PROGRAMMING IN PYTHON
MCA-161
4 Credits (3-0-2)
MCA 5th Sem (2020-21)

R K Dwivedi
Assistant Professor
Department of ITCA
MMMUT Gorakhpur

UNIT IV: Advance Concepts

A. Problem solving:

- Use of Python to solve real time problems
- How Python helps to research problems
- Creating various types of graphs corresponding to any data to show different kinds of results and analysis

B. Data Analysis:

- Understanding problems of data science and machine learning
- Creating codes for data analysis problems in Python
- Other advance programs

A. Problem solving

1. Use of Python to solve real time problems

1. Use of Python to solve real time problems

- Python can be used on a server to create web applications.
- It can be used to create GUI based desktop applications(Games, Scientific and Business Applications).
- It is also used to create test frameworks and multimedia applications.
- It is used to develop operating systems and programming language.
- It can be used to handle image processing, text processing and natural language processing.
- It can be used to create programs for machine learning, deep learning, data science, big data and data analytics applications.
- It can also perform complex mathematics along with all cutting edge technology in software industry.

Organizations and tech-giant companies using Python :

1) Google(Components of Google spider and Search Engine)
2) Yahoo(Maps)
3) YouTube
4) Mozilla
5) Dropbox
6) Microsoft
7) Cisco
8) Spotify
9) Quora
10) Instagram
11)Amazon
12)Facebook
13)Uber etc.

1. Use of Python to solve real time problems

…continued

Some Real Time Projects, their Python Codes and Datasets :

https://data-flair.training/blogs/python-project-ideas/
https://data-flair.training/blogs/django-project-ideas/
https://data-flair.training/blogs/data-science-project-ideas/
https://data-flair.training/blogs/artificial-intelligence-project-ideas/
https://data-flair.training/blogs/machine-learning-project-ideas/
https://data-flair.training/blogs/deep-learning-project-ideas/
https://data-flair.training/blogs/iot-project-ideas/
https://data-flair.training/blogs/computer-vision-project-ideas/
https://archive.ics.uci.edu/ml/datasets.php
https://www.kaggle.com/datasets
https://github.com/topics/covid-19

2. How Python helps to research problems

2. How Python helps to research problems

It can be used in various types of research areas such as:

- Image Processing
- Text Processing
- Natural Language Processing
- Machine Learning
- Deep Learning
- Data Science
- Big Data Analytics

3. Creating various types of graphs corresponding to any data

(to show different kinds of results and analysis)

1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)

- **Matplotlib** is a graph plotting library in python that serves as a visualization utility.
- **NumPy** (Numerical Python) is a python library used for working with arrays.
- **NumPy** also has functions for working in the domain of linear algebra, fourier transform, and matrices.
- **subplot( )** allows to draw multiple plots in one fig. (**subplot**(no of rows, no of columns, index of current plot)
- All modern browsers support 140 color names (**Syntax:** color='r' **or** color='red' **or** c='r' **or** c='red').
- A hexadecimal color is specified with: #RRGGBB (**Syntax:** color='#0000ff' **or** c='0000ff').

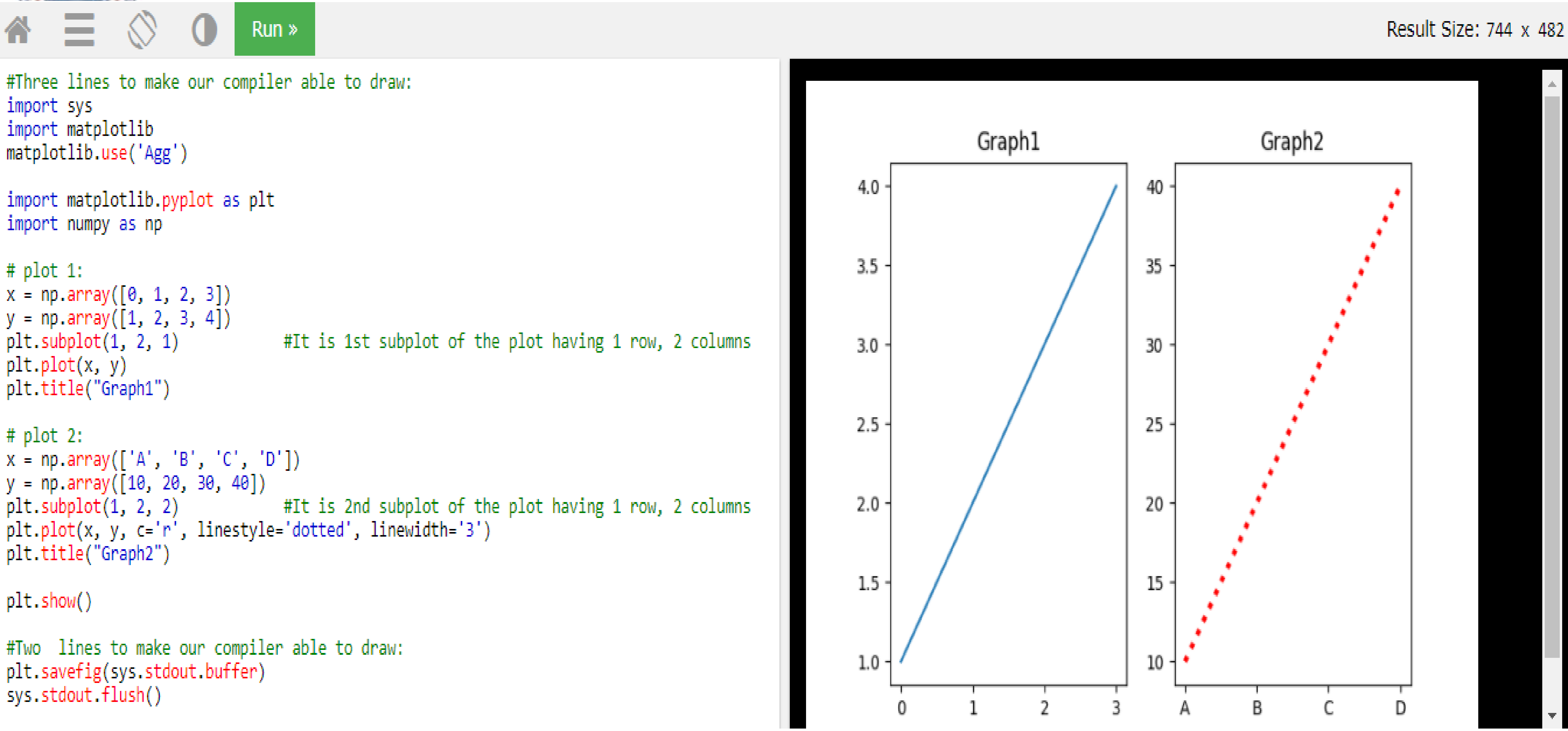
A. Line Graph:

- **linestyle** can be written as **ls** in a shorter syntax.
- **linewidth** can be written as **lw** in a shorter syntax.
- **color** can be written as **c** in a shorter syntax.

linestyle	**short syntax**
solid' **(default)**	'-'
'dotted'	':'
'dashed'	'--'
'dashdot'	'-.'
'None'	'' or ' '

'r'	- Red
'g'	- Green
'b'	- Blue
'c'	- Cyan
'm'	- Magenta
'y'	- Yellow
'k'	- Black
'w'	- White

1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)



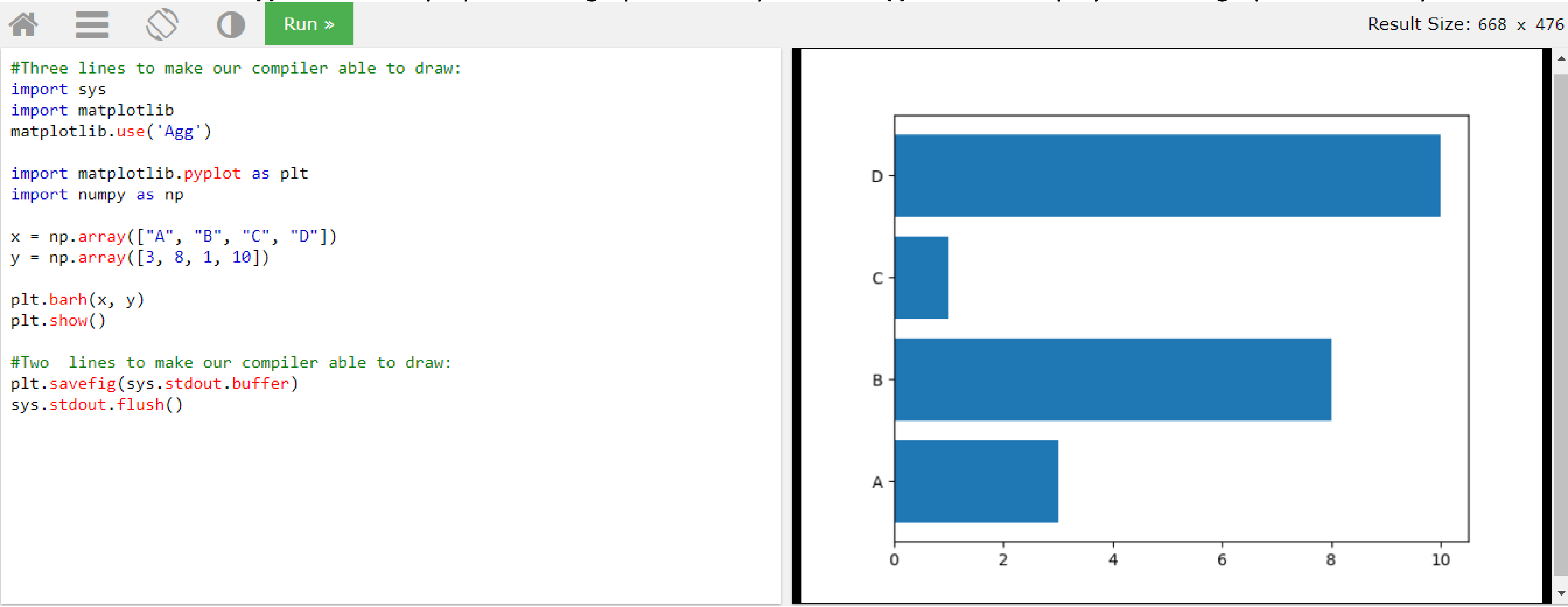
```
#Three lines to make our compiler able to draw:
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

# plot 1:
x = np.array([0, 1, 2, 3])
y = np.array([1, 2, 3, 4])
plt.subplot(1, 2, 1)          #It is 1st subplot of the plot having 1 row, 2 columns
plt.plot(x, y)
plt.title("Graph1")

# plot 2:
x = np.array(['A', 'B', 'C', 'D'])
y = np.array([10, 20, 30, 40])
plt.subplot(1, 2, 2)          #It is 2nd subplot of the plot having 1 row, 2 columns
plt.plot(x, y, c='r', linestyle='dotted', linewidth='3')
plt.title("Graph2")

plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)

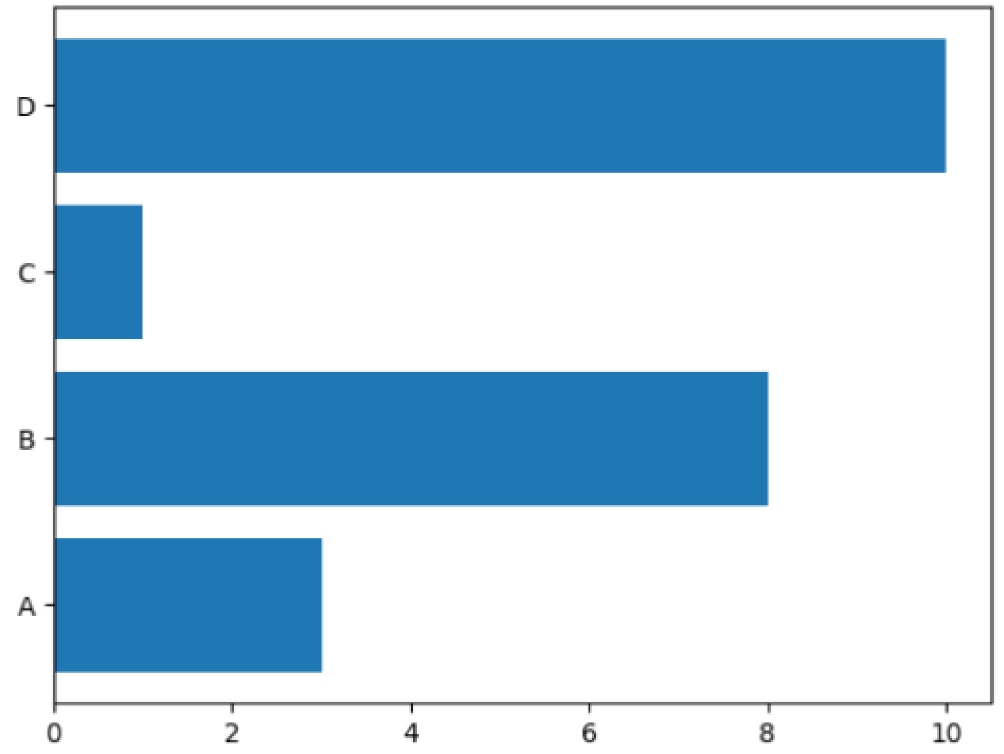
…continued

B. Bar Graph:

- The default width value of the bars is **0.8**.
- **bar ( )** function displays the bar graph vertically and **barh( )** function displays the bar graph horizontally.



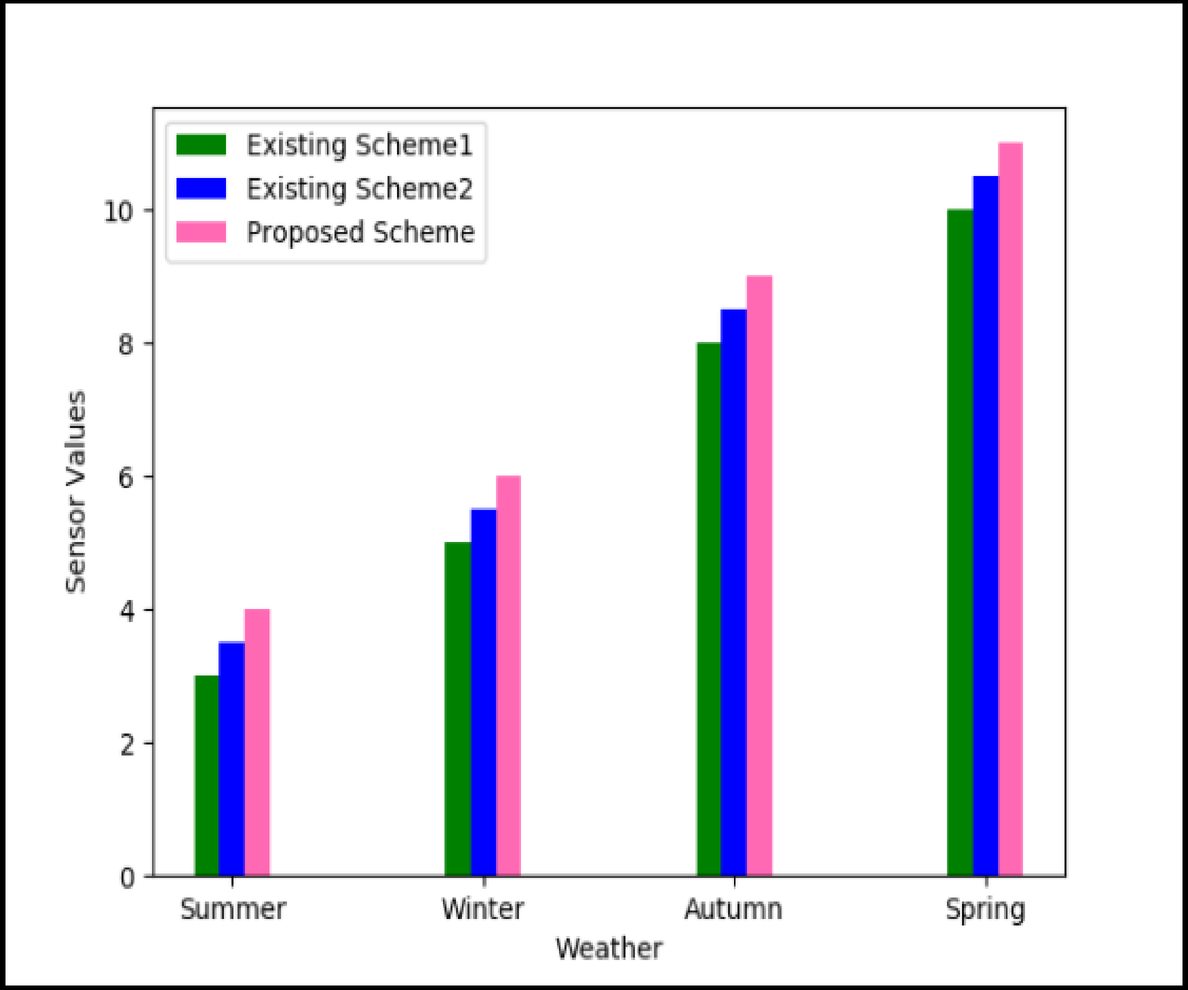
```
#Three lines to make our compiler able to draw:
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.barh(x, y)
plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```



1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis) …continued

Run » Result Size: 668 x 476

```
#Three lines to make our compiler able to draw:
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

x = np.array([1, 2, 3, 4])
y1 = np.array([3, 5, 8, 10])
y2 = np.array([3.5, 5.5, 8.5, 10.5])
y3 = np.array([4, 6, 9, 11])

plt.bar(x, y1, color = 'g', width = 0.1)
plt.bar(x+0.1, y2, color = '#0000ff', width = 0.1)
plt.bar(x+0.2, y3, color = 'hotpink', width = 0.1)
plt.xticks(x + 0.2 / 2, ('Summer', 'Winter', 'Autumn','Spring'))
plt.show()

plt.legend(['Existing Scheme1','Existing Scheme2','Proposed Scheme'])
plt.xlabel('Weather')
plt.ylabel('Sensor Values')

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```



1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)

...continued

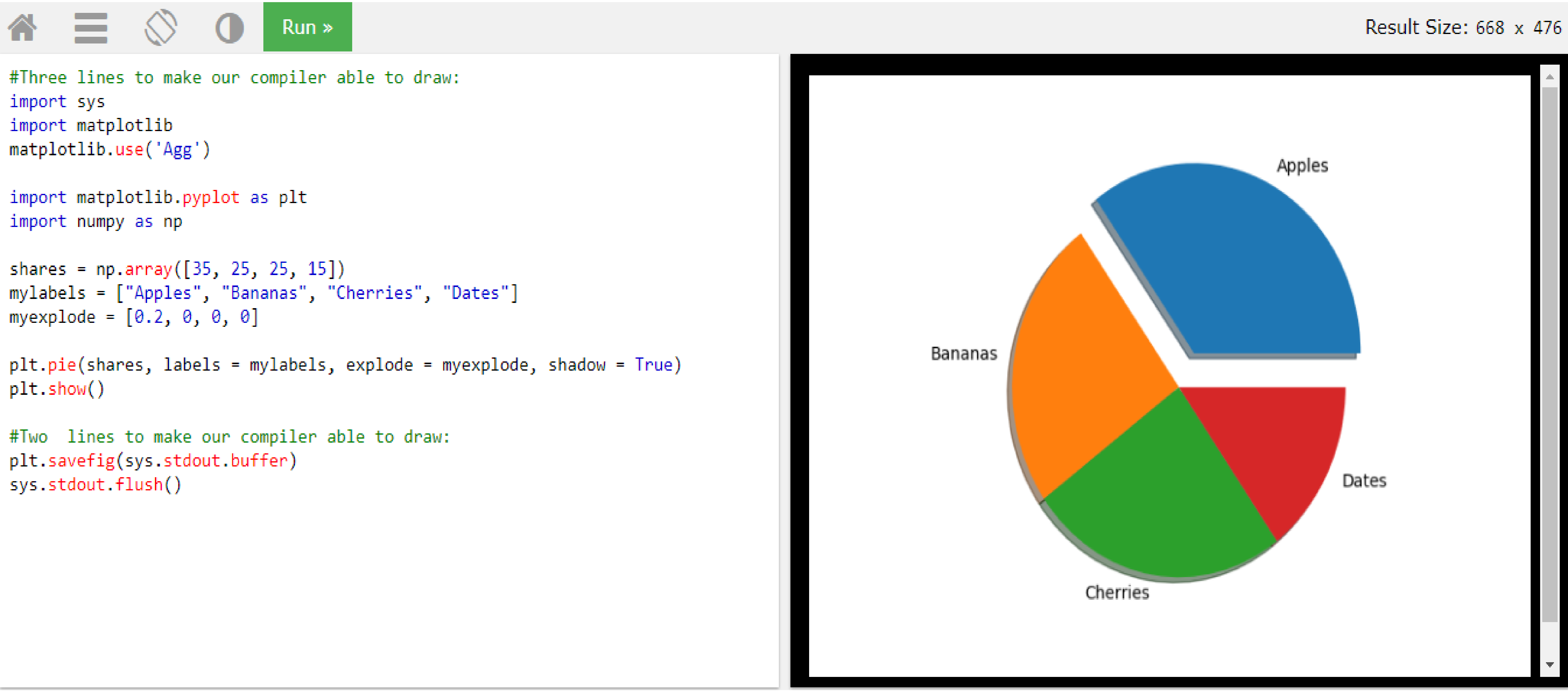
C. Pie Chart:

- By default the plotting of the first wedge starts from the **x-axis** and move **counterclockwise.**
- **pie( )** function is used to draw the pie charts.
- **pie**(populationShare, labelsWedge, colors, startAngle, explode, shadow)
- **legend**(title = "Four Fruits:", loc='lower right')

Location String	Location Code
'best'	0
'upper right'	1
'upper left'	2
'lower left'	3
'lower right'	4
'right'	5
'center left'	6
'center right'	7
'lower center'	8
'upper center'	9
'center'	10

1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)

...continued



```
#Three lines to make our compiler able to draw:
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

shares = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
myexplode = [0.2, 0, 0, 0]

plt.pie(shares, labels = mylabels, explode = myexplode, shadow = True)
plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

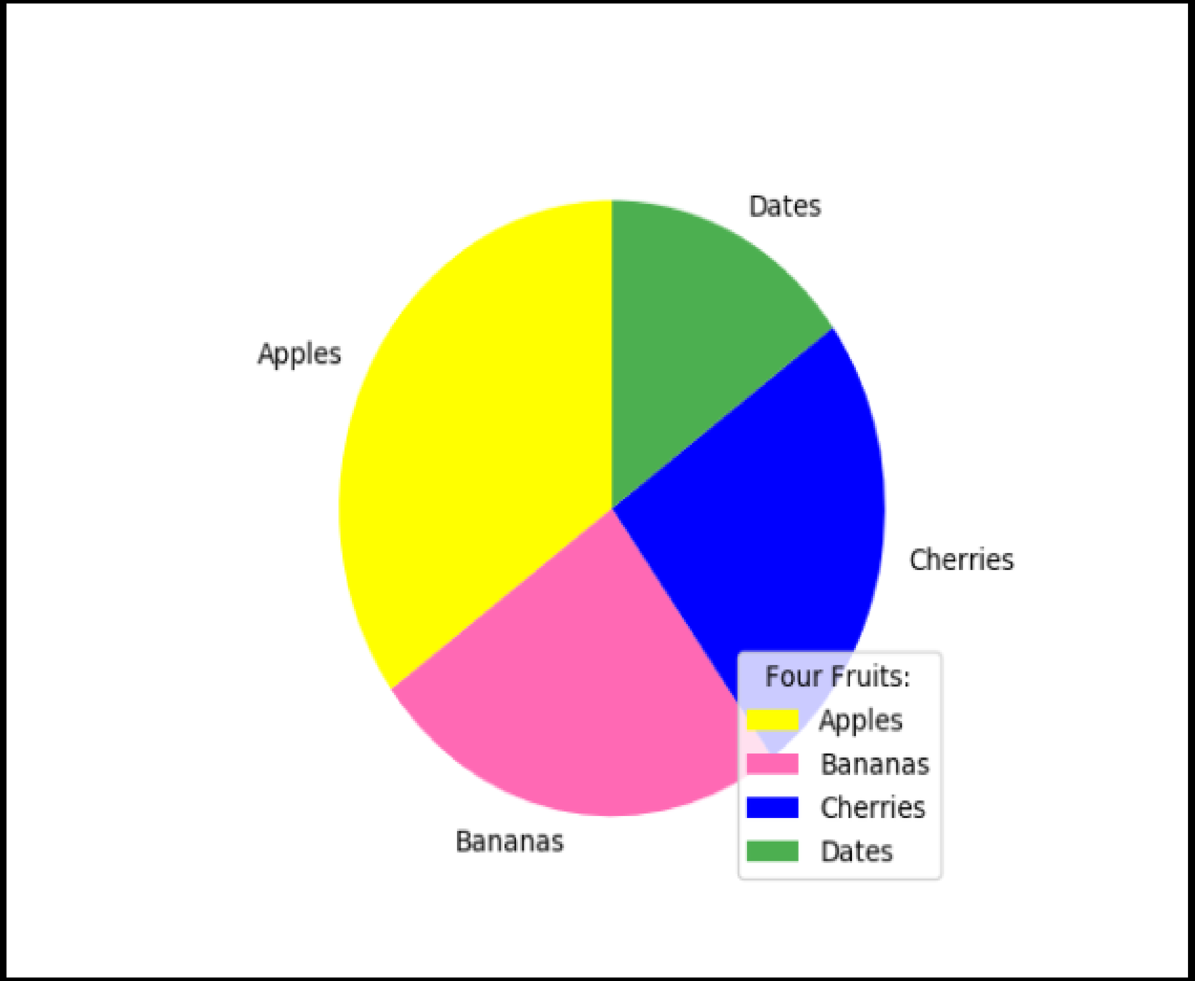
1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)

...continued

Run »

Result Size: 668 x 476

```
#Three lines to make our compiler able to draw:
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

shares = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
mycolors = ['yellow', "hotpink", "b", "#4CAF50"]

plt.pie(shares, labels = mylabels, colors = mycolors, startangle = 90)
plt.legend(title = "Four Fruits:", loc='lower right')          #loc=4
plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```



1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)

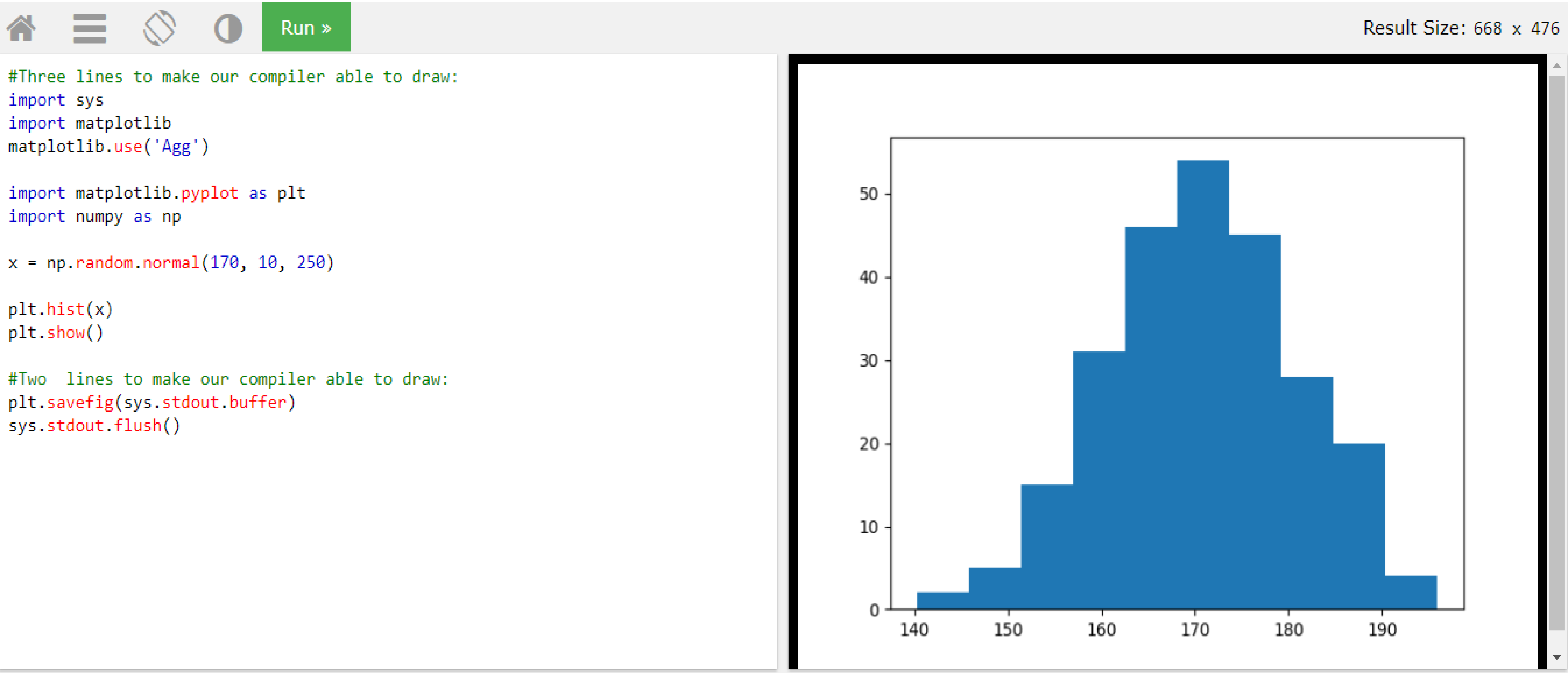
…continued

D. Histogram:

- A histogram is a graph showing **frequency distributions**.
- It is a graph showing the **number of observations within each given interval**.
- **hist()** function to create histograms.
- Create a histogram to represent following:
    - ❖ 2 people from 140 to 145cm
    - ❖ 5 people from 145 to 150cm
    - ❖ 15 people from 151 to 156cm
    - ❖ 31 people from 157 to 162cm
    - ❖ 46 people from 163 to 168cm
    - ❖ 53 people from 168 to 173cm
    - ❖ 45 people from 173 to 178cm
    - ❖ 28 people from 179 to 184cm
    - ❖ 21 people from 185 to 190cm
    - ❖ 4 people from 190 to 195cm
- For this, function **numpy.random.normal(170, 10, 250)** can be used which shows that **NumPy** uses **Normal Distribution** to **randomly** generate an array with **250** values, where the values will concentrate around **170**, and the standard deviation is **10**.

1. Creating various types of graphs corresponding to any data (to show different kinds of results and analysis)

...continued



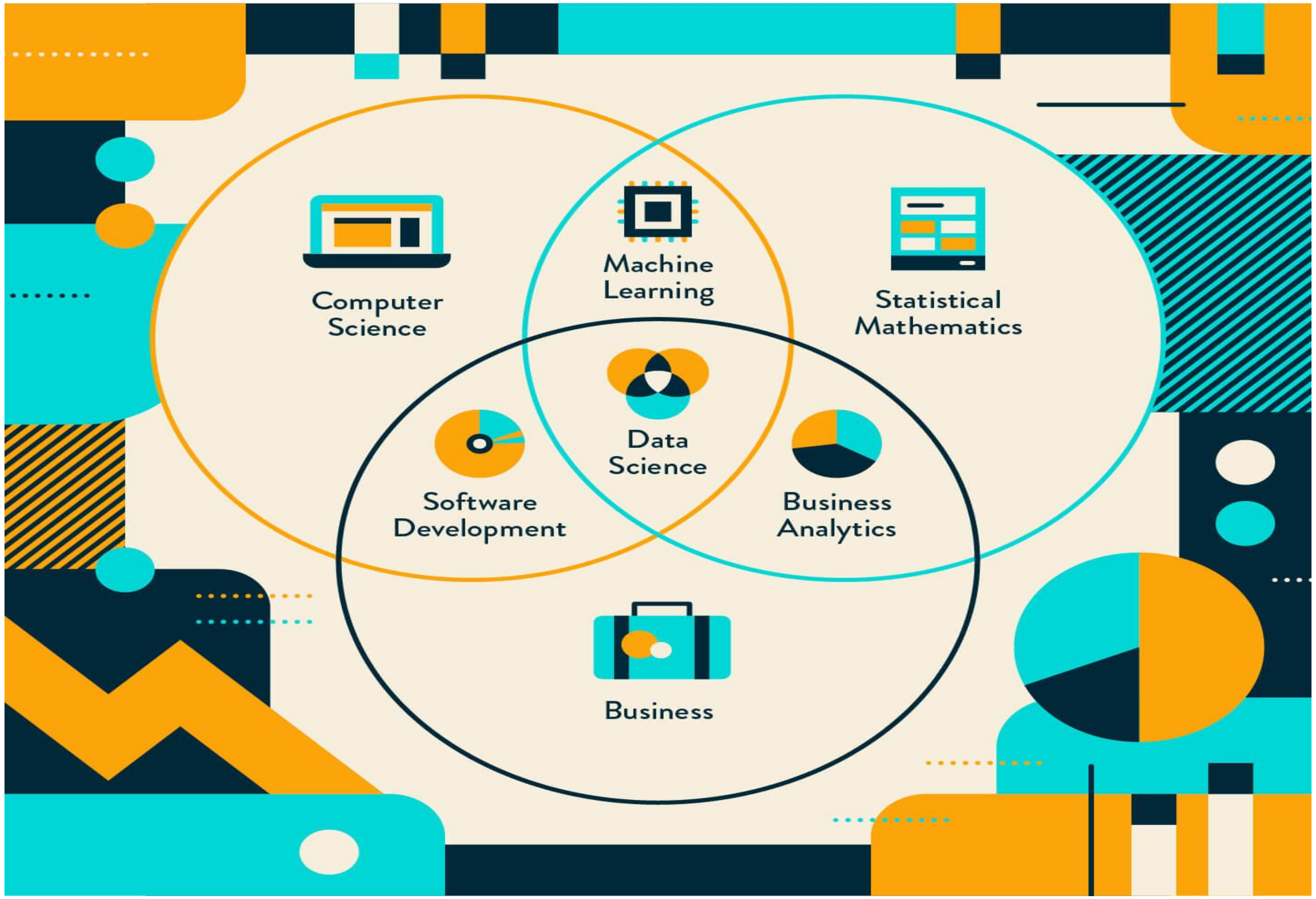
```
#Three lines to make our compiler able to draw:
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

x = np.random.normal(170, 10, 250)

plt.hist(x)
plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

B. Data Analysis

1. Understanding problems of data science and machine learning

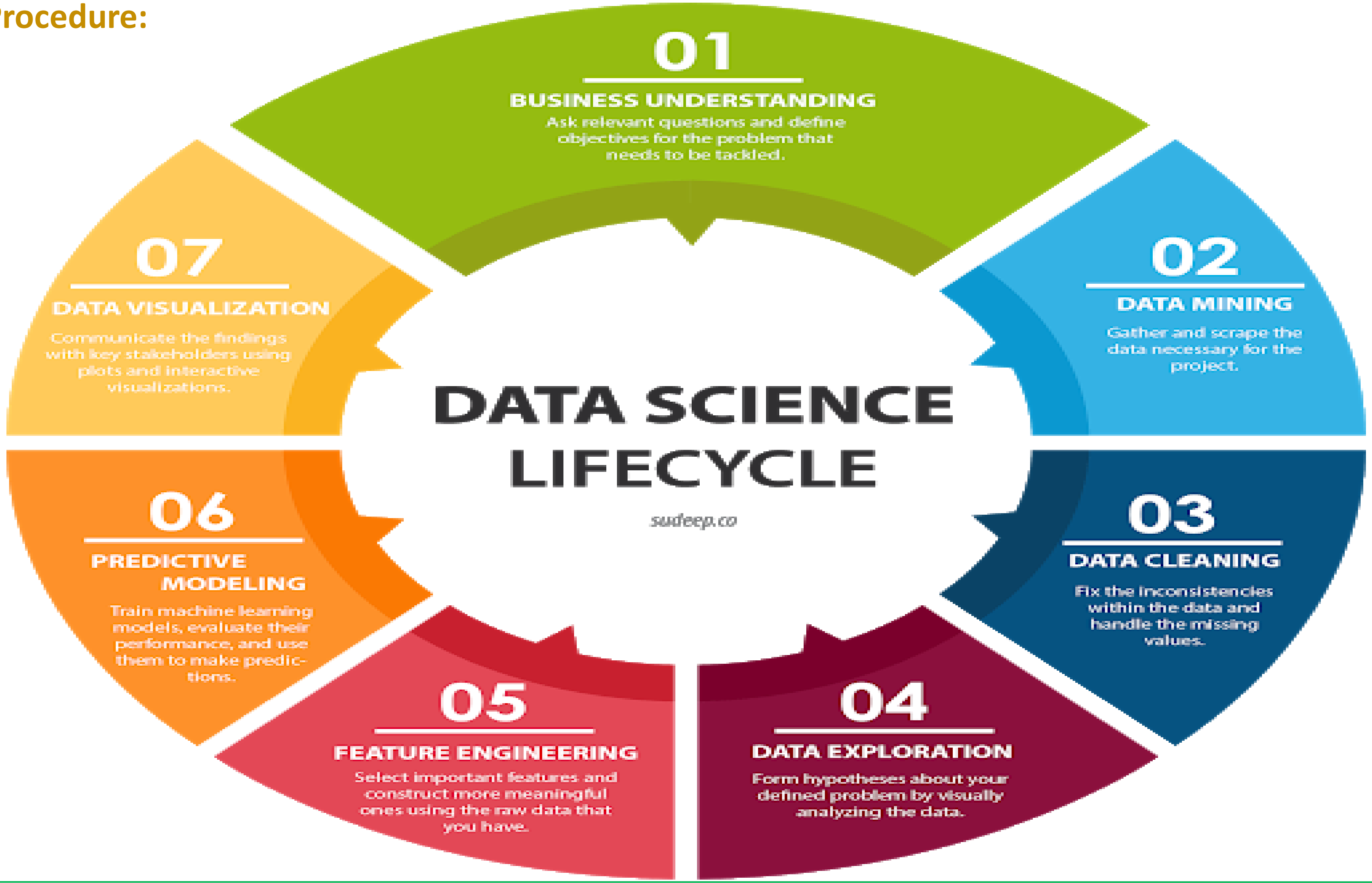
1. Understanding problems of data science and machine learning

A. Introduction:



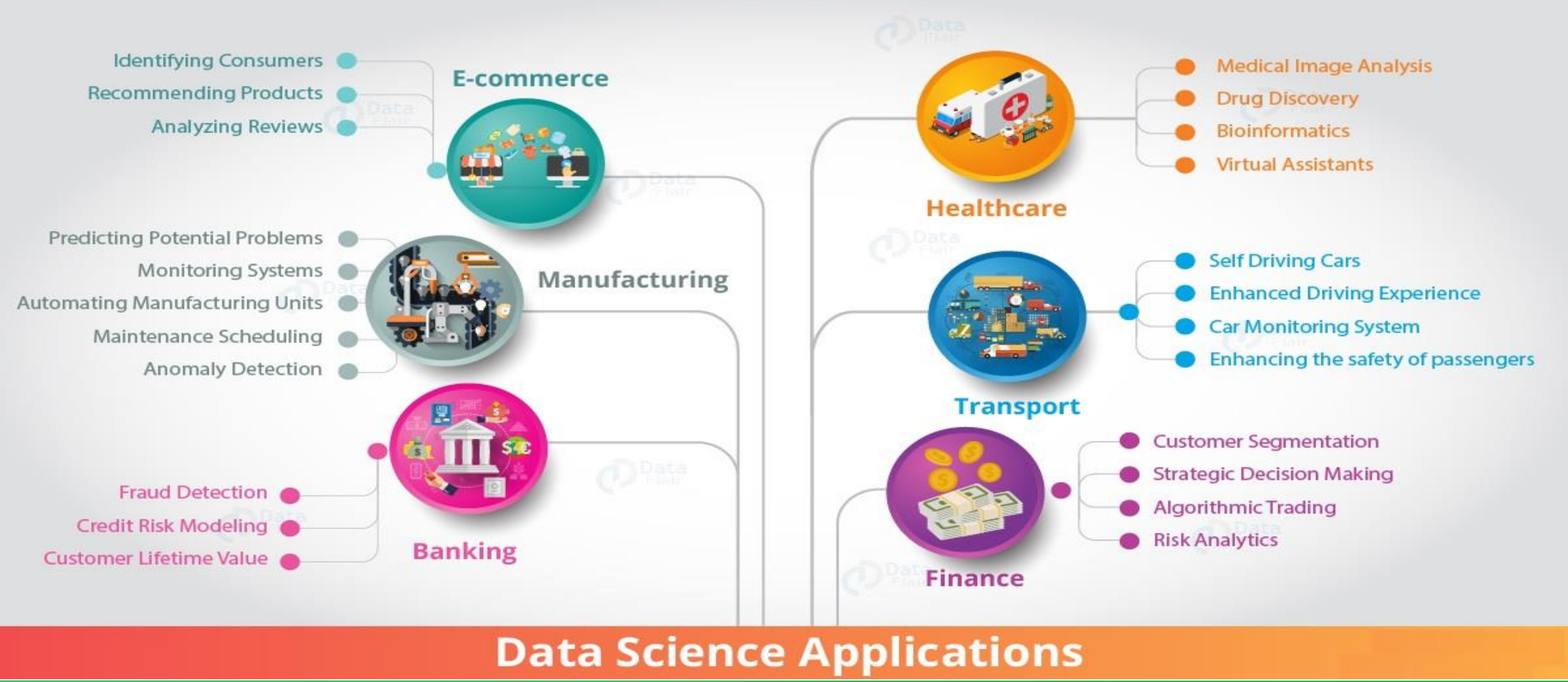
1. Understanding problems of data science and machine learning ...continued

B. Procedure:


DATA SCIENCE LIFECYCLE
sudeep.co
01
BUSINESS UNDERSTANDING
Ask relevant questions and define objectives for the problem that needs to be tackled.
02
DATA MINING
Gather and scrape the data necessary for the project.
03
DATA CLEANING
Fix the inconsistencies within the data and handle the missing values.
04
DATA EXPLORATION
Form hypotheses about your defined problem by visually analyzing the data.
05
FEATURE ENGINEERING
Select important features and construct more meaningful ones using the raw data that you have.
06
PREDICTIVE MODELING
Train machine learning models, evaluate their performance, and use them to make predictions.
07
DATA VISUALIZATION
Communicate the findings with key stakeholders using plots and interactive visualizations.

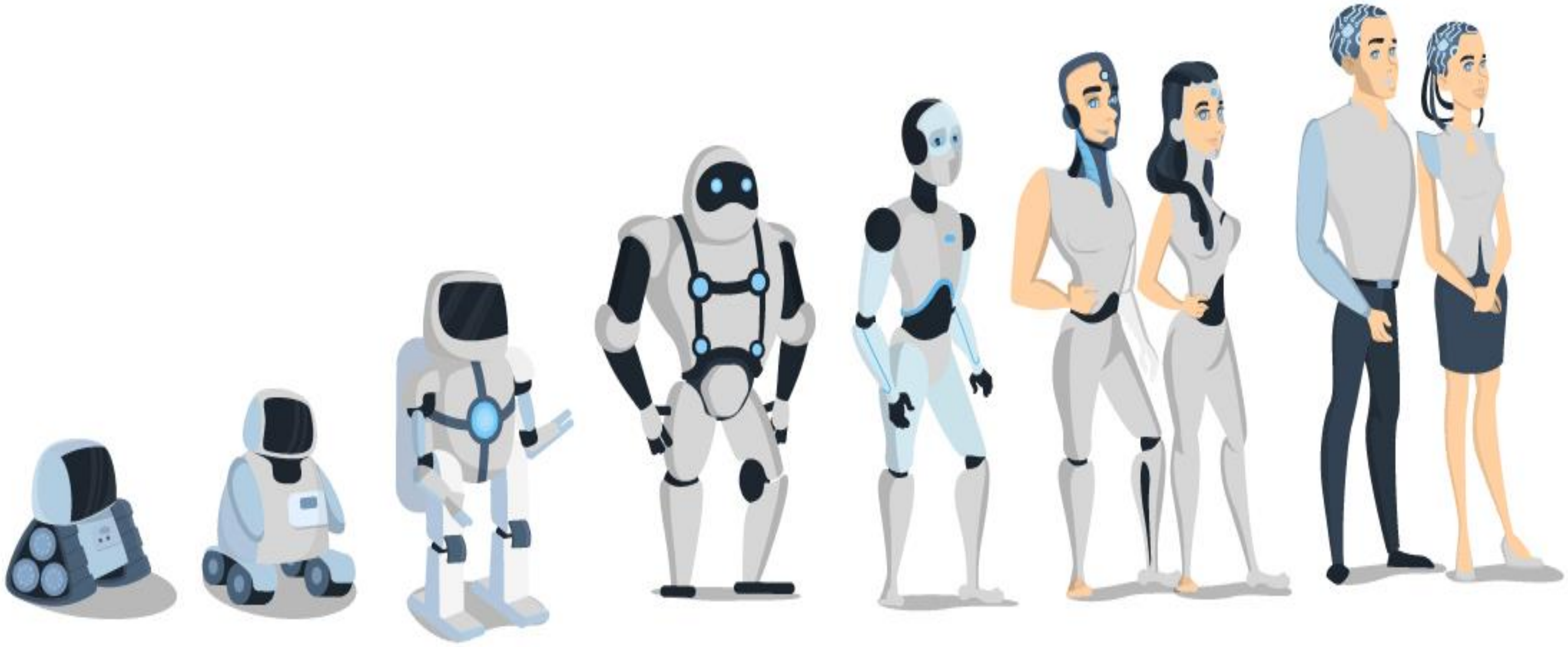
1. Understanding problems of data science and machine learning …continued

C. Applications:



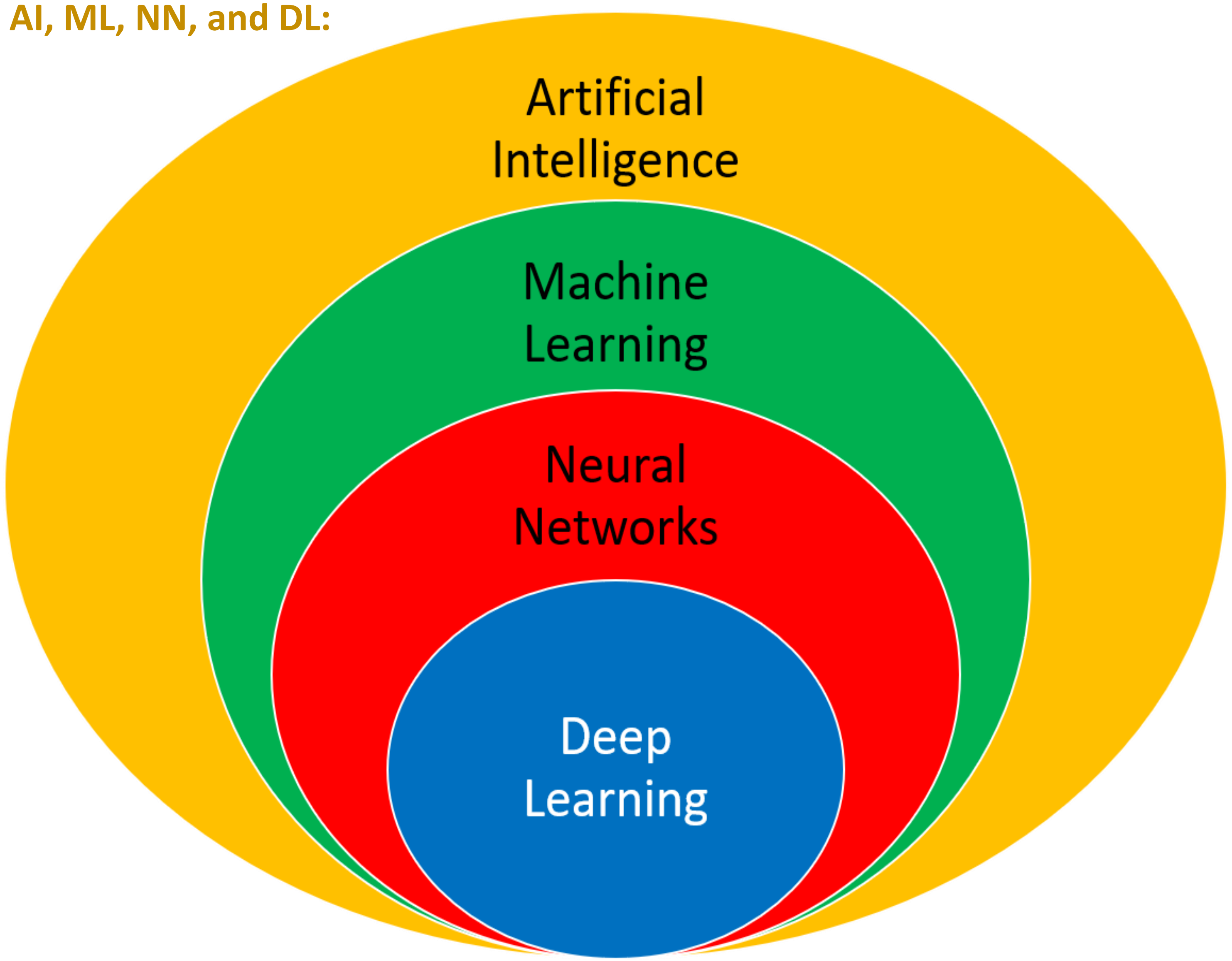
1. Understanding problems of data science and machine learning …continued

A. Introduction:



1. Understanding problems of data science and machine learning ...continued

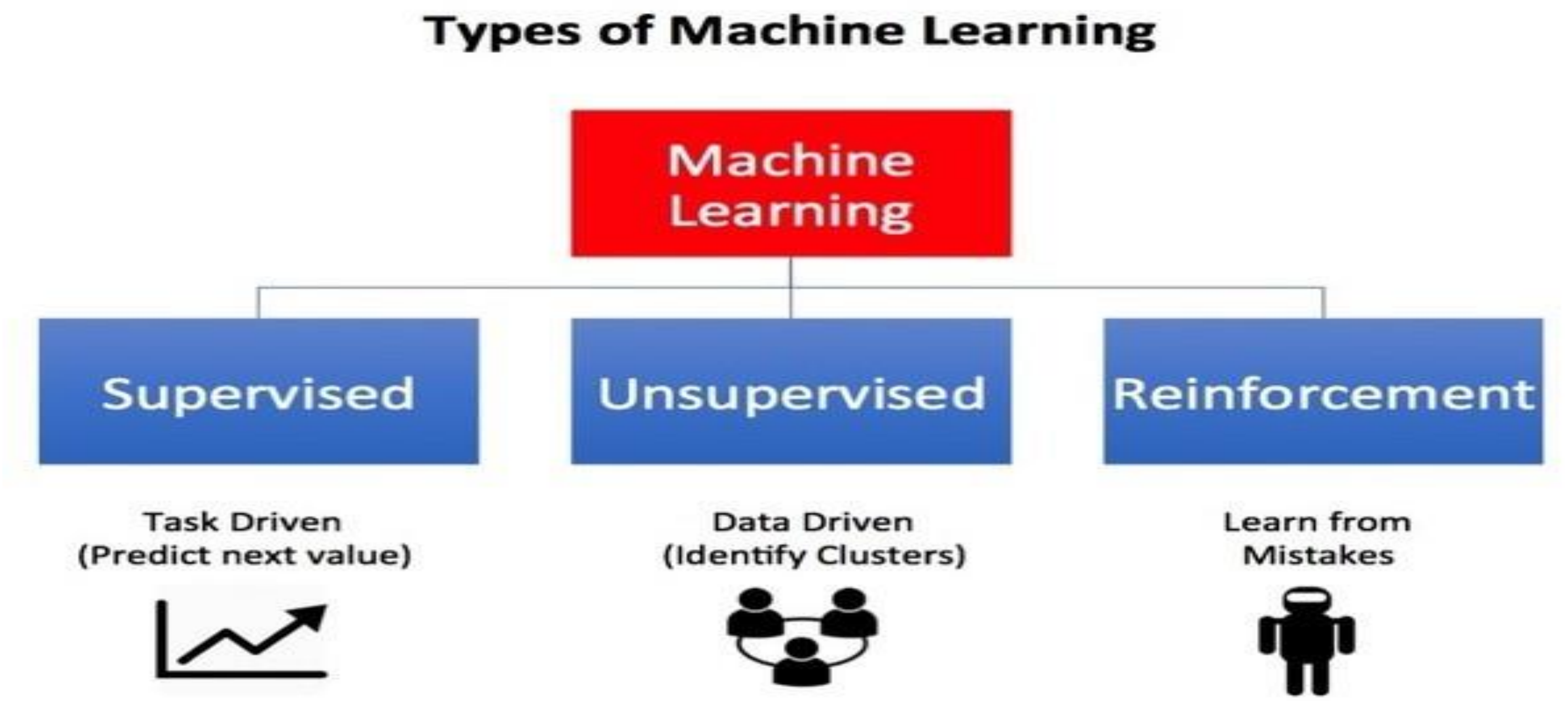
B. Relation among AI, ML, NN, and DL:



1. Understanding problems of data science and machine learning …continued

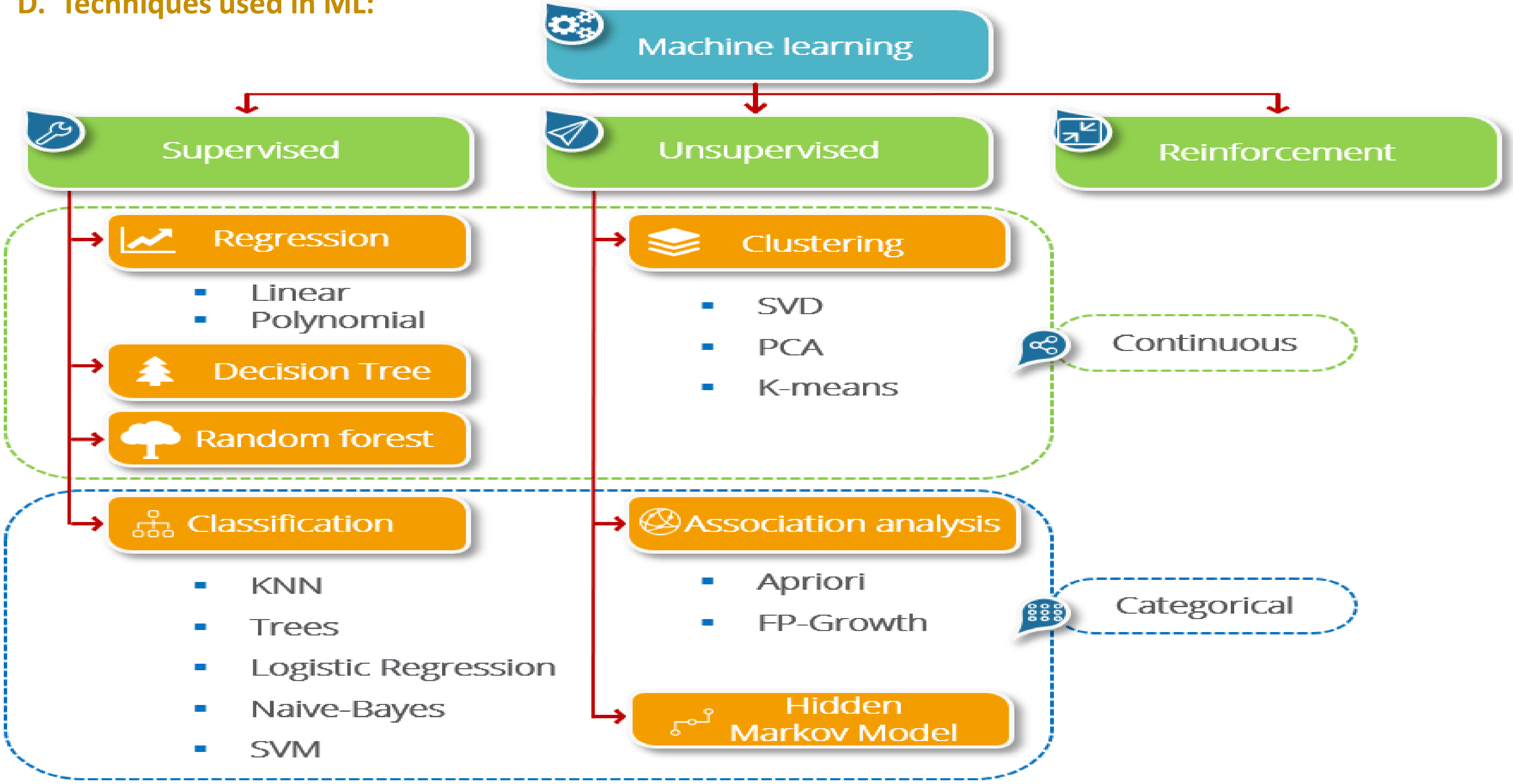
C. Types of ML:

•**Supervised Learning** – Train Me!
•**Unsupervised Learning** – I am self sufficient in learning
•**Reinforcement Learning** – My life My rules! (Hit & Trial)


Types of Machine Learning
Machine Learning
Supervised
Unsupervised
Reinforcement
Task Driven (Predict next value)
Data Driven (Identify Clusters)
Learn from Mistakes

1. Understanding problems of data science and machine learning …continued

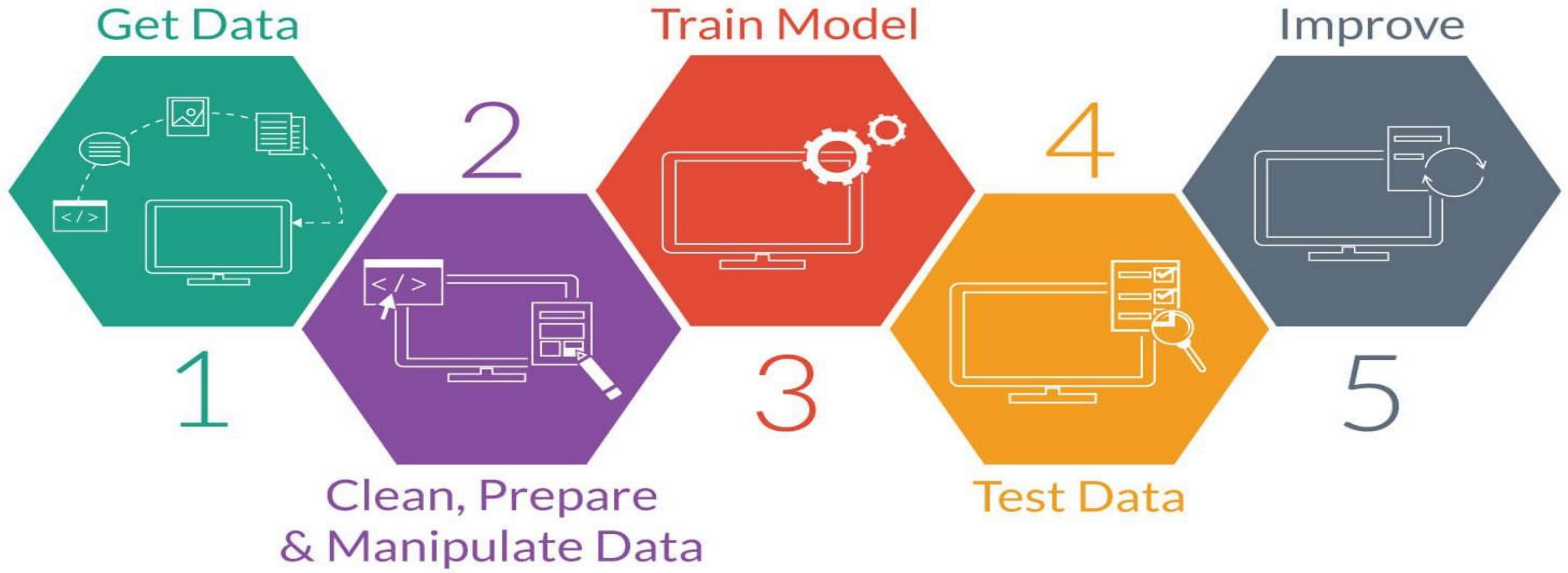
D. Techniques used in ML:



1. Understanding problems of data science and machine learning ...continued

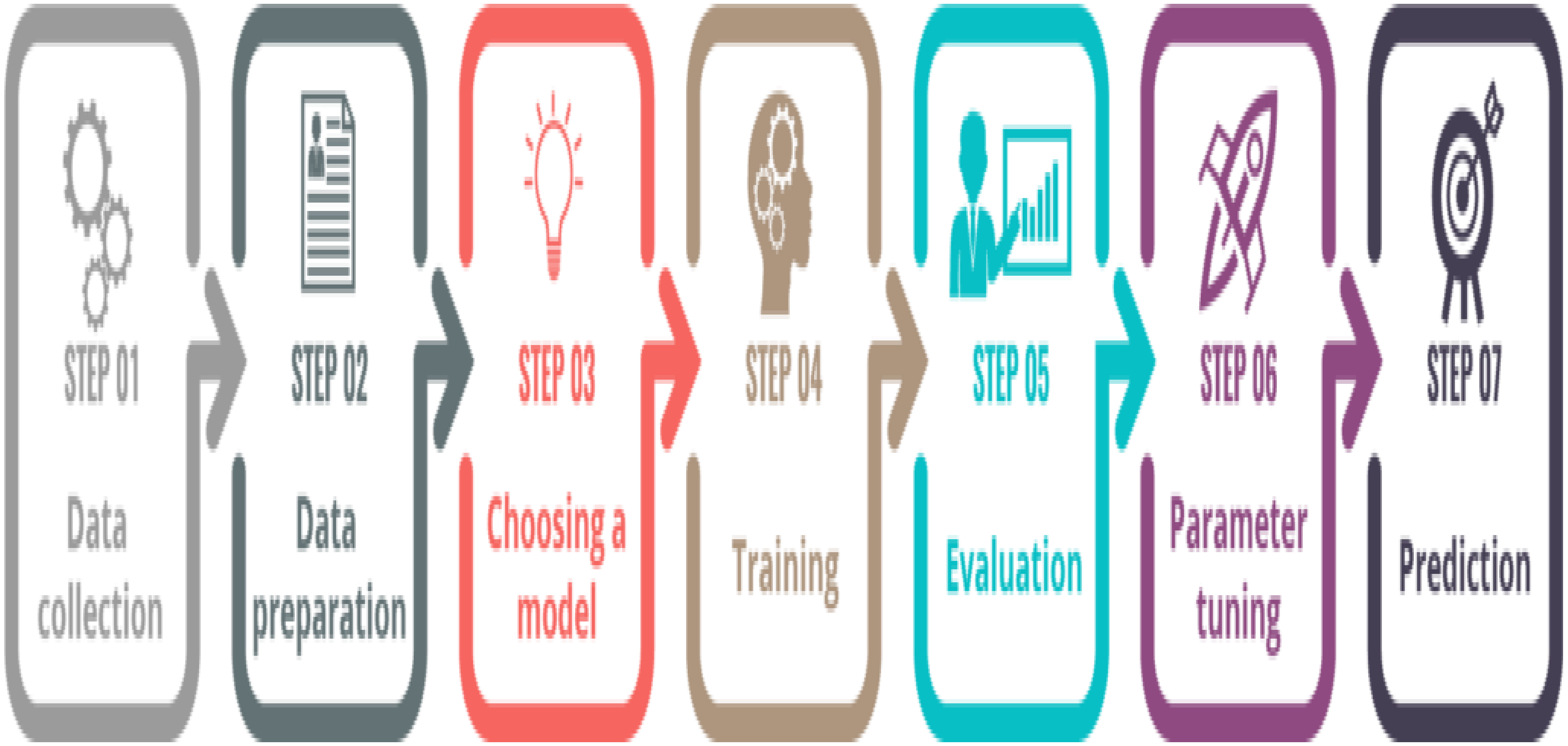
E. Procedure (View 1)

Usually 80% data for training, and 20% data for testing



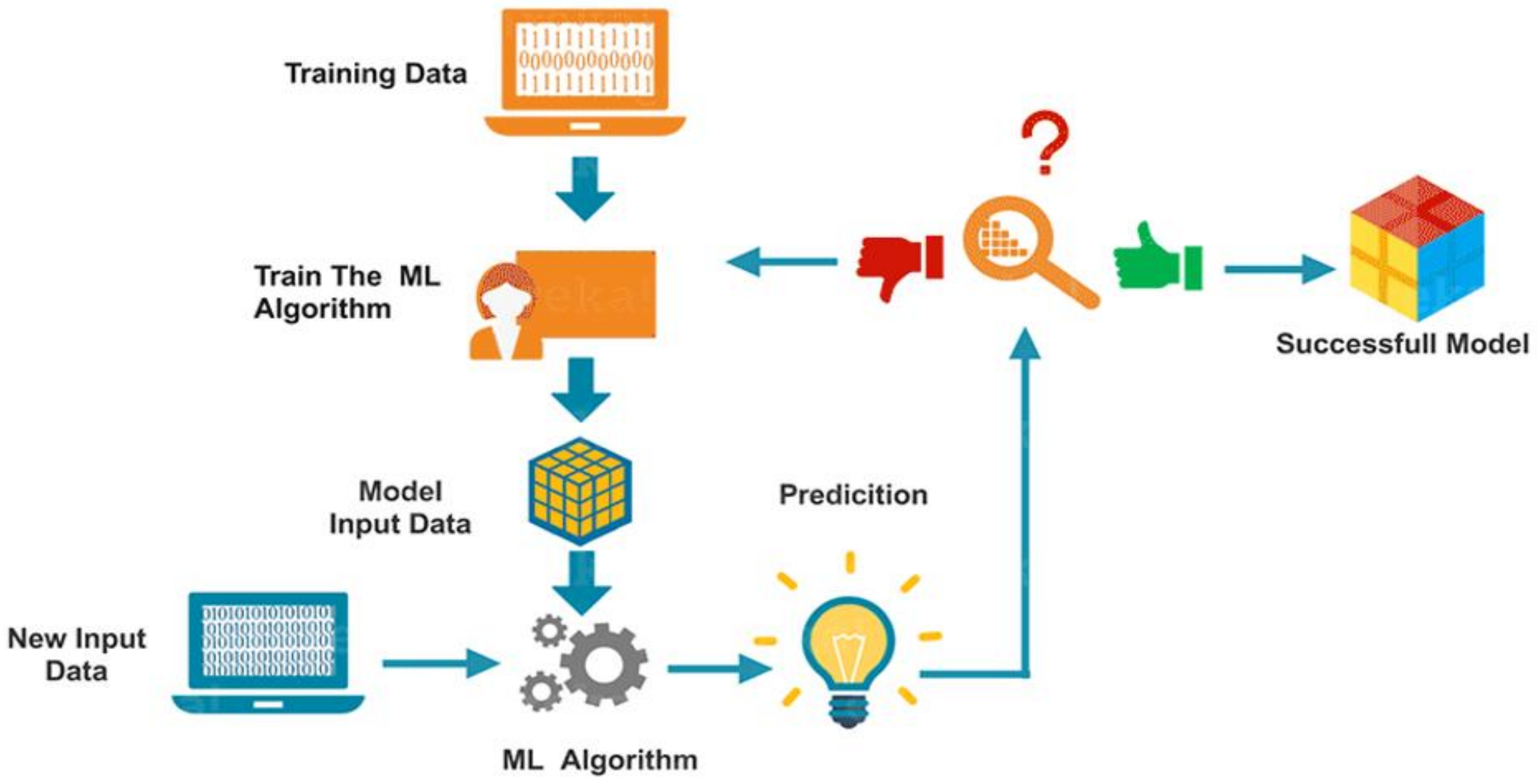
1. Understanding problems of data science and machine learning …continued

E. Procedure (View 2)



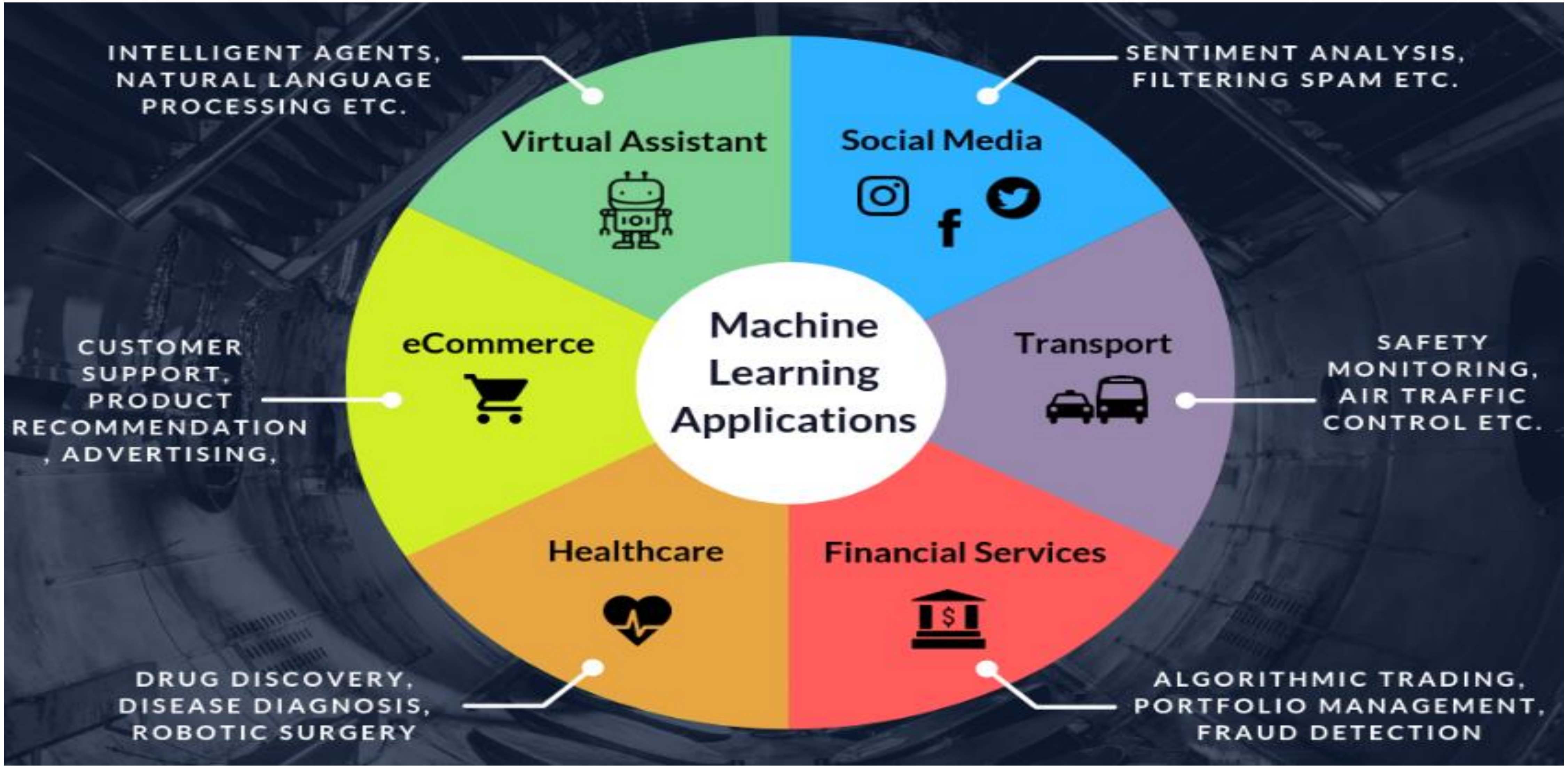
1. Understanding problems of data science and machine learning …continued

E. Procedure (View 3)



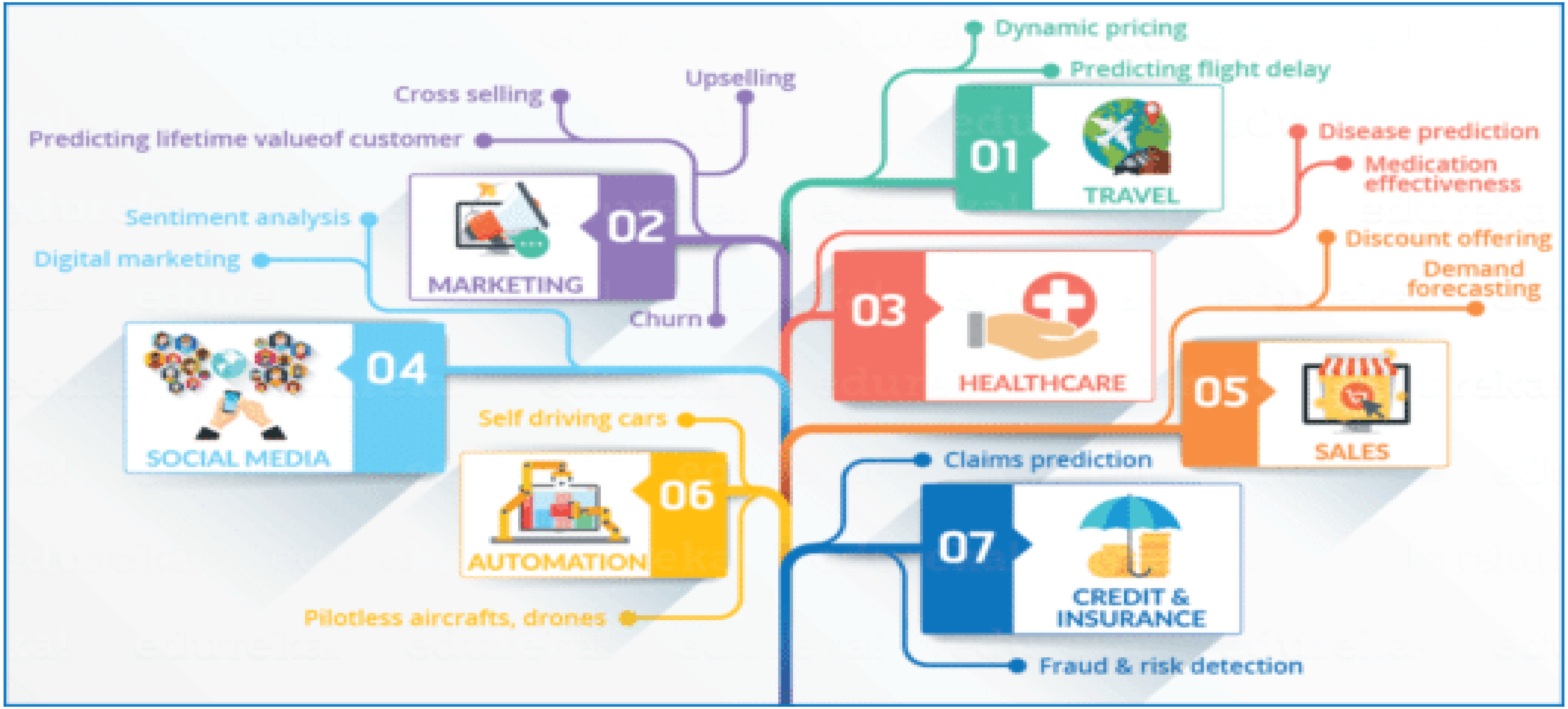
1. Understanding problems of data science and machine learning …continued

F. Applications (View 1):


Machine Learning Applications
Virtual Assistant
INTELLIGENT AGENTS, NATURAL LANGUAGE PROCESSING ETC.
Social Media
SENTIMENT ANALYSIS, FILTERING SPAM ETC.
eCommerce
CUSTOMER SUPPORT, PRODUCT RECOMMENDATION, ADVERTISING,
Transport
SAFETY MONITORING, AIR TRAFFIC CONTROL ETC.
Healthcare
DRUG DISCOVERY, DISEASE DIAGNOSIS, ROBOTIC SURGERY
Financial Services
ALGORITHMIC TRADING, PORTFOLIO MANAGEMENT, FRAUD DETECTION

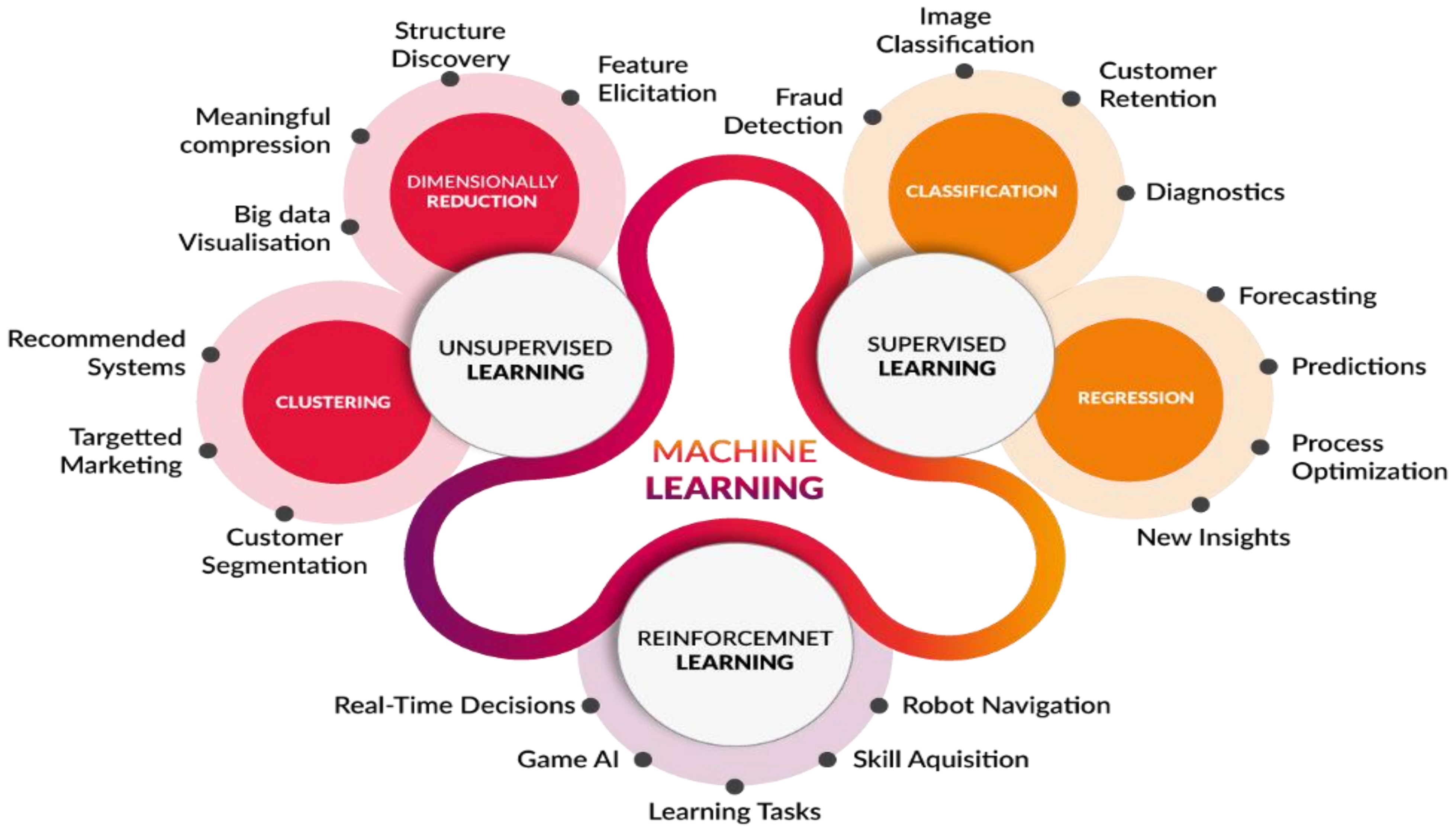
1. Understanding problems of data science and machine learning …continued

F. Applications (View 2):



1. Understanding problems of data science and machine learning ...continued

F. Applications (View 3):



2. Creating codes for data analysis problems in Python

2. Creating codes for data analysis problems in Python

First of all, import or load the dataset and then analyse it.

A. The basic process of loading data from a CSV file with Pandas

```
# Load the Pandas libraries with alias 'pd'
import pandas as pd

# Read data from file 'filename.csv' (in the same directory)
data = pd.read_csv("filename.csv")

# Preview the first 5 lines of the loaded data
data.head()
```

OR

```
import pandas
filename = ‘indians-diabetes.data.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
data = pandas.read_csv(filename, names=names)
print(data.shape)
```

2. Creating codes for data analysis problems in Python …continued

B. The basic process of loading data from a CSV file with NumPy

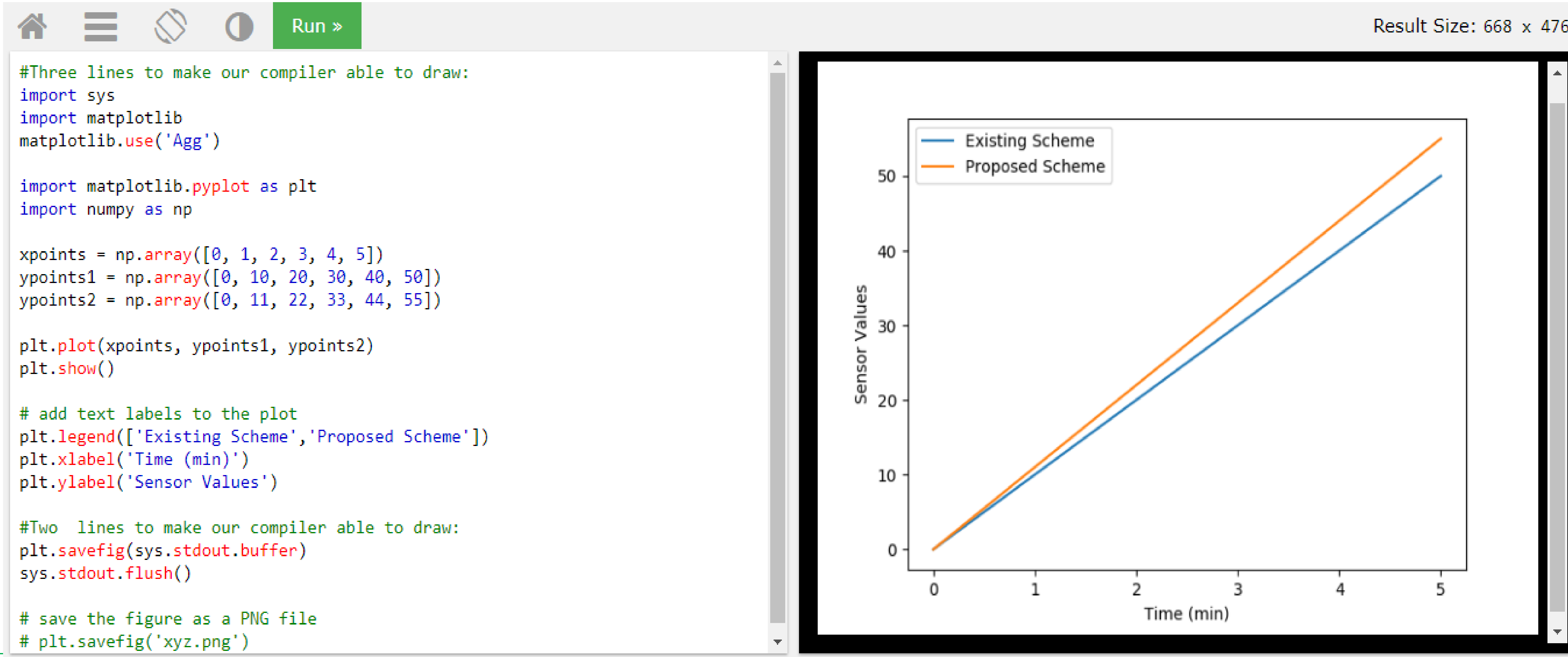
```
import numpy
filename = 'indians-diabetes.data.csv'
raw_data = open(filename, 'rt')
data = numpy.loadtxt(raw_data, delimiter=",")
print(data.shape)
```

C. The basic process of loading data from a CSV file with Python Standard Library

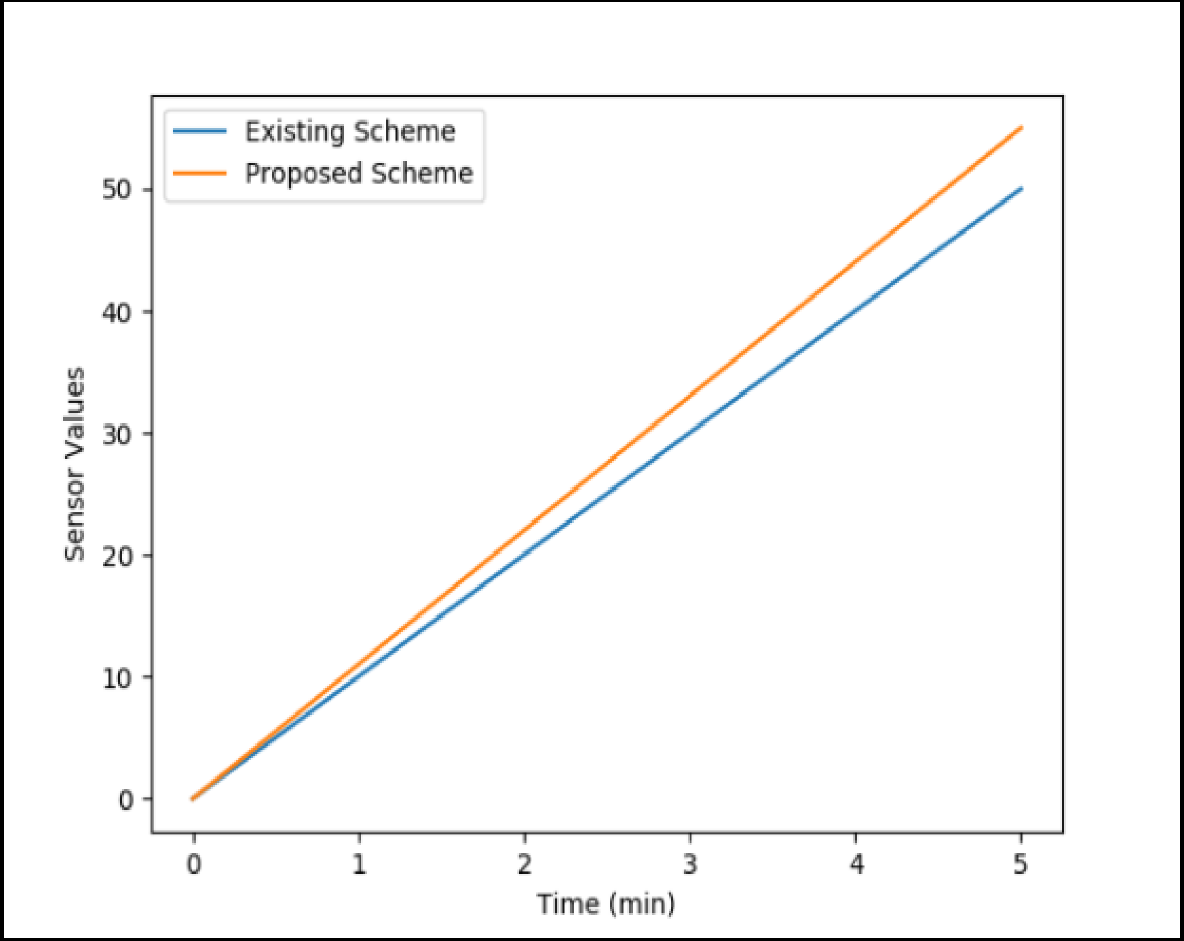
```
import csv
import numpy
filename = 'indians-diabetes.data.csv'
raw_data = open(filename, 'rt')
reader = csv.reader(raw_data, delimiter=‘,’  , quoting=csv.QUOTE_NONE)
x = list(reader)
data = numpy.array(x).astype('float')
print(data.shape)
```

2. Creating codes for data analysis problems in Python ...continued

D. Data Analysis

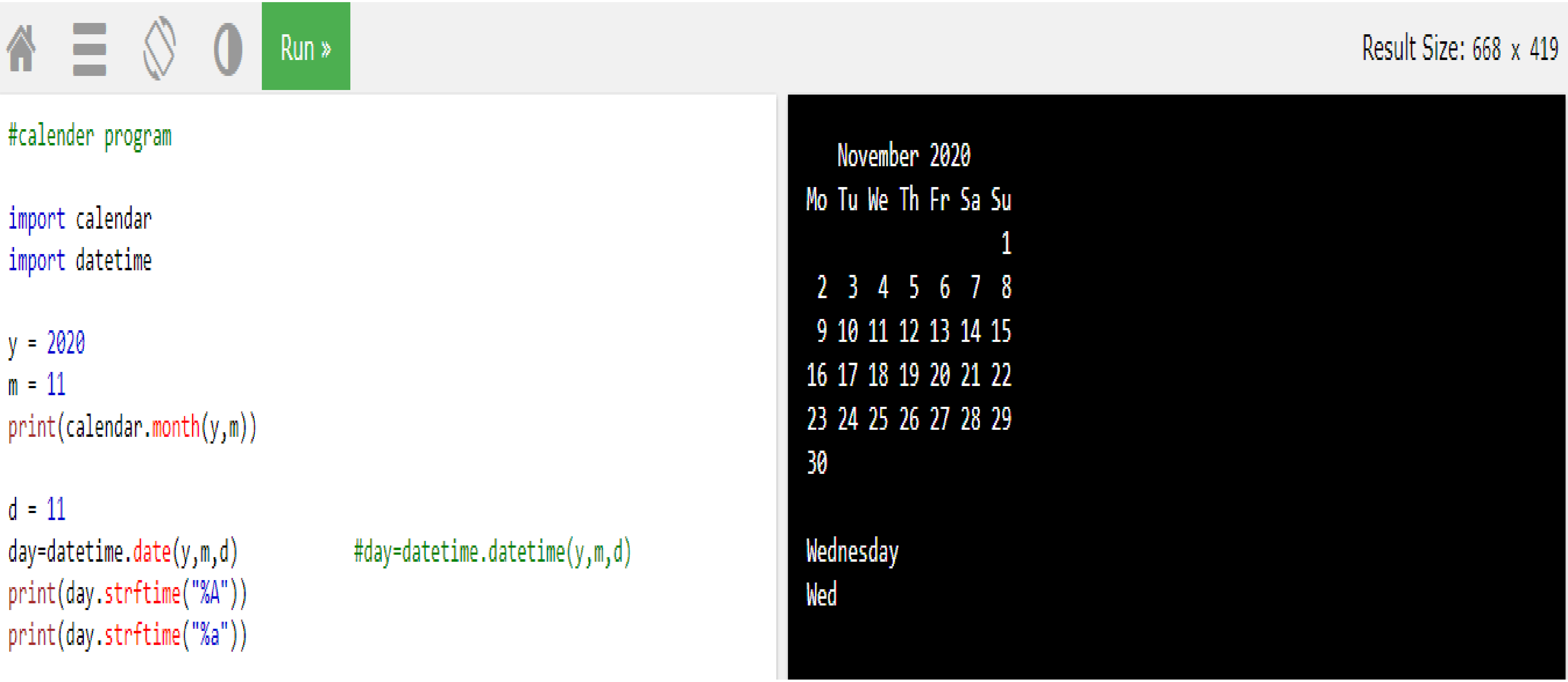


```
#Three lines to make our compiler able to draw:
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([0, 1, 2, 3, 4, 5])
ypoints1 = np.array([0, 10, 20, 30, 40, 50])
ypoints2 = np.array([0, 11, 22, 33, 44, 55])

plt.plot(xpoints, ypoints1, ypoints2)
plt.show()

# add text labels to the plot
plt.legend(['Existing Scheme','Proposed Scheme'])
plt.xlabel('Time (min)')
plt.ylabel('Sensor Values')

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()

# save the figure as a PNG file
# plt.savefig('xyz.png')
```



3. Other advance programs

3. Other advance programs: Calendar (I)

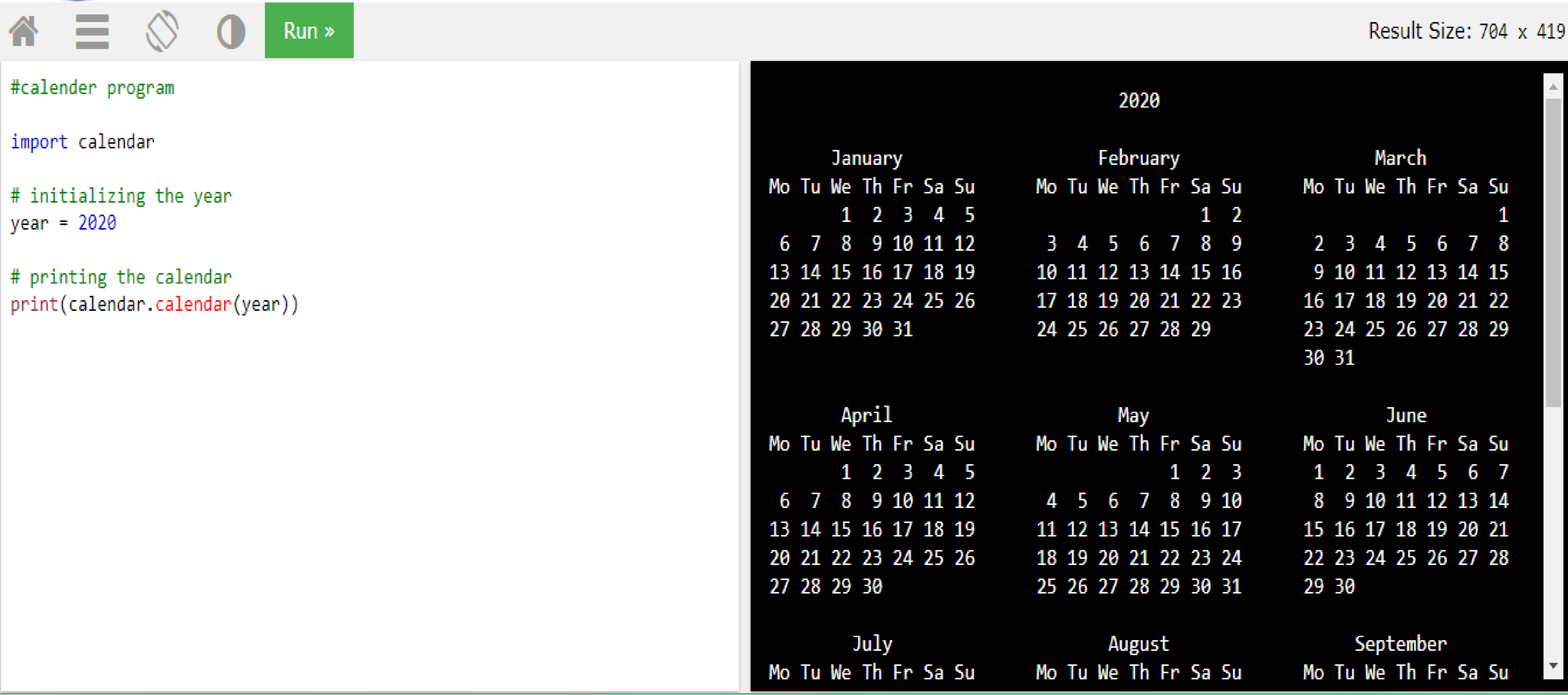


```
#calender program

import calendar
import datetime

y = 2020
m = 11
print(calendar.month(y,m))

d = 11
day=datetime.date(y,m,d)           #day=datetime.datetime(y,m,d)
print(day.strftime("%A"))
print(day.strftime("%a"))
```

```
   November 2020
Mo Tu We Th Fr Sa Su
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30

Wednesday
Wed
```

3. Other advance programs : Calendar (II)

…continued



```
#calender program

import calendar

# initializing the year
year = 2020

# printing the calendar
print(calendar.calendar(year))
```

```
                              2020

      January                   February                   March
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
       1  2  3  4  5                      1  2                         1
 6  7  8  9 10 11 12       3  4  5  6  7  8  9       2  3  4  5  6  7  8
13 14 15 16 17 18 19      10 11 12 13 14 15 16       9 10 11 12 13 14 15
20 21 22 23 24 25 26      17 18 19 20 21 22 23      16 17 18 19 20 21 22
27 28 29 30 31            24 25 26 27 28 29         23 24 25 26 27 28 29
                                                    30 31

       April                      May                       June
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
       1  2  3  4  5                   1  2  3       1  2  3  4  5  6  7
 6  7  8  9 10 11 12       4  5  6  7  8  9 10       8  9 10 11 12 13 14
13 14 15 16 17 18 19      11 12 13 14 15 16 17      15 16 17 18 19 20 21
20 21 22 23 24 25 26      18 19 20 21 22 23 24      22 23 24 25 26 27 28
27 28 29 30               25 26 27 28 29 30 31      29 30

        July                     August                  September
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
```

3. Other advance programs: Calculator

…continued

https://data-flair.s3.ap-south-1.amazonaws.com/python-projects/dataflair-python-calculator.zip

3. Other advance programs: Currency Converter

…continued

https://data-flair.s3.ap-south-1.amazonaws.com/python-projects/currency-converter-project.zip

3. Other advance programs: Music Player

…continued

https://project-gurukul.s3.ap-south-1.amazonaws.com/python-projects/music-player-python.zip

3. Other advance programs: Alarm Clock

…continued

https://data-flair.s3.ap-south-1.amazonaws.com/python-projects/DataFlair-Alarm-Clock.zip

Queries ?