Practical-1

AIM:- Explore any one machine learning tool. (Google Colab)

Description:

To start with machine learning we need the tool by which using we can develop machine learning

model. In market, there are so many tools used such as Weka, Tensorflow, Scikit-learn, Colab, etc.)

Here we are going to use google colab which is from Google inc. advantages of using google colab

are as follows

Google Colab is a cloud-based machine learning platform that offers many benefits, including:

- **Easy to use**

  No setup is required, and you can start coding immediately after creating an account

- **Affordable**

  Free for most tasks, with paid plans for more demanding needs

- **Flexible**

  You can use Colab to train and run machine learning models, process data, createvisualizations, and collaborate with others

- Saves time and resources

  You don't need to set up your own computing environment, which can save time and resources when working on complex projects

- Access to powerful hardware

  You can run your code on Google's powerful hardware infrastructure without any setup or installation

- Access to GPU and TPU resources

  You can access GPU and TPU resources for faster training of machine learning models

- Integrated with GitHub

  You can save and load notebooks from GitHub in the Colab environment.

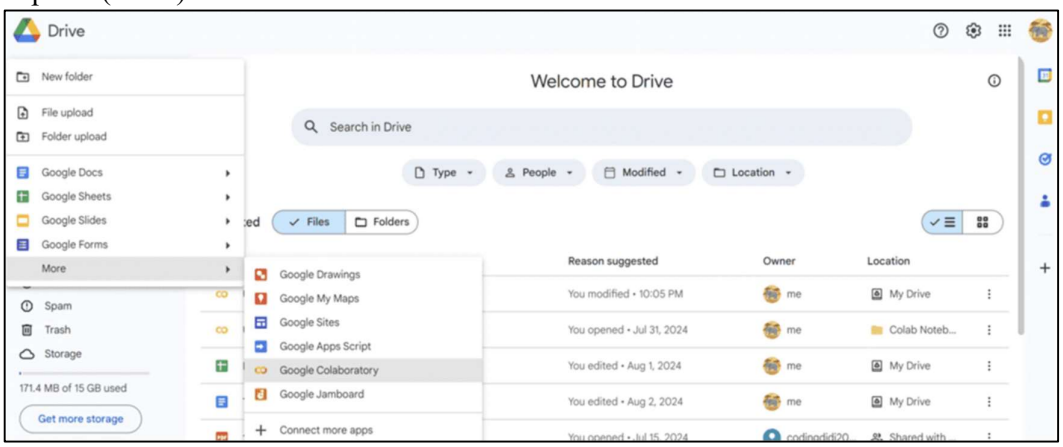
- Access control

  Access control is easier in Google Colab if your work team uses Gmail or Gsuite.

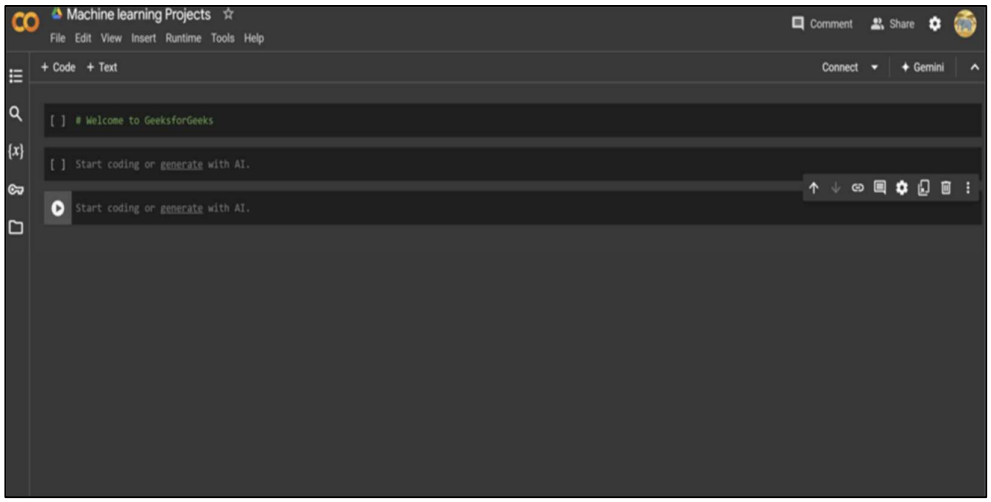
Google Colab Guide for Machine Learning Projects

You can open the colab by clicking the following step is as follows –

- Step 1: Create a Google account for the drive.
- Step 2: Open Google Drive and click "+new" in the top left corner.
- Step 3: Start navigating and click to more options and you will get the specified option (colab).

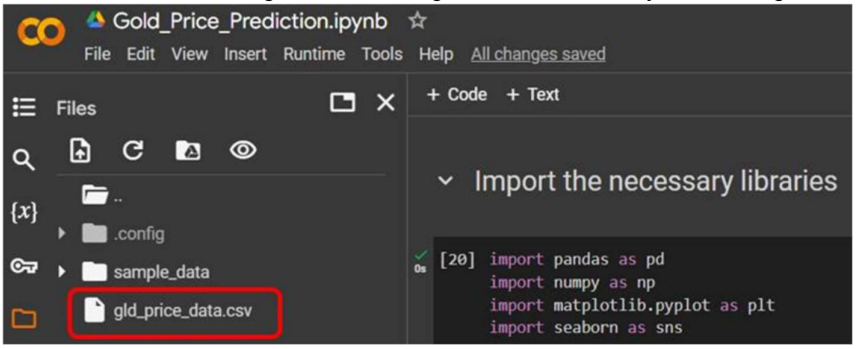


- Step 4: Click on Google Colaboratory to get the interface.



- Step 5: In this Step, We will first discuss the upload of datasets in colab.

  Go to Files -> See the upload icon -> upload the file from your desktop files.



To run the particular cell click in play button located at left side of the cell.

Conclusion

The Google Colab platform is similar to Jupyter Notebook, but sometimes we have to go through the library installation in a code section, upload the file, and keep it on the drive. We get the information about the datasets through the code. As you can see, the working functionality of both the platforms are same but the Jupyter Notebook has very advanced features. There are so many features available tool to tool. We can use any tool as per our convenience.

Practical-2

Practical-2.1

AIM:- Write a NumPy program to implement following operation

- **to convert a list of numeric values into a one-dimensional NumPy array**

INPUT:-

```
import numpy as np

my_list = [1,2,3,4,5,6,7]

my_array = np.array(my_list)

print(my_array)
```

OUTPUT

```
Original list: [1, 2, 3, 4, 5, 6, 7]
Convert list into array: [1 2 3 4 5 6 7]
```

Practical-2.2

AIM:- TO CREATE A 3X3 MATRIX WITH VALUES RANGING FROM 2 TO 10.

INPUT:-

```
import numpy as np

myarray=np.arange(2,11)

matrix=myarray.reshape(3,3)

print(matrix)
```

OUTPUT:-

```
[[ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]
```

Practical-2.3

AIM:- TO APPEND VALUES AT THE END OF AN ARRAY.

INPUT:-

```
import numpy as np

array1=np.array([2,3,4,5])

array2=[1,6,7,8]

append_array=np.append(array1,array2)

print("Before Adding element in array:",array1)

print("After Adding element in array :",append_array)
```

OUTPUT:-

```
Before Adding element in array: [2 3 4 5]
After Adding element in array : [2 3 4 5 1 6 7 8]
```

Practical-2.4

AIM:- TO CREATE ANOTHER SHAPE FROM AN ARRAY WITHOUT CHANGING ITS DATA(3*2 TO 2*3)

INPUT:-

```
import numpy as np
original_array=np.array([[1,2],
                         [3,4],
                         [5,6]])
reshaped_array=original_array.reshape(2,3)
print("Original array (3*2):")
print(original_array)
print("Reshaped array (2*3):")
print(reshaped_array)
```

OUTPUT:-

```
Original array (3*2):
[[1 2]
 [3 4]
 [5 6]]
Reshaped array (2*3):
[[1 2 3]
 [4 5 6]]
```

Practical-3

Practical-3.1

AIM:- WRITE A NUMPY PROGRAM TO SPLIT AN ARRAY OF 14 ELEMENTS INTO 3 ARRAYS, EACH WITH 2, 4, AND 8 ELEMENTS IN THE ORIGINAL ORDER.

INPUT:-

```
import numpy as np

myarray=np.arange(1,15)

array1=myarray[:2]

array2=myarray[2:6]

array3=myarray[6:]

print("array 1:",array1)

print("array 2:",array2)

print("array 3:",array3)
```

OUTPUT:-

```
array 1: [1 2]
array 2: [3 4 5 6]
array 3: [ 7  8  9 10 11 12 13 14]
```

Practical-3.2

AIM: - WRITE A NUMPY PROGRAM TO STACK ARRAYS HORIZONTALLY (COLUMN WISE).

INPUT: -

```
import numpy as np
array1=np.array([[1,4],
[4,6]])
array2=np.array([[1,3],
[2,5]])
hstacked_arr=np.hstack((array1,array2))
print("Array 1:",array1)
print("Array 2:",array2)
print("Stack array horizontally (column wise):")
print(hstacked_arr)
```

OUTPUT:-

```
Array 1: [[1 4]
 [4 6]]
Array 2: [[1 3]
 [2 5]]
Stack array horizontally (column wise):
[[1 4 1 3]
 [4 6 2 5]]
```

Practical-4

Practical-4.1

AIM:- WRITE A NUMPY PROGRAM TO ADD, SUBTRACT, MULTIPLY, DIVIDE ARGUMENTS ELEMENT-WISE.

INPUT:-

```
import numpy as np
array1=np.array([10,40,60])
array2=np.array([50,20,30])
print("Addition:",np.add(array1,array2))
print("Subtraction:",np.subtract(array1,array2))
print("Multiplication:",np.multiply(array1,array2))
print("Division:",np.divide(array1,array2))
```

OUTPUT:-

```
Addition: [60 60 90]
Subtraction: [-40  20  30]
Multiplication: [ 500  800 1800]
Division: [0.2 2.  2. ]
```

Practical-4.2

AIM:- WRITE A NUMPY PROGRAM TO ROUND ELEMENTS OF THE ARRAY TO THE NEAREST INTEGER.

INPUT:-

```
import numpy as np

myarray=np.array([1.2,3.4,5.6,7.8,9.0])

round_array=np.round(myarray)

print("Original Array:")

print(myarray)

print("Rounded Array:")

print(round_array)
```

OUTPUT:-

```
Original Array:
[1.2 3.4 5.6 7.8 9. ]
Rounded Array:
[1. 3. 6. 8. 9.]
```

Practical-4.3

AIM:- WRITE A NUMPY PROGRAM TO CALCULATE MEAN ACROSS DIMENSION, IN A 2D NUMPY ARRAY.

INPUT:-

```
import numpy as np
myarray=np.array([[1,2,3],
[4,5,6],
[7,8,9]])
mean_across_rows=np.mean(myarray,axis=0)
print("Original Array:",myarray)
print("Mean Across Rows:",mean_across_rows)
mean_across_columns=np.mean(myarray,axis=1)
print("Mean Across Columns:",mean_across_columns)
```

OUTPUT:-

```
Original Array: [[1 2 3]
 [4 5 6]
 [7 8 9]]
Mean Across Rows: [4. 5. 6.]
Mean Across Columns: [2. 5. 8.]
```

Practical-4.4

AIM:- WRITE A NUMPY PROGRAM TO CALCULATE THE DIFFERENCE BETWEEN NEIGHBORING ELEMENTS, ELEMENT-WISE OF A GIVEN ARRAY.

INPUT:-

```
import numpy as np
array_1d=np.array([10,20,30,40])
diff_1d=np.diff(array_1d)
print("Original 1D array:",array_1d)
print("1D array Differences:",diff_1d)
array_2d=np.array([[1,2,3],
[4,5,6],
[7,8,9]])
diff_2d_row=np.diff(array_2d,axis=1)
print("Original 2D array:",array_2d)
print("2D array row differences:\n",diff_2d_row)
diff_2d_col=np.diff(array_2d,axis=0)
print("2D array columns differences:\n",diff_2d_col)
```

OUTPUT:-

```
Original 1D array: [10 20 30 40]
1D array Differences: [10 10 10]
Original 2D array: [[1 2 3]
 [4 5 6]
 [7 8 9]]
2D array row differences:
 [[1 1]
 [1 1]
 [1 1]]
2D array columns differences:
 [[3 3 3]
 [3 3 3]]
```

Practical-5

Practical-5.1

AIM:- WRITE A NUMPY PROGRAM TO FIND THE MAXIMUM AND MINIMUM VALUE OF A GIVEN FLATTENED ARRAY.

INPUT:-

```
import numpy as np

myarray=np.array([60,20,30,81,50,40,10])

minarray=np.min(myarray)

print("Original Array:",myarray)

print("Minimum value:",minarray)

maxarray=np.max(myarray)

print("Maximum value:",maxarray)
```

OUTPUT:-

```
Original Array: [60 20 30 81 50 40 10]
Minimum value: 10
Maximum value: 81
```

Practical-5.2

AIM:- WRITE A NUMPY PROGRAM TO COMPUTE THE MEAN, STANDARD DEVIATION, AND VARIANCE OF A GIVEN ARRAY ALONG THE SECOND AXIS.

INPUT:-

```
import numpy as np
myarray=np.array([[10,20,30],
                  [40,50,60],
                   [70,80,90]])
mean_axis1=np.mean(myarray,axis=1)
print("Original Array:",myarray)
print("Mean along the second axis:",mean_axis1)
std_axis1=np.std(myarray,axis=1)
print("Standard deviation along the second axis:",std_axis1)
var_axis1=np.var(myarray,axis=1)
print("variance along the second axis:",var_axis1)
```

OUTPUT:-

```
Original Array: [[10 20 30]
 [40 50 60]
 [70 80 90]]
Mean along the second axis: [20. 50. 80.]
Standard deviation along the second axis: [8.16496581 8.16496581 8.16496581]
variance along the second axis: [66.66666667 66.66666667 66.66666667]
```

Practical-6

Practical-6.1

AIM:- WRITE A PANDAS PROGRAM TO CONVERT A NUMPY ARRAY TO A PANDAS SERIES.

INPUT:-

```
import pandas as pd

import numpy as np

myarray=np.array([10,20,30,40])

series=pd.Series(myarray)

print("Array:",myarray)

print(series)
```

OUTPUT:-

```
Array: [10 20 30 40]
0    10
1    20
2    30
3    40
dtype: int64
```

Practical-6.2

AIM:- WRITE A PANDAS PROGRAM TO CONVERT THE FIRST COLUMN OF A DATAFRAME AS A SERIES.

INPUT:-

```
import pandas as pd
data={
   'A':[10,20,30],
   'B':[40,50,60],
   'C':[70,80,90]
}
df=pd.DataFrame(data)
first_column_series=df['A']
print("DataFrame:",df)
print(first_column_series)
```

OUTPUT:-

```
DataFrame:     A   B   C
0  10  40  70
1  20  50  80
2  30  60  90
0    10
1    20
2    30
Name: A, dtype: int64
```

Practical-6.3

AIM:- WRITE A PANDAS PROGRAM TO CREATE THE MEAN AND STANDARD DEVIATION OF THE DATA OF A GIVEN SERIES.

INPUT:-

```
import pandas as pd

data=pd.Series([10,40,50,60,30])

mean_value=data.mean()

std_value=data.std()

print("Original dataseries:\n",data)

print("Mean value:",mean_value)

print("Standard deviation:",std_value)
```

OUTPUT:-

```
Original dataseries:
 0    10
1    40
2    50
3    60
4    30
dtype: int64
Mean value: 38.0
Standard deviation: 19.235384061671343
```

Practical-6.3

AIM:- WRITE A PANDAS PROGRAM TO SORT A GIVEN SERIES.

INPUT:-

```
import pandas as pd
data=pd.Series([30,70,10,50,20,40,80,60])
sorted_series_asc=data.sort_values()
sorted_series_desc=data.sort_values(ascending=False)
print("Original Series:\n",data)
print("Sorted series in ascending:")
print(sorted_series_asc)
print("Sorted series in descending:")
print(sorted_series_desc)
```

OUTPUT:-

```
Original Series:
 0    30
1    70
2    10
3    50
4    20
5    40
6    80
7    60
dtype: int64
Sorted series in ascending:
2    10
4    20
0    30
5    40
3    50
7    60
1    70
6    80
dtype: int64
Sorted series in descending:
6    80
1    70
7    60
3    50
5    40
0    30
4    20
2    10
dtype: int64
```

Practical-7

Practical-7.1

AIM:- WRITE A PANDAS PROGRAM TO CREATE A DATAFRAME FROM A DICTIONARY AND DISPLAY IT.

INPUT:-

```
import pandas as pd
data={
"Name":['Harshi;','Jayveer','Vatsal'],
"Er No":[41,11,74],
"Age":[18,17,18]
}
df=pd.DataFrame(data)
print(df)
```

OUTPUT:-

```
       Name  Er No  Age
0  Harshi;     41   18
1  Jayveer     11   17
2   Vatsal     74   18
```

Practical-7.2

AIM:- WRITE A PANDAS PROGRAM TO SORT THE DATAFRAME FIRST BY 'NAME' IN ASCENDING ORDER.

INPUT:-

```
import pandas as pd
data={
'Name':['Harshil','Jayveer','Vatsal','Parvej','Sohail'],
'Enrollment No':[41,11,74,33,32],
'City':['Kosamba','Kosamba','Kosambs','Anklehswer','Anklehswer']
}
df=pd.DataFrame(data)
sorted_data=df.sort_values(by='Name',ascending=True)
print(sorted_data)
```

OUTPUT:-

```
       Name  Enrollment No        City
0  Harshil             41     Kosamba
1  Jayveer             11     Kosamba
3   Parvej             33  Anklehswer
4   Sohail             32  Anklehswer
2   Vatsal             74     Kosambs
```

Practical-7.3

AIM:- WRITE A PANDAS PROGRAM TO DELETE THE ONE SPECIFIC COLUMN FROM THE DATAFRAME.

INPUT:-

```
import pandas as pd
data={
"Name":['Harshil','Jayveer','Vatsal','Parvej','Sohail'],
"Enrollment No":[41,11,74,33,32],
"City":['Kosamba','Kosamba','Kosambs','Anklehswer','Anklehswer']
}
df=pd.DataFrame(data)
data_drop=df.drop(columns=["City"])
print(data_drop)
```

OUTPUT:-

```
      Name  Enrollment No
0  Harshil             41
1  Jayveer             11
2   Vatsal             74
3   Parvej             33
4   Sohail             32
```

Practical-7.4

AIM:- WRITE A PANDAS PROGRAM TO WRITE A DATAFRAME TO CSV FILE USING TAB SEPARATOR.

INPUT:-

```
import pandas as pd
data = {
'Name': ['Harshil','Jayveer','Vatsal','Parvej','Sohail'],
'Enrollment No': [41, 11, 74,33,32],
'City': ['Kosamba','Kosamba','Kosambs','Anklehswer','Anklehswer']
}
df = pd.DataFrame(data)
df.to_csv('output.csv', sep='\t', index=False)
df_read = pd.read_csv('output.csv', sep='\t')
print(df_read)
```

OUTPUT:-

```
      Name  Enrollment No       City
0  Harshil             41     Kosamba
1  Jayveer             11     Kosamba
2   Vatsal             74     Kosambs
3   Parvej             33  Anklehswer
4   Sohail             32  Anklehswer
```

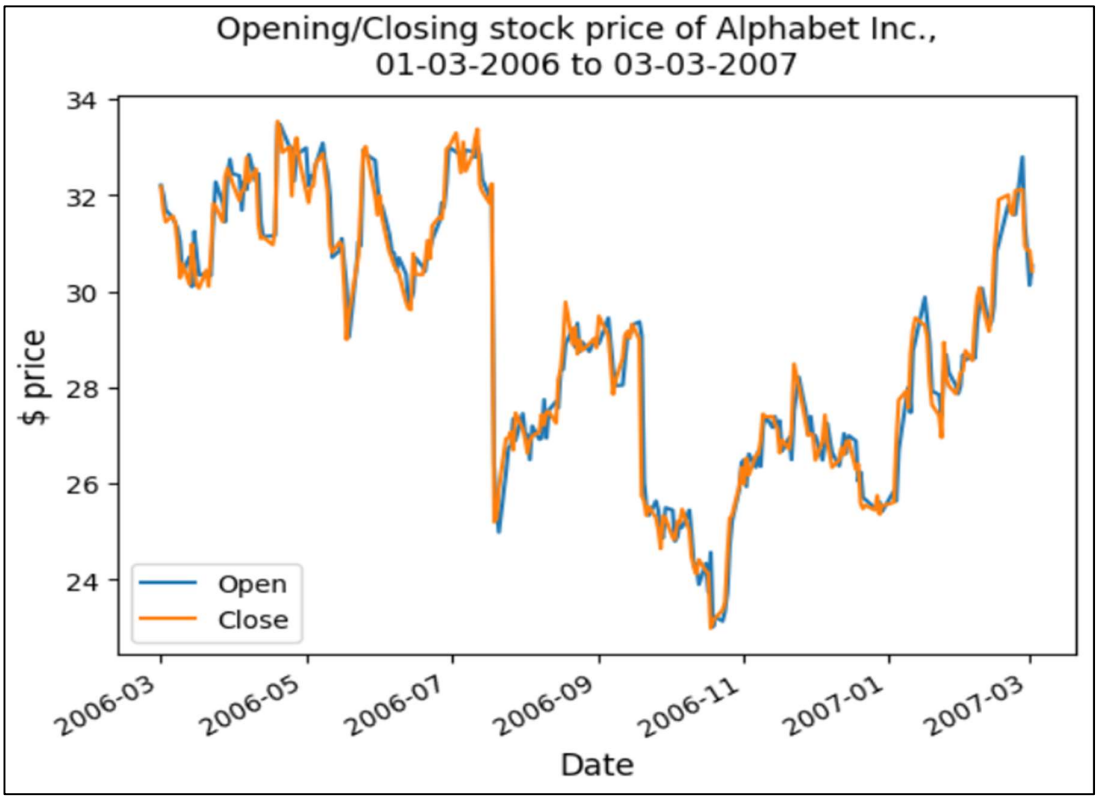
Practical-8

AIM:- WRITE A PANDAS PROGRAM TO CREATE A LINE PLOT OF THE OPENING, CLOSING STOCK PRICES OF GIVEN COMPANY BETWEEN TWO SPECIFIC DATES.

INPUT:-

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('stock_data.csv')
df['Date'] = pd.to_datetime(df['Date'], dayfirst=True)
start_date = pd.to_datetime("03-06-2024", dayfirst=True)
end_date = pd.to_datetime("31-07-2024", dayfirst=True)
new_df = (df['Date'] >= start_date) & (df['Date'] <= end_date)
df2 = df[new_df]
plt.figure(figsize=(10, 10))
df2.plot(x='Date', y=['Open', 'Close'])
plt.suptitle('Opening/Closing stock price of Alphabet Inc., \n 03-06-2024 to 31-07-2024')
plt.xlabel("Date", fontsize=12, color="black")
plt.ylabel("$ price", fontsize=12, color="black")
plt.show()
```

OUTPUT:-



Practical-9

AIM:- WRITE A PANDAS PROGRAM TO IMPLEMENT FOLLOWING OPERATION TO PRINT AND CONVERT DATA TYPES.

INPUT:-

Description:

To convert the data type of a data frame follows the steps below:

- Import module
- Create data frame
- Check data type
- Convert data type using convert_dtypes().dtypes function

The data type of columns are changed accordingly. But the datatype of dataframe will remain object because it contains multiple columns with each column has a different datatype.

Program:

```
import pandas as pd
import numpy as np

# creating a dataframe
df = pd.DataFrame({"Roll_No.": ([1, 2, 3]),
                   "Name": ["Raj", "Ritu", "Rohan"],
                   "Result": ["Pass", "Fail", np.nan],
                   "Promoted": [True, False, np.nan],
                   "Marks": [90.33, 30.6, np.nan]})

# printing the dataframe
print("PRINTING DATAFRAME")
display(df)
# checking datatype
print()
print("PRINTING DATATYPE")
print(df.dtypes)
```

```
# converting datatype
print()
print("AFTER CONVERTING DATATYPE")
print(df.convert_dtypes().dtypes)
```

OUTPUT:-

```
PRINTING DATAFRAME
   Roll_No.   Name  Result  Promoted  Marks
0         1    Raj    Pass      True  90.33
1         2   Ritu    Fail     False  30.60
2         3  Rohan     NaN       NaN    NaN

PRINTING DATATYPE
Roll_No.      int64
Name         object
Result       object
Promoted     object
Marks       float64
dtype: object

AFTER CONVERTING DATATYPE
Roll_No.             Int64
Name        string[python]
Result      string[python]
Promoted           boolean
Marks              Float64
dtype: object
```

Practical-10

Practical-10.1

AIM:- WRITE A PANDAS PROGRAM TO FIND AND DROP THE MISSING VALUES FROM THE GIVEN DATASET.

INPUT:-

```
import pandas as pd
mis = pd.read_csv('user_data.csv')
print("4th semester result:")
print(mis)
print("Missing values in result:")
print(mis.isnull())
print("Drop missing values in result:")
print(mis.dropna())
```

user_data.csv:-

Sr. No.	Name	IWD	AOOP	ISE	CN	SPI
1	Harshil Maisuria	AA	BB	AA	AB	9.5
2	Jayveer chauhan	AB	AA		AB	9.75
3	Amaan Malek	AA		AA	AA	10
4	Sneh pamer	BB	BC	CD	FF	4.55
5	Vatsal pamer	AB		BB	BB	8.85
6	Abhi aamir	AB	AA	AB	AB	9.15
7	Hanif	AA	AB	AA		9.65
8	Saad Khwaja	BB	BC	AB	BC	8.32
9	Sohil Khan	AA		AB	BC	8.63
10	Parvej Khatri	AA	BB	FF		3.52

OUTPUT:-

```
4th semester result:
   Sr. No.              Name IWD AOOP   ISE   CN    SPI
0        1  Harshil Maisuria  AA    BB   AA   AB   9.50
1        2   Jayveer chauhan  AB    AA  NaN   AB   9.75
2        3       Amaan Malek  AA   NaN   AA   AA  10.00
3        4        Sneh pamer  BB    BC   CD   FF   4.55
4        5      Vatsal pamer  AB   NaN   BB   BB   8.85
5        6        Abhi aamir  AB    AA   AB   AB   9.15
6        7             Hanif  AA    AB   AA  NaN   9.65
7        8       Saad Khwaja  BB    BC   AB   BC   8.32
8        9        Sohil Khan  AA   NaN   AB   BC   8.63
9       10     Parvej Khatri  AA    BB   FF  NaN   3.52
Missing values in result:
   Sr. No.   Name    IWD  AOOP     ISE     CN    SPI
0    False  False  False  False  False  False  False
1    False  False  False  False   True  False  False
2    False  False  False   True  False  False  False
3    False  False  False  False  False  False  False
4    False  False  False   True  False  False  False
5    False  False  False  False  False  False  False
6    False  False  False  False  False   True  False
7    False  False  False  False  False  False  False
8    False  False  False   True  False  False  False
9    False  False  False  False  False   True  False
Drop missing values in result:
   Sr. No.              Name IWD AOOP  ISE  CN   SPI
0        1  Harshil Maisuria  AA    BB  AA  AB  9.50
3        4        Sneh pamer  BB    BC  CD  FF  4.55
5        6        Abhi aamir  AB    AA  AB  AB  9.15
7        8       Saad Khwaja  BB    BC  AB  BC  8.32
```

Practical-10.2

AIM:- WRITE A PANDAS PROGRAM TO REMOVE THE DUPLICATES FROM THE GIVEN DATASET.

INPUT:-

```
import pandas as pd
dup = pd.read_csv('user_data.csv')
print("4th semester result:")
print(dup)
print("Result after droping duplicate SPI")
print(dup.drop_duplicates('IWD'))
```

user_data.csv:-

Sr. No.	Name	IWD	AOOP	ISE	CN	SPI
1	Harshil Maisuria	AA	BB	AA	AB	9.5
2	Jayveer chauhan	AB	AA	AB	AB	9.75
3	Amaan Malek	AA	AA	AA	AA	10
4	Sneh pamer	BB	BC	CD	FF	4.55
5	Vatsal pamer	AB	BB	BB	BB	8.85
6	Abhi aamir	AB	AA	AB	AB	9.15
7	Hanif	AA	AB	AA	AB	9.65
8	Saad Khwaja	BB	BC	AB	BC	8.32
9	Sohil Khan	AA	AB	AB	BC	8.63
10	Parvej Khatri	AA	BB	FF	FF	3.52

OUTPUT:-

```
4th semester result:
   Sr. No.              Name IWD AOOP  ISE  CN    SPI
0        1  Harshil Maisuria  AA    BB  AA  AB   9.50
1        2   Jayveer chauhan  AB    AA  AB  AB   9.75
2        3       Amaan Malek  AA    AA  AA  AA  10.00
3        4        Sneh pamer  BB    BC  CD  FF   4.55
4        5      Vatsal pamer  AB    BB  BB  BB   8.85
5        6        Abhi aamir  AB    AA  AB  AB   9.15
6        7             Hanif  AA    AB  AA  AB   9.65
7        8       Saad Khwaja  BB    BC  AB  BC   8.32
8        9        Sohil Khan  AA    AB  AB  BC   8.63
9       10     Parvej Khatri  AA    BB  FF  FF   3.52
Result after droping duplicate SPI
   Sr. No.              Name IWD AOOP  ISE  CN   SPI
0        1  Harshil Maisuria  AA    BB  AA  AB  9.50
1        2   Jayveer chauhan  AB    AA  AB  AB  9.75
3        4        Sneh pamer  BB    BC  CD  FF  4.55
```

Practical-11

AIM:- WRITE A PANDAS PROGRAM TO FILTER ALL COLUMNS WHERE ALL ENTRIES PRESENT, CHECK WHICH ROWS AND COLUMNS HAS A NAN AND FINALLY DROP ROWS WITH ANY NANS FROM THE GIVEN DATASET.

INPUT:-

```
import pandas as pd
result = pd.read_csv('user_data.csv')
print("4th semester result:")
print(result)
print("Column with all values present:")
print(result.loc[:, result.notnull().all()])
print("Columns having missing values:")
print(result.loc[:, result.isnull().any()])
print("Dropping rows having any null values:")
print(result.dropna(how='any'))
```

user_data.csv:-

Sr. No.	Name	IWD	AOOP	ISE	CN	SPI
1	Harshil Maisuria	AA	BB	AA	AB	9.5
2	Jayveer chauhan	AB	AA		AB	9.75
3	Amaan Malek	AA	AA	AA		10
4	Sneh pamer	BB	BC		FF	4.55
5	Vatsal pamer	AB	BB	BB	BB	8.85
6	Abhi aamir	AB	AA	AB	AB	9.15
7	Hanif	AA	AB		AB	9.65
8	Saad Khwaja	BB	BC	AB	BC	8.32
9	Sohil Khan	AA	AB	AB		8.63
10	Parvej Khatri	AA	BB	FF	FF	3.52
11	Ujef	AA		AA	AA	10
12	Dhruvil	BC	FF		BC	4.25
13	Atif	AB		BB	BB	8.96
14	Raj Singh	AB	FF		FF	3.25
15	Harshil Patel	BB		BC	AA	8.65
16	Umer	AB		AA	AA	9.69
17	Taiyab	BB	BB		BC	8.76
18	Smit	AB		FF	FF	2.33
19	Amaan	BB	BC		BB	8.56
20	SachinPatel	AB	BB	BC		8.32

OUTPUT:-

```
4th semester result:
    Sr. No.              Name IWD AOOP   ISE   CN    SPI
0         1  Harshil Maisuria  AA    BB   AA   AB   9.50
1         2   Jayveer chauhan  AB    AA  NaN   AB   9.75
2         3       Amaan Malek  AA    AA   AA  NaN  10.00
3         4        Sneh pamer  BB    BC  NaN   FF   4.55
4         5      Vatsal pamer  AB    BB   BB   BB   8.85
5         6        Abhi aamir  AB    AA   AB   AB   9.15
6         7             Hanif  AA    AB  NaN   AB   9.65
7         8       Saad Khwaja  BB    BC   AB   BC   8.32
8         9        Sohil Khan  AA    AB   AB  NaN   8.63
9        10     Parvej Khatri  AA    BB   FF   FF   3.52
10       11              Ujef  AA   NaN   AA   AA  10.00
11       12          Dhruvil   BC    FF  NaN   BC   4.25
12       13              Atif  AB   NaN   BB   BB   8.96
13       14         Raj Singh  AB    FF  NaN   FF   3.25
14       15     Harshil Patel  BB   NaN   BC   AA   8.65
15       16              Umer  AB   NaN   AA   AA   9.69
16       17           Taiyab   BB    BB  NaN   BC   8.76
17       18              Smit  AB   NaN   FF   FF   2.33
18       19             Amaan  BB    BC  NaN   BB   8.56
19       20       SachinPatel  AB    BB   BC  NaN   8.32
Column with all values present:
    Sr. No.              Name IWD    SPI
0         1  Harshil Maisuria  AA   9.50
1         2   Jayveer chauhan  AB   9.75
2         3       Amaan Malek  AA  10.00
3         4        Sneh pamer  BB   4.55
4         5      Vatsal pamer  AB   8.85
5         6        Abhi aamir  AB   9.15
6         7             Hanif  AA   9.65
7         8       Saad Khwaja  BB   8.32
8         9        Sohil Khan  AA   8.63
9        10     Parvej Khatri  AA   3.52
10       11              Ujef  AA  10.00
11       12          Dhruvil   BC   4.25
12       13              Atif  AB   8.96
13       14         Raj Singh  AB   3.25
14       15     Harshil Patel  BB   8.65
15       16              Umer  AB   9.69
16       17           Taiyab   BB   8.76
17       18              Smit  AB   2.33
18       19             Amaan  BB   8.56
19       20       SachinPatel  AB   8.32
Columns having missing values:
   AOOP   ISE   CN
0    BB   AA   AB
1    AA  NaN   AB
2    AA   AA  NaN
3    BC  NaN   FF
4    BB   BB   BB
5    AA   AB   AB
6    AB  NaN   AB
7    BC   AB   BC
8    AB   AB  NaN
9    BB   FF   FF
10  NaN   AA   AA
11   FF  NaN   BC
```

Practical-12

AIM:- WRITE A PYTHON PROGRAM USING SCIKIT-LEARN TO PRINT THE KEYS, NUMBER OF ROWS-COLUMNS, FEATURE NAMES AND THE DESCRIPTION OF THE GIVEN DATA.

INPUT:-

```
import pandas as pd
data = pd.read_csv('practical11.csv')
print("Dataset: ")
print(data)
print("\nAll the column name in dataset:")
print(data.keys())
print("No. of rows and column in the the data set:")
print(data.shape)
```

OUTPUT:-

```
Dataset:
    Sr. No.              Name IWD AOOP   ISE   CN    SPI
0         1  Harshil Maisuria  AA    BB   AA   AB   9.50
1         2   Jayveer chauhan  AB    AA  NaN   AB   9.75
2         3       Amaan Malek  AA    AA   AA  NaN  10.00
3         4        Sneh pamer  BB    BC  NaN   FF   4.55
4         5      Vatsal pamer  AB    BB   BB   BB   8.85
5         6        Abhi aamir  AB    AA   AB   AB   9.15
6         7             Hanif  AA    AB  NaN   AB   9.65
7         8       Saad Khwaja  BB    BC   AB   BC   8.32
8         9        Sohil Khan  AA    AB   AB  NaN   8.63
9        10     Parvej Khatri  AA    BB   FF   FF   3.52
10       11              Ujef  AA   NaN   AA   AA  10.00
11       12          Dhruvil   BC    FF  NaN   BC   4.25
12       13              Atif  AB   NaN   BB   BB   8.96
13       14         Raj Singh  AB    FF  NaN   FF   3.25
14       15     Harshil Patel  BB   NaN   BC   AA   8.65
15       16              Umer  AB   NaN   AA   AA   9.69
16       17           Taiyab   BB    BB  NaN   BC   8.76
17       18              Smit  AB   NaN   FF   FF   2.33
18       19             Amaan  BB    BC  NaN   BB   8.56
19       20       SachinPatel  AB    BB   BC  NaN   8.32

All the column name in dataset:
Index(['Sr. No.', 'Name', 'IWD', 'AOOP ', 'ISE', 'CN', 'SPI'], dtype='object')
No. of rows and column in the the data set:
(20, 7)
```

Practical-13

AIM:- WRITE A PYTHON PROGRAM TO IMPLEMENT K-NEAREST NEIGHBOUR SUPERVISED MACHINE LEARNING ALGORITHM FOR GIVEN DATASET.

INPUT:-

```
# Import necessary modules
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
import numpy as np
import matplotlib.pyplot as plt
irisData = load_iris()
# Create feature and target arrays
X = irisData.data
y = irisData.target
# Split into training and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=42)
neighbors = np.arange(1, 9)
train_accuracy = np.empty(len(neighbors))
test_accuracy = np.empty(len(neighbors))
# Loop over K values
for i, k in enumerate(neighbors):
  knn = KNeighborsClassifier(n_neighbors=k)
  knn.fit(X_train, y_train)
# Compute training and test data accuracy
  train_accuracy[i] = knn.score(X_train, y_train)
  test_accuracy[i] = knn.score(X_test, y_test)
# Generate plot
plt.plot(neighbors, test_accuracy, label = 'Testing dataset Accuracy')
plt.plot(neighbors, train_accuracy, label = 'Training dataset Accuracy')
```

```
plt.legend()
plt.xlabel('n_neighbors')
plt.ylabel('Accuracy')
plt.show()
```

OUTPUT:-