Python Libraries for Data Analysis

1Tiji Tom
1Assistant Professor
1JPM Arts And Science College

Abstract **- Python is one of the most widely used languages in the world of programming languages. Python is a language that can be used to improve the quality of programming. It has added features to improve the application areas of this language. One of the main attractions of Python is its language library packages. Data Science and Machine Learning are the most popular technologies of the current scenario. So this situation is pushed everyone to learn the various libraries and packages in python to implement Data Science and Machine Learning. This article focuses on the Python libraries for Data Science and Machine Learning.**

keywords **- Python language, a python library, data science, machine learning, pip**

INTRODUCTION

Python is a very emerging programming language in current trends in computer science. It mainly focuses on beginners and each one can learn python very easily with very little programming knowledge. Python language was developed in the year December 1989 by Guido Van Rossum. There are so many open-source versions available for Python. The goal should be clear before learning Python. Python is an easy, vast language as well. It includes several libraries, modules, in-built functions, and data structures. If the goal is unclear then it will be a boring and monotonous journey of learning Python. Without any clear goal, you perhaps won't make it done.

So, first figure out the motivation behind learning, which can anything be such as knowing something new, develop projects using Python, switch to Python, etc. The main application areas of python include Data Analysis and Processing, Artificial Intelligence, Games, Hardware/Sensors/Robots, Desktop Applications. Python is the most popular programming language used to implement Machine Learning and Data Science. The most important reason for the popularity of Python in the field of AI and Machine Learning is the fact that Python provides 100's of inbuilt libraries that have in-built functions and methods to easily carry out data analysis, processing, modeling, and so on.

PYTHON LIBRARIES

Packages are generally a group of classes with similar functionalities. Python has a wide variety of packages. A huge library of Python contains several packages. Packages are already developed programs that are available to the programmers. To download and install packages we can use a special command called pip. The abbreviation of pip is Python Installation of Packages. We should go to command prompt and then use the command as shown below:

Pip install *packagename*

*Th*is pip command is already included in Python software by default. After executing this pip command, it searches the latest version of the given package on the internet, downloads it, and then installs that package in our system.

DATA SCIENCE

Data science is the process of collecting and modifying useful information from data to solve real-world problems, mainly in the field of Artificial intelligence.

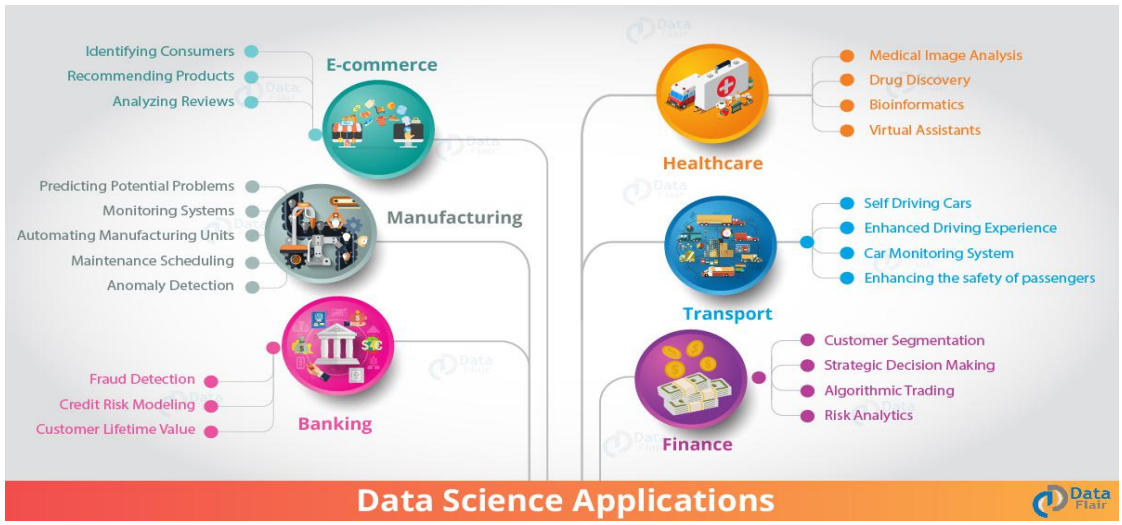

Figure 1: Data Science Applications

MACHINE LEARNING

Machine learning is the concept that a computer program can learn and form new data without human interaction. Machine learning is a field of artificial intelligence (AI) that keeps a computer's built-in algorithms current regardless of changes in the worldwide economy. The various data applications of machine learning are formed through a complex algorithm built into the machine or computer. This programming code creates a model that identifies the data and builds predictions around the data it identifies.

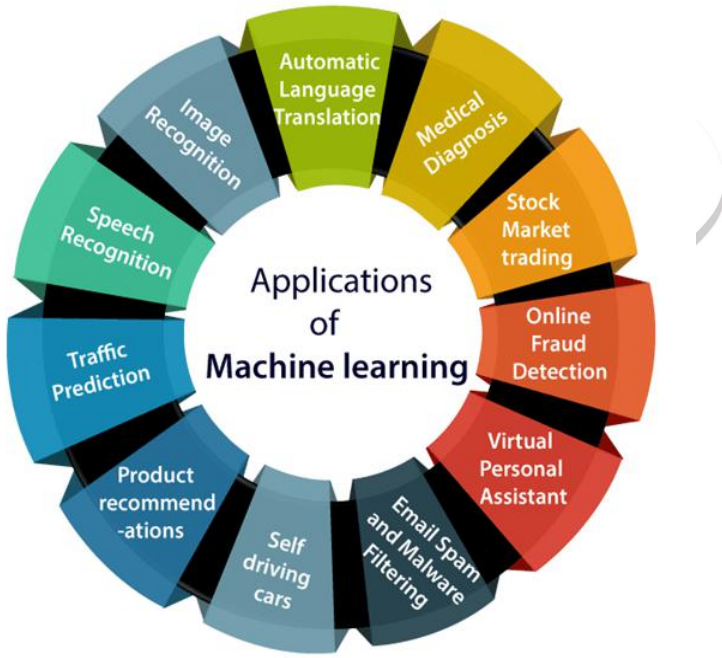

Figure 2:Applications of Machine learning

PACKAGE COLLECTIONS IN PYTHON FOR DATA SCIENCE AND MACHINE LEARNING

1.1. **Numpy Package**

By default Python support only single dimensional arrays. Numpy package is mainly used to create multidimensional arrays in python. Numpy is the abbreviation of the name Numerical Python Packages. It can be used for processing images, sound waves, and other binary operations. If you have to concentrate on data science or ML field, you must have a deep knowledge of NumPy to process your real-world data sets. The command used to install Numpy is,

pip install NumPy

NumPy arrays are very easy to create given the complex problems they solve. To create a very basic ndarray, you use the np.array() method. All you have to pass are the values of the array as a list:

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

we can also create multidimensional arrays using NumPy with the following syntax

np.array([[1,2,3,4],[5,6,7,8]])

1.2. SciPy

This python library includes modules for linear algebra, integration, optimization, and statistics. SciPy works great for all kinds of scientific programming projects (science, mathematics, and engineering). It offers efficient numerical processing such as numerical optimization, integration. The extensive documentation makes working with this library easy.

SciPy has several sub packages for various scientific computations which are shown in the following table:

Table 1:SciPy Sub Packages

Name	Description
Cluster	Clustering algorithms
Constants	Physical and mathematical constants
Fftpack	Fast Fourier Transform routines
Integrate	Integration and ordinary differential equation solvers
Interpolate	Interpolation and smoothing splines
Io	Input and Output
Linalg	Linear algebra
Ndimage	N-dimensional image processing
Odr	Orthogonal distance regression
Optimize	Optimization and root-finding routines
Signal	Signal processing
Sparse	Sparse matrices and associated routines
Spatial	Spatial data structures and algorithms
Special	Special functions
Stats	Statistical distributions and functions

These above packages should be imported prior to using them in the application.
For example: from scipy import cluster

1.3. **Pandas**

Pandas stand for 'Python Data Analysis Library', one of the most important Python tools used by Data Scientists today**.** The main application area of the Pandas package in Python is data analysis and data science. It is mainly used to create applications for stock prediction, advertisement, Big data analysis, etc. Pandas allow converting different types of huge data into DataFrame objects, handling missing data, and adding/deleting columns from DataFrame, imputing missing files, and plotting data with histogram or plot box.

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.

```
import pandas as pd
data = {
   'apples': [3, 2, 0, 1],
   'oranges': [0, 3, 7, 2]
}
```

And then pass it to the pandas DataFrame constructor:

```
purchases = pd.DataFrame(data)
purchases
```

1.4. Keras

One of the most powerful and easy-to-use Python libraries for developing and evaluating data science models in Keras; It covers the efficient numerical computation libraries Theano and TensorFlow. The advantage of this is mainly that you can understand and work in an easy and fun way. Keras is used by many scientific organizations around the world like CERN, NASA, NIH. Keras has the low-level flexibility to implement arbitrary research ideas while offering optional high-level convenience features to speed up experimentation cycles.

1.4.1. Keras Model class

tf. keras.Model()

Model groups layer into an object with training and inference features.

Arguments

- **inputs**: The input(s) of the model object
- **outputs**: The output(s) of the model after processing the input
- **name**: String, the name of the model object.

There are two ways to instantiate a Model:

The first method is shown below in the following table:

Table 2:Keras Model Class

```
import tensorflow as tf
inputs = tf.keras.Input(shape=(5,))
x = tf.keras.layers.Dense(7, activation=tf.nn.relu)(inputs)
outputs = tf.keras.layers.Dense(8, activation=tf.nn.softmax)(x)
model = tf.keras.Model(inputs=inputs, outputs=outputs)
```

By sub classing the Model class: in that case define the layers in __init__ and implement the model's forward pass in call.

```
import tensorflow as tf
class TestModel(tf.keras.Model):
  def __init__(self):
    super(TestModel, self).__init__()
    self.dense1 = tf.keras.layers.Dense(4, activation=tf.nn.relu)
    self.dense2 = tf.keras.layers.Dense(5, activation=tf.nn.softmax)
  def call(self, inputs):
    x = self.dense1(inputs)
    return self.dense2(x)
model = TestModel()
```

summary method

```
Model.summary(line_length=None, positions=None, print_fn=None)
```

Prints a string summary of the network.

Arguments

- **line_length**: Total length of printed lines of summary
- **positions**: Relative or absolute positions of log elements in each line. If not provided, defaults to [.33, .55, .67, 1.].
- **print_fn**: Print function to use. Defaults to print. It will be called on each line of the summary. You can set it to a custom function to capture the string summary.

1.5. **PyTorch**

PyTorch is considered one of the largest machine learning libraries for data scientists and researchers. It helps in dynamic computational graphs design, fast tensor computations accelerated through GPUs., and various other complex tasks. In neural network algorithms, PyTorch APIs play an effective role. The hybrid front-end PyTorch platform is very easy to use in graph mode for optimizations. For achieving accurate results in asynchronous collective operations and establishing a peer to peer communication it provides native supports to the users. Nowadays PyTorch is getting more popular among data scientists due to trending data-centric demands.

1.5.1. PyTorch Tensors

Tensors are the basic building blocks in PyTorch. In this part, I will list down some of the most used operations we can use while working with Tensors. This is by no means an exhaustive list of operations you can do with Tensors, but it is helpful to understand what tensors are before going towards the more exciting parts.

5.5.5.1. Create a Tensor

We can create a PyTorch tensor in multiple ways. This includes converting to tensor from a NumPy array. Below is an example to start with PyTorch tensor

```
t = torch.Tensor([[6,7,8],[3,4,5]])
print(f"Created Tensor Using torch.Tensor:\n{t}")
# Using torch. Randn
t = torch.randn(3, 5)
print(f"Created Tensor Using torch.randn:\n{t}")

# using torch.[ones|zeros](*size)
t = torch.ones(3, 5)
print(f"Created Tensor Using torch.ones:\n{t}")
t = torch.zeros(3, 5)
print(f"Created Tensor Using torch.zeros:\n{t}")

# using torch.randint - a tensor of size 4,5 with entries between 0 and 10(excluded)
t = torch.randint(low = 0,high = 10,size = (4,5))
print(f"Created Tensor Using torch.randint:\n{t}")

# Using from_numpy to convert from Numpy Array to Tensor
```

```
a = np.array([[1,2,3],[3,4,5]])
t = torch.from_numpy(a)
print(f"Convert to Tensor From Numpy Array:\n{t}")

# Using .numpy() to convert from Tensor to Numpy array
t = t.numpy()
print(f"Convert to Numpy Array From Tensor:\n{t}")
```

Tensor Operations

There are a lot of operations you can do on these tensors. The full list of functions are demonstrated in the following program

```
P = torch.randn(3,4)
Q = torch.randn(4,2)
# Multiply Matrix P and Q
t = P.mm(Q)
print(f"Created Tensor t by Multiplying P and Q:\n{t}")
# Transpose Tensor t
t = t.t()
print(f"Transpose of Tensor t:\n{t}")
# Square each element of t
t = t**2
print(f"Square each element of Tensor t:\n{t}")
# return the size of a tensor
print(f"Size of Tensor t using .size():\n{t.size()}")
```

CONCLUSION

A Python library is a set of reusable code that we can include in your programs/ projects. The wide range of application areas of python libraries includes image processing, graphics drawing, data analysis, data science, etc. Compared to languages like C++ or C, Python libraries loosely describe a collection of core modules. The packages can be installed using a package manager like pip. In this current scenario data science, data analysis and machine learning are the most emerging technologies. It has abundant uses in the real-world. Use python for these technologies is the good option which ever had.because python is the easy and fast learning programming language and very easy to learn even if we are beginners in programming.

REFERENCE

[1] https://techvidvan.com/tutorials/python-numpy-tutorial
[2] https://www.analyticsvidhya.com/blog/2020/04/the-ultimate-numpy-tutorial-for-data-science-beginners/
[3] https://www.edureka.co/blog/scipy-tutorial/
[4] https://bigdata-madesimple.com/top-20-python-libraries-for-data-science/
[5] https://beginnersbook.com/2018/01/introduction-to-python-programming/
[6] R Nageswara Rao,Python Programming,2nd Edition,Wiley India
[7] https://www.datarobot.com/wiki/data-science/
[8] https://www.investopedia.com/terms/m/machine-learning.asp
[9] https://www.ijcaonline.org/archives/volume178/number49/butwall-2019-ijca-919404.pdf
[10] https://www.dataquest.io/blog/15-python-libraries-for-data-science/
[11] Lt Col Rahul Dutt Sharma, "Python Tools for Big Data Analytics", International Journal of Science and Research (IJSR), https://www.ijsr.net/search_index_results_paperid.php?id=SR20507222308, Volume 9 Issue 5, May 2020, 597 - 602
[12] https://www.datacamp.com/community/tutorials/deep-learning-python
[13] https://keras.io/
[14] https://keras.io/api/models/model/