## Opencv:

OpenCV stands for Open Source Computer Vision Library. It is an open-source computer vision and machine learning software library that provides various tools and algorithms for image and video processing, object detection, and recognition.

In simpler terms, OpenCV is a programming library that allows developers to create software that can interpret and analyze digital images and videos. With OpenCV, you can create applications that can detect and track objects, recognize faces, and perform many other computer vision tasks.

OpenCV is used in a wide range of applications, including robotics, surveillance, automotive safety, and augmented reality. It is written in C++ and has bindings for many other programming languages, including Python and Java, making it accessible to a wide range of developers.

Overall, OpenCV is a powerful tool for anyone who wants to work with digital images and videos, regardless of their level of experience in computer vision.

## Built in functions:

Here are some examples of the built-in functions of the OpenCV library:

1. **Image and video I/O functions**: imread(), imwrite(), VideoCapture(), VideoWriter()

2. **Image processing functions**: resize(), cvtColor(), GaussianBlur(), Canny()

3. **Feature detection and extraction functions**: cornerHarris(), goodFeaturesToTrack(), ORB(), SIFT(), SURF()

4. **Object detection and recognition functions**: CascadeClassifier(), HOGDescriptor(), DNN module for deep learning-based detection and recognition

5. **Camera calibration functions**: findChessboardCorners(), calibrateCamera(), undistort()

6. **Machine learning functions**: SVM(), KNearest(), decision trees, Random Forests, neural networks.

## Matplotlib:

Matplotlib is a data visualization library in Python that is used to create 2D plots and graphs. It is one of the most popular data visualization libraries available in Python and is widely used by data scientists and researchers.

In simpler terms, Matplotlib helps you create graphs and charts from data you have collected. It can be used to plot various types of data, such as time series, scatter plots, histograms, and more. You can customize the appearance of your graphs, such as changing the colors, fonts, and axes labels.

Matplotlib is an open-source library, which means that it is free to use and can be modified by anyone. It is compatible with many different operating systems and platforms, making it a versatile tool for data visualization. Overall, Matplotlib is a powerful tool for creating high-quality visualizations of data, helping you to better understand and communicate your findings.

## Built in functions:

here are some of the most commonly used built-in functions of Matplotlib:

1. **pyplot**: This is the primary module of Matplotlib that provides a simple interface for creating plots and charts. It includes functions for creating line plots, scatter plots, histograms, bar charts, and more.

2. **figure**: This function creates a new figure for plotting. You can customize the size, resolution, and background color of the figure using the optional arguments.

3. **plot**: This function is used to create line plots. It accepts two arrays of data, one for the x-axis and one for the y-axis.

4. **scatter**: This function is used to create scatter plots. It accepts two arrays of data, one for the x-axis and one for the y-axis.

5. **hist**: This function is used to create histograms. It accepts an array of data and bins to group the data into.

6. **xlabel and ylabel**: These functions are used to label the x-axis and y-axis of a plot.

7. **legend**: This function adds a legend to a plot, which describes the meaning of different lines or markers in the plot.

8. **title**: This function adds a title to a plot.

9. **subplots**: This function is used to create multiple plots within a single figure.

10. **savefig**: This function saves a plot to a file in various formats, such as PNG, PDF, and SVG.

## Pandas:

Pandas is a Python library used for data manipulation and analysis. It provides a powerful set of tools for working with structured data, such as spreadsheets or SQL tables.

In simpler terms, Pandas is like Excel or SQL, but in Python. You can use Pandas to read data from different sources, such as CSV files or SQL databases, and manipulate the data in various ways. You can filter data, sort data, group data, aggregate data, and more.

Pandas also provides a way to work with time-series data, which is data that is organized by time. This can be useful for analyzing trends over time, such as stock prices or website traffic.

One of the key data structures in Pandas is the DataFrame, which is a two-dimensional table that can hold data of different types. You can think of it like a spreadsheet in Excel.

DataFrames allow you to easily manipulate and analyze data using various built-in functions and methods.

Overall, Pandas is a powerful tool for data analysis and manipulation, and is widely used in the data science community.

## Built in functions:

Sure, here are some of the most commonly used built-in functions of Pandas:

1. **read_csv()**: This function is used to read data from a CSV file into a DataFrame.

2. **head()**: This function returns the first n rows of a DataFrame, where n is an integer argument.

3. **tail():** This function returns the last n rows of a DataFrame.

4. **info():** This function provides information about a DataFrame, such as the data types of each column, the number of non-null values, and the memory usage.

5. **describe():** This function provides summary statistics for numerical columns in a DataFrame, such as the mean, standard deviation, minimum, and maximum values.

6. **value_counts()**: This function returns the count of unique values in a column of a DataFrame.

7. **drop():** This function is used to drop rows or columns from a DataFrame.

8. **groupby()**: This function is used to group rows of a DataFrame by one or more columns, and then perform operations on each group.

9. **merge()**: This function is used to merge two or more DataFrames based on a common column.

10. **pivot_table()**: This function is used to create a summary table of a DataFrame, where the rows and columns are based on different variables.

These are just a few examples of the many built-in functions provided by Pandas. Each function performs a specific task related to data analysis and manipulation.

## Scikit-Image:

Scikit-image is a Python library used for image processing and computer vision tasks. It provides a collection of algorithms and tools for tasks such as image filtering, segmentation, feature extraction, and more.

In simpler terms, scikit-image is like a toolbox for working with images in Python. You can use it to perform various operations on images, such as blurring, sharpening, edge detection, and object detection. It also provides tools for working with color images and for measuring various properties of objects in an image.

Scikit-image is built on top of NumPy, another popular Python library for scientific computing, and integrates well with other Python libraries such as Matplotlib and OpenCV.

Some of the key features of scikit-image include:

1. **Image filtering**: scikit-image provides functions for various types of image filtering, such as Gaussian smoothing, median filtering, and bilateral filtering.

2. **Image segmentation**: scikit-image provides functions for segmenting images into regions, such as thresholding, watershed segmentation, and region growing.

3. **Feature extraction**: scikit-image provides functions for extracting various types of features from images, such as texture features, shape features, and edge features.

4. **Object detection:** scikit-image provides functions for detecting objects in images, such as template matching and blob detection.

Overall, scikit-image is a powerful tool for working with images in Python, and is widely used in the computer vision community.

## Built in functions: here are some of the most commonly used built-in functions in scikit-image:

1. **imread()**: This function is used to read an image file into a NumPy array.

2. **imshow()**: This function is used to display an image from a NumPy array.

3. **imsave()**: This function is used to save a NumPy array as an image file.

4. **color.rgb2gray()**: This function converts a color image to a grayscale image.

5. **filters.gaussian()**: This function applies a Gaussian filter to an image.

6. **filters.sobel():** This function applies a Sobel filter to an image to detect edges.

7. **feature.canny()**: This function applies the Canny edge detection algorithm to an image.

8. **morphology.dilation()**: This function performs image dilation, which expands the boundaries of objects in an image.

9. **segmentation.slic()**: This function performs image segmentation using the SLIC algorithm.

10. **measure.regionprops()**: This function computes various properties of objects in an image, such as area, perimeter, and centroid.

## Numpy:

NumPy is a Python library used for scientific computing and numerical analysis. It provides support for large, multi-dimensional arrays and matrices, along with a wide range of mathematical functions to operate on these arrays.

In simpler terms, NumPy is like a toolbox for working with numbers and arrays in Python. It allows you to create and manipulate arrays of numbers efficiently, and provides a range of functions for performing mathematical operations on these arrays. NumPy is particularly useful for tasks such as data analysis, machine learning, and image processing, where large amounts of numerical data need to be processed quickly.

Some of the key features of NumPy include:

1. **Multi-dimensional arrays**: NumPy provides support for arrays with any number of dimensions, from simple one-dimensional arrays to complex multi-dimensional arrays.

2. **Mathematical functions**: NumPy provides a wide range of mathematical functions, such as trigonometric functions, logarithmic functions, and exponential functions, that can be applied to arrays of numbers.
3. **Broadcasting**: NumPy allows you to perform operations on arrays of different shapes and sizes, by automatically broadcasting the smaller array to match the larger array.

4. **Linear algebra**: NumPy provides a range of functions for performing linear algebra operations on arrays, such as matrix multiplication, eigenvalue computation, and singular value decomposition.

## Built in functions:

Here are some of the most commonly used built-in functions in NumPy:

1. **np.array()**: This function is used to create a new NumPy array from a Python list or tuple.

2. **np.arange()**: This function creates an array of evenly spaced values between a start and end point, with a specified step size.

3. **np.zeros()**: This function creates an array of all zeros with a specified shape.

4. **np.ones()**: This function creates an array of all ones with a specified shape.

5. **np.random.rand()**: This function creates an array of random numbers between 0 and 1 with a specified shape.

6. **np.reshape():** This function is used to change the shape of an existing array.

7. **np.concatenate()**: This function is used to combine two or more arrays along a specified axis.

8. **np.max()**: This function returns the maximum value in an array.

9. **np.min()**: This function returns the minimum value in an array.

10. **np.mean()**: This function returns the mean (average) value of an array.