_group('athi'))
```

```
    Name  Age
0   athi   20
5   athi   20
10  athi   20
```

18. The code to print count, max and min values of a group……..

```
[ ] #18. Print count, max and min values of a group
```

```
print(data_frame.groupby('Age').agg(['count','max','min']))
```

```
     Name
    count    max    min
Age
10      3  qwdjq  qwdjq
20      3   athi   athi
22      3  as;sq  as;sq
27      3    als    als
90      3  qd;kj  qd;kj
```

NUMPY

1. To create an array with three rows and three columns.…….

```
[ ] #Numpy Examples
```

```
[ ] #1. Create a 3x3 array
```

```
[ ] import numpy as np
    arr=np.array([[1,2,3],[4,5,6],[7,8,9]])
    print(arr)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

2. The code to print dimensions of the array ……

```
[ ] #2. Print the dimension of the array
```

```
[ ] print(arr.shape)
```

```
(3, 3)
```

3. Create a 1D and 2D array with default initialization of zeros ……..

```
[ ] #3. Create a 1D and 2D array of zeros
```

```
[ ] arr1=np.zeros(5)
    arr2=np.zeros((5,5))
    print(arr1)
    print("-------------------")
    print(arr2)
```

```
[0. 0. 0. 0. 0.]
-------------------
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

4. To create an identity matrix with 5 rows and 5 columns …….

```
[ ] #4. Create an identity matrix of 5x5
```

```
[ ] id_mat=np.eye(5)
    print(id_mat)
```

```
[[1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1.]]
```

5. The code to retrieve second row first column value in a 3x3 matrix ……

```
#5. Retrieve second row, first column value
```

```
[ ] value= arr[1,2]
    print(value)
```

```
6
```

6. The code to print all column values of second row in a 3x3 matrix ……

```
[ ] #6. Print all column values of second row
```

```
row2=arr[1,:]
print(row2)
```

```
[4 5 6]
```

7. The code to retrieve all the values from second column in a 3x3 matrix …….

```
#7. Retrive all values from the second column
```

```
[ ] col2=arr[:,1]
    print(col2)
```

```
[2 5 8]
```

8. to retrieve last column value in a 3x3 matrix …….

```
[ ] #8. Retrive the last column in 3x3 matrix
```

```
[ ] last_col=arr[:,-1]
    print(last_col)
```

```
[3 6 9]
```

9. To retrieve all column values from second and third row in a 3x3 matrix

……

```
[ ] #9. Retrive values from second and third row, all columns
```

```
row2_3=arr[1:3,:]
print(row2_3)
```

```
[[4 5 6]
 [7 8 9]]
```

10. To create an array with numbers specified in the range …….

```
[ ] #10. Create an array with numbers specified in a range
```

```
range_arr=np.arange(0,10,2)
print(range_arr)
```

```
[0 2 4 6 8]
```

11. To find transpose of a matrix …….

```
[ ] #11. Find the transpose of a matrix
```

```
[ ] transpose_arr=arr.T
    print(transpose_arr)
```

```
[[1 4 7]
 [2 5 8]
 [3 6 9]]
```

12. To find determinant of a matrix …….

```
[ ] #12. Find the determinant of  a matrix
```

```
[ ] determinent_value= np.linalg.det(arr)
    print(round(determinent_value))
```

```
0
```

13. To print diagonal elements of a matrix ……

```
[ ] #13. Print diagonal element of a matrix
```

```
[ ] diagonal=np.diagonal(arr)
    print(diagonal)
```

```
[1 5 9]
```

14. **Basic Matrix operations:**

x1=np.array([[1,2],[3,4]])

x2=np.array([[8,7],[6,9]])

…….

```
[ ] #14. Basic matrix operations(example of addition)
```

```
x1=np.array([[1,2],[3,4]])
x2=np.array([[8,7],[6,9]])
result=x1*x2
print(result)
```

```
[[ 8 14]
 [18 36]]
```

15. To create an array with random integer values of size (3,5) ……..

```
[ ] #15. Create an array with random integer values of size(3,5)
```

```
[ ] rand_arr=np.random.randint(0,10,size=(3,5))
    print(rand_arr)
```

```
[[5 1 9 9 6]
 [8 2 8 4 3]
 [0 5 5 3 9]]
```

16. To create 1D array with 5 values of equal step size ……

```
#16. Create a 1D array with 5 values of equal step size
```

```
[ ] linear_arr=np.linspace(0,10,5)
    print(linear_arr)
    #Formula for equal step size: (stop-start)/(num-1) ....the x= x +
```

```
[ 0.   2.5  5.   7.5 10. ]
```

SUB PLOTS

1. **Code:**

```
import matplotlib.pyplot as plt
#Data for subplots
temperature=[20, 25, 30, 35,40]
icesales [13, 21, 25, 35, 38]
coffeesales=[45, 37, 28, 22,18]
#Create Figure for subplots
fig, ax=plt.subplots (nrows=1,ncols=2, figsize=(9,3)
#Set title for subplots
ax[0].set_title('a. Icecream Sales')
ax[1].set_title('b. Coffee Sales')
#Generate the subplots
ax[0].plot(temperature, icesales, '-o',c='orange')
ax[1].plot(temperature, coffeesales, '-*',c='red')
#Set the Xlabel and Ylabel
ax[0].set_xlabel('Temperature')
ax[0].set_ylabel('Ice Sales (in litres)')
ax[1].set_xlabel('Temperature')
ax[1].set_ylabel('Coffee Sales (in litres)')
fig.subtitle('SALES')
```

```
Text(0.5, 0.98, 'SALES')
```