Name: Ishwari Parag Gawade Class: SY MCA (Div. B, Roll no. 83)

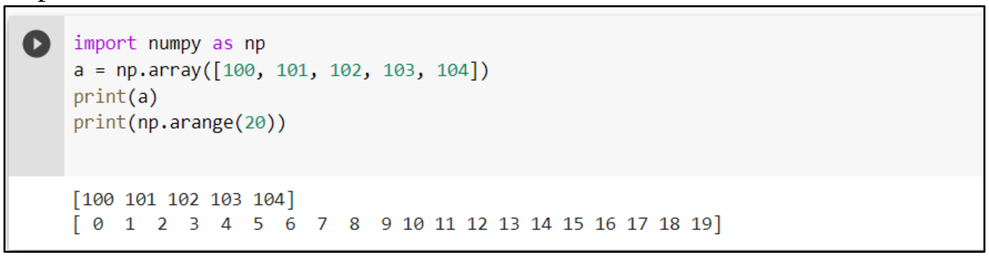
Practical No 1:

Introduction to Python programming and various librariesused for machine learning.

1) NumPy

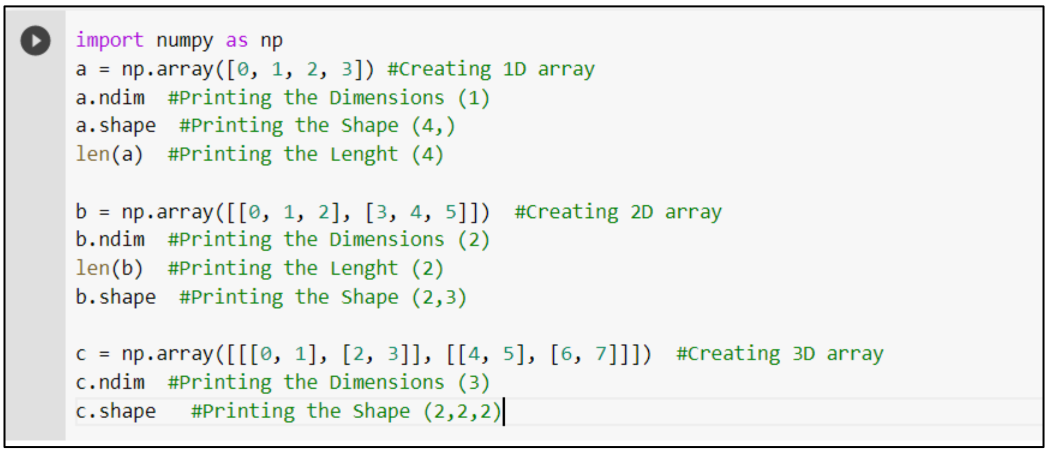
NumPy is a well-known general-purpose array-processing package. An extensive collection of highcomplexity mathematical functions makes NumPy powerful to process large multi-dimensional arraysand matrices.

Exp:



```
import numpy as np
a = np.array([100, 101, 102, 103, 104])
print(a)
print(np.arange(20))
```

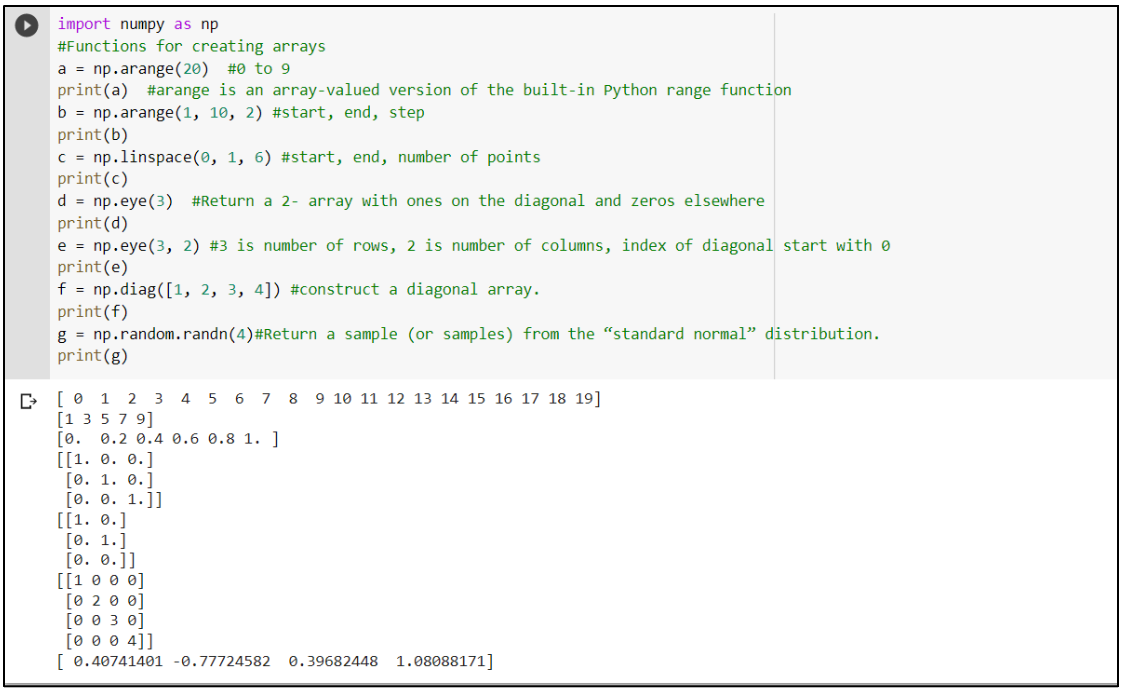
```
[100 101 102 103 104]
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
```



```
import numpy as np
a = np.array([0, 1, 2, 3]) #Creating 1D array
a.ndim  #Printing the Dimensions (1)
a.shape  #Printing the Shape (4,)
len(a)  #Printing the Lenght (4)

b = np.array([[0, 1, 2], [3, 4, 5]])  #Creating 2D array
b.ndim  #Printing the Dimensions (2)
len(b)  #Printing the Lenght (2)
b.shape  #Printing the Shape (2,3)

c = np.array([[[0, 1], [2, 3]], [[4, 5], [6, 7]]])  #Creating 3D array
c.ndim  #Printing the Dimensions (3)
c.shape   #Printing the Shape (2,2,2)
```



```
import numpy as np
#Functions for creating arrays
a = np.arange(20)  #0 to 9
print(a)  #arange is an array-valued version of the built-in Python range function
b = np.arange(1, 10, 2) #start, end, step
print(b)
c = np.linspace(0, 1, 6) #start, end, number of points
print(c)
d = np.eye(3)  #Return a 2- array with ones on the diagonal and zeros elsewhere
print(d)
e = np.eye(3, 2) #3 is number of rows, 2 is number of columns, index of diagonal start with 0
print(e)
f = np.diag([1, 2, 3, 4]) #construct a diagonal array.
print(f)
g = np.random.randn(4)#Return a sample (or samples) from the “standard normal” distribution.
print(g)
```

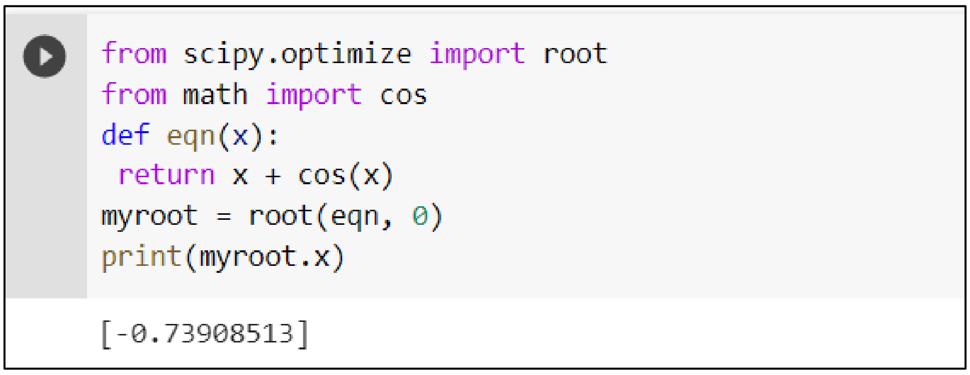
```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
[1 3 5 7 9]
[0.  0.2 0.4 0.6 0.8 1. ]
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
[[1. 0.]
 [0. 1.]
 [0. 0.]]
[[1 0 0 0]
 [0 2 0 0]
 [0 0 3 0]
 [0 0 0 4]]
[ 0.40741401 -0.77724582  0.39682448  1.08088171]
```

2) SciPy

The SciPy library offers modules for linear algebra, image optimization, integration interpolation, special functions, Fast Fourier transform, signal and image processing, Ordinary Differential Equation (ODE) solving, and other computational tasks in science and analytics.
SciPy Installation - pip install scipy

Example

Find root of the equation x + cos(x):



```
from scipy.optimize import root
from math import cos
def eqn(x):
 return x + cos(x)
myroot = root(eqn, 0)
print(myroot.x)
```

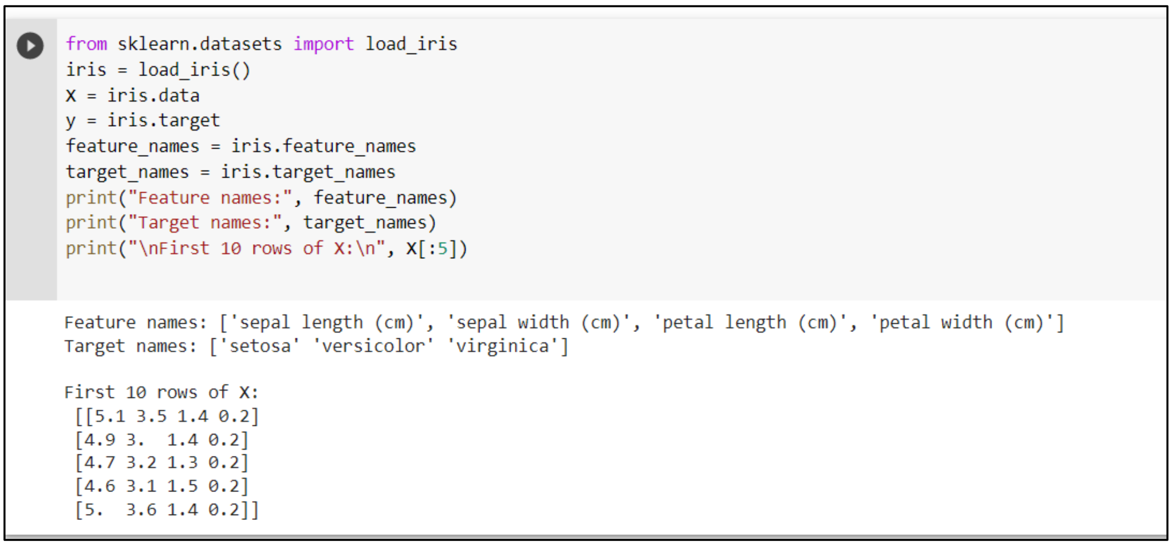
```
[-0.73908513]
```

3) Scikit-learn

Scikit-learn has a wide range of supervised and unsupervised learning algorithms that works on aconsistent interface in Python. The library can also be used for data-mining and data analysis. The main machine learning functions that the Scikit-learn library can handle are classification, regression, clustering, dimensionality reduction, model selection, and pre-processing.

Installation -> pip install -U scikit-learn

Example Following is an example to load iris dataset –

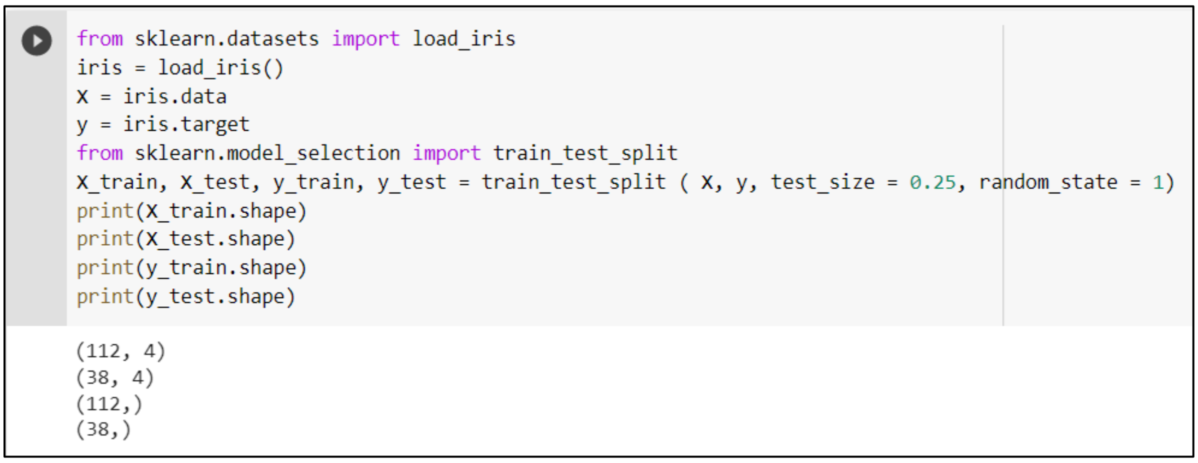


```
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names
target_names = iris.target_names
print("Feature names:", feature_names)
print("Target names:", target_names)
print("\nFirst 10 rows of X:\n", X[:5])
```

```
Feature names: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
Target names: ['setosa' 'versicolor' 'virginica']

First 10 rows of X:
 [[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]]
```

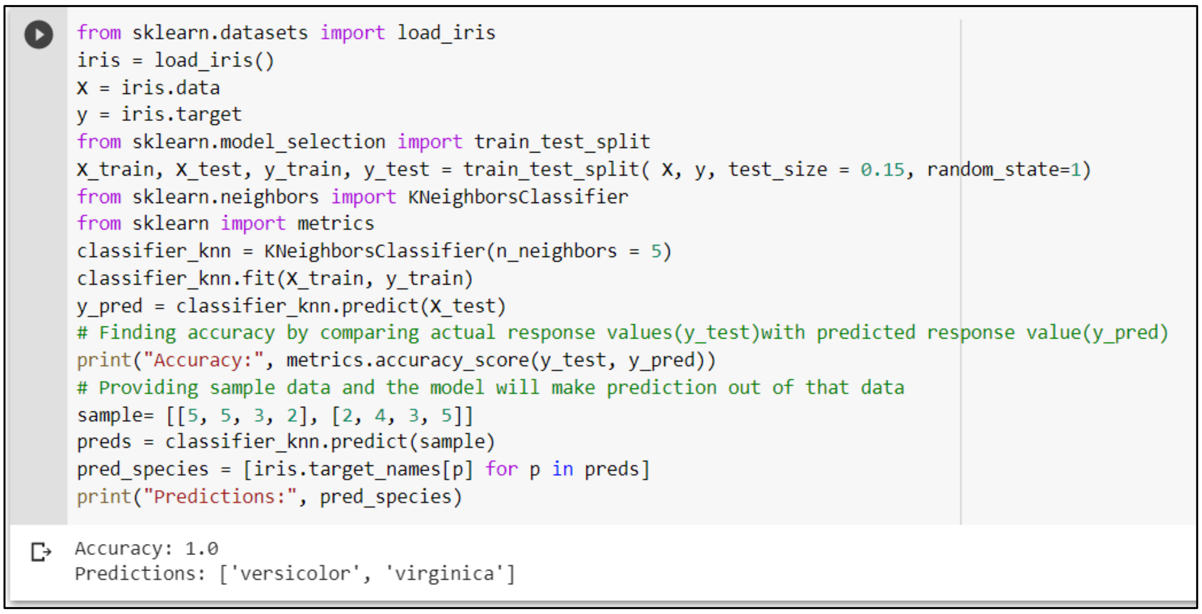
Splitting the dataset -
Example The following example will split the data into 70:30 ratio, i.e. 70% data will be used as training data and 30% will be used as testing data. The dataset is iris dataset as in above example.



```python
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split ( X, y, test_size = 0.25, random_state = 1)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(112, 4)
(38, 4)
(112,)
(38,)
```

Train the Model -
Example -In the example below, we are going to use KNN (K nearest neighbors) classifier. Don't go into the details of KNN algorithms, as there will be a separate chapter for that. This example is used to make you understand the implementation part only.



```python
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size = 0.15, random_state=1)
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
classifier_knn = KNeighborsClassifier(n_neighbors = 5)
classifier_knn.fit(X_train, y_train)
y_pred = classifier_knn.predict(X_test)
# Finding accuracy by comparing actual response values(y_test)with predicted response value(y_pred)
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
# Providing sample data and the model will make prediction out of that data
sample= [[5, 5, 3, 2], [2, 4, 3, 5]]
preds = classifier_knn.predict(sample)
pred_species = [iris.target_names[p] for p in preds]
print("Predictions:", pred_species)
```

```
Accuracy: 1.0
Predictions: ['versicolor', 'virginica']
```

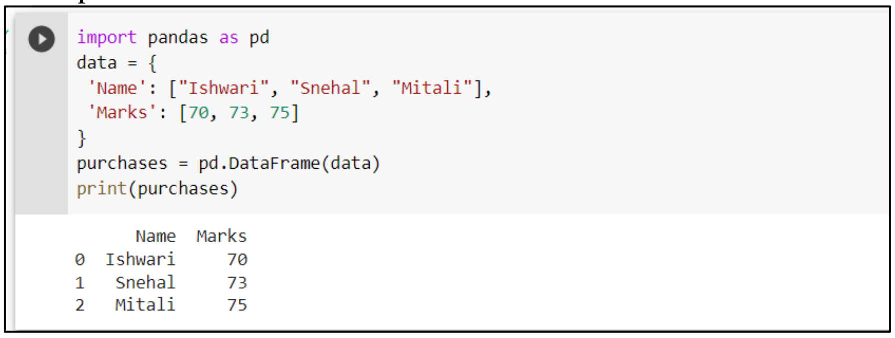
4) Pandas

Pandas are turning up to be the most popular Python library that is used for data analysis with support for fast, flexible, and expressive data structures designed to work on both “relational” or “labeled” data. Through pandas, you get acquainted with your data by cleaning, transforming, and analyzing it.

Install -> pip install pandas

The primary two components of pandas are the Series and DataFrame -
A Series is essentially a column, and a DataFrame is a multi-dimensional table made up of a collection of Series.

Example -



```
import pandas as pd
data = {
 'Name': ["Ishwari", "Snehal", "Mitali"],
 'Marks': [70, 73, 75]
}
purchases = pd.DataFrame(data)
print(purchases)
```

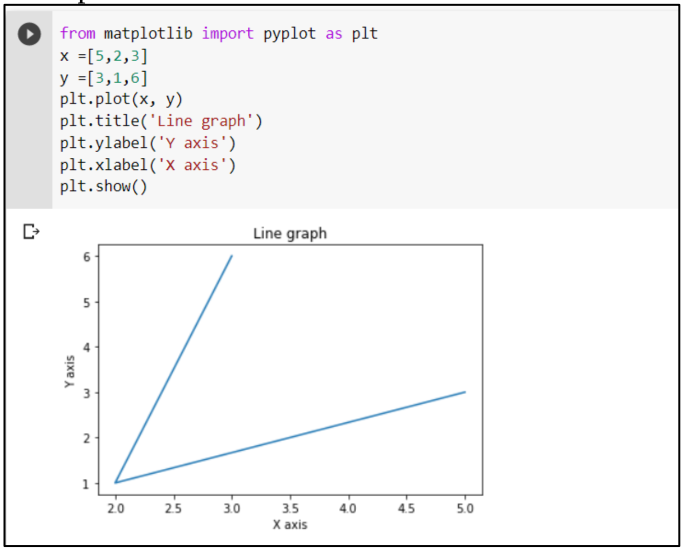
```
      Name  Marks
0  Ishwari     70
1   Snehal     73
2   Mitali     75
```

5) Matplotlib (Python Plotting Library)

Matplotlib is a data visualization library that is used for 2D plotting to produce publication-quality image plots and figures in a variety of formats. The library helps to generate histograms, plots, error charts, scatter plots, bar charts with just a few lines of code.

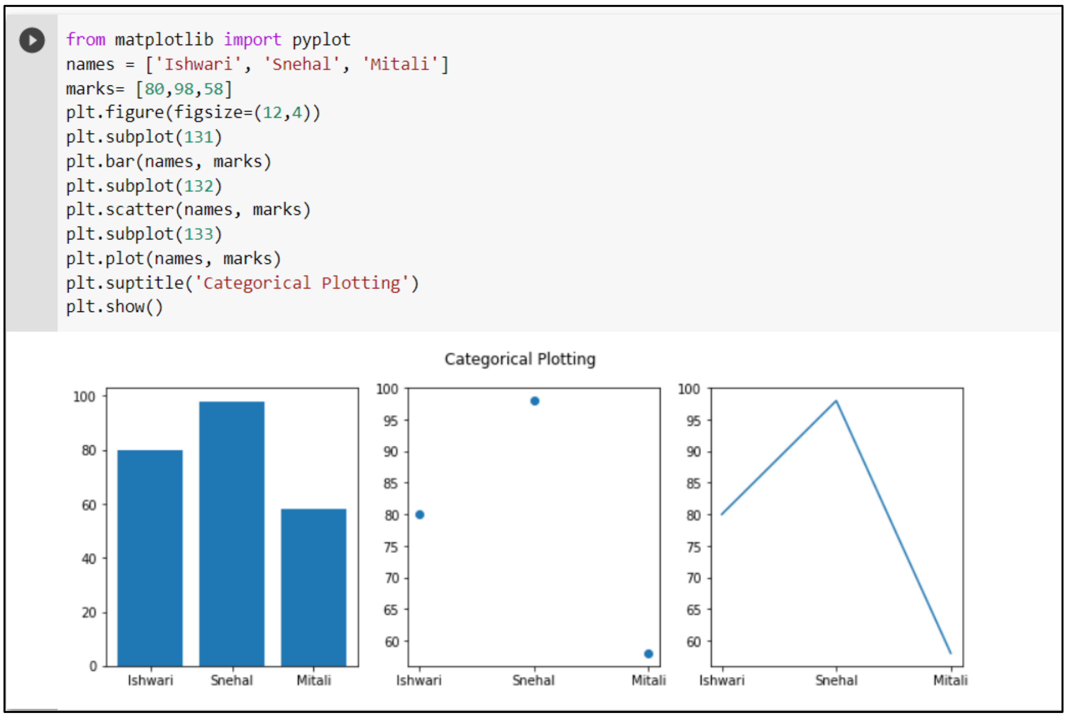
Install Matplotlib: pip install matplotlib

Example-



```
from matplotlib import pyplot as plt
x =[5,2,3]
y =[3,1,6]
plt.plot(x, y)
plt.title('Line graph')
plt.ylabel('Y axis')
plt.xlabel('X axis')
plt.show()
```

Matplotlib allows us to pass categorical variables directly to many plotting functions: consider the following example



```
from matplotlib import pyplot
names = ['Ishwari', 'Snehal', 'Mitali']
marks= [80,98,58]
plt.figure(figsize=(12,4))
plt.subplot(131)
plt.bar(names, marks)
plt.subplot(132)
plt.scatter(names, marks)
plt.subplot(133)
plt.plot(names, marks)
plt.suptitle('Categorical Plotting')
plt.show()
```