Machine Learning

Introduction

- Technology that enables computers to learn automatically from past data.
- Uses various algorithms for building mathematical models and making predictions using historical data or information
- Used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging and many more.
- Techniques example – Supervised, Unsupervised and Reinforcement learnings.
- Machine Learning – Machine learns from experiences or past data like a human does

- Machine Learning – subset of artificial intelligence
- Main concern – development of algorithms which allow a computer to learn from the data and past experiences
- Machine Learning – Term – introduced by Arthur Samuel – 1959

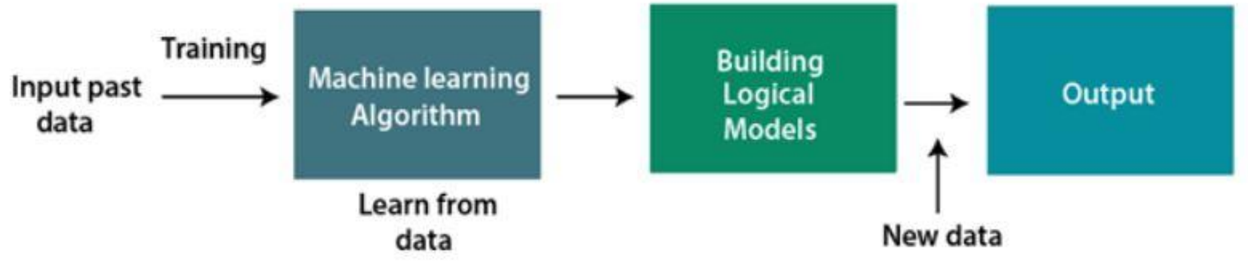
Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.

- Historical data – training data
- Help of historical data, Machine learning algorithms build a mathematical model that helps in making predictions / decision making without explicitly programmed.
- The more we will provide the information, the higher will be the performance

A machine has the ability to learn if it can improve its performance by gaining more data.

Machine Learning Work Flow

- A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it.
- The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.
- Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output.



Features

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

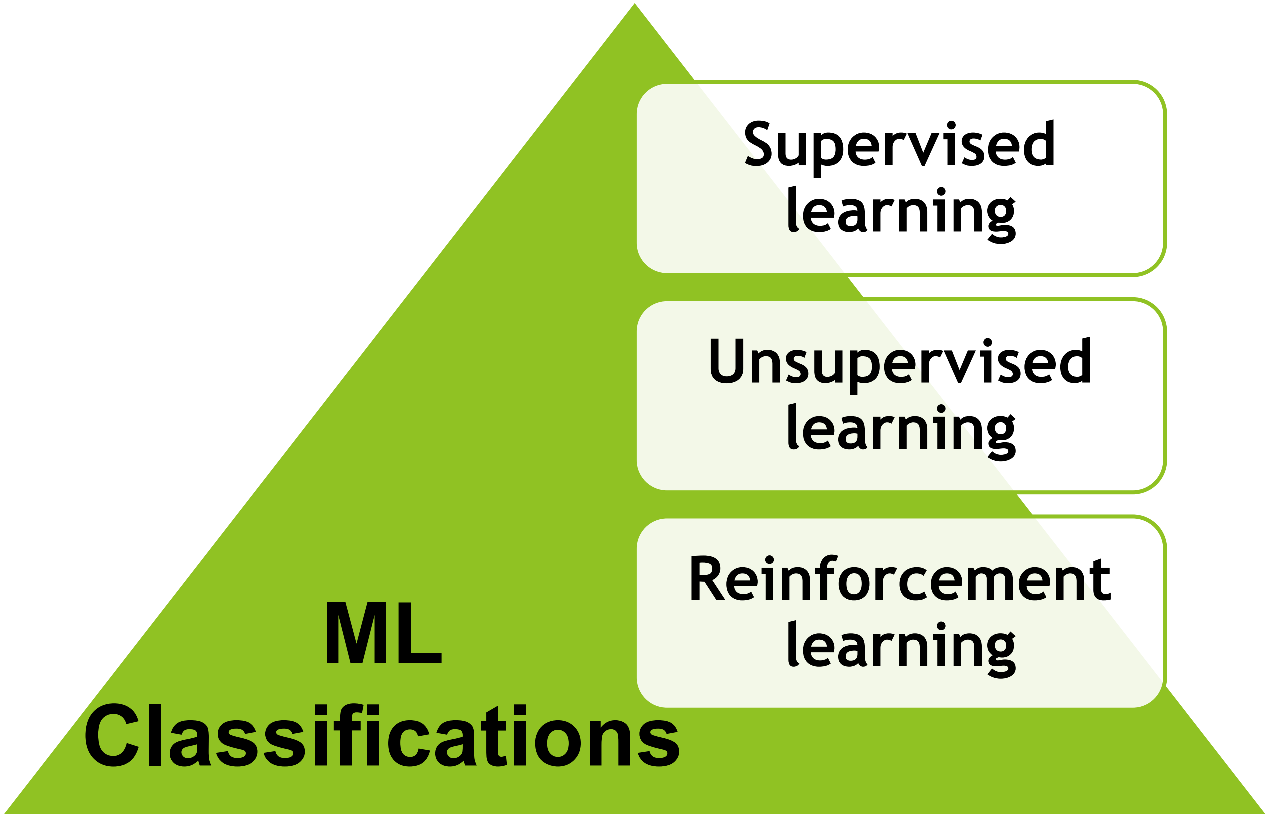
The Need

- The reason behind the need for machine learning is that it is capable of doing tasks that are too complex for a person to implement directly.
- we have some limitations as we cannot access the huge amount of data manually, so for this, we need some computer systems and here comes the machine learning to make things easy for us.
- We can train machine learning algorithms by providing them the huge amount of data and let them explore the data, construct the models, and predict the required output automatically.
- The performance of the machine learning algorithm depends on the amount of data, and it can be determined by the cost function
- we can save both time and money.

- The importance of machine learning can be easily understood by its uses cases
- machine learning is used in **self-driving cars**, **cyber fraud detection**, **face recognition**, and **friend suggestion by Facebook**, etc.
- Various top companies such as Netflix and Amazon have build machine learning models that are using a vast amount of data to analyze the user interest and recommend product accordingly.

Importance

- Rapid increment in the production of data
- Solving complex problems, which are difficult for a human
- Decision making in various sector including finance
- Finding hidden patterns and extracting useful information from data.



ML Classifications

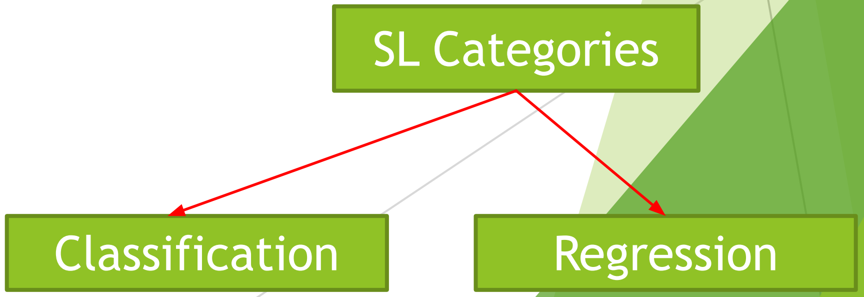
Supervised learning

Unsupervised learning

Reinforcement learning

Supervised Learning

- machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.
- The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.
- The goal of supervised learning is to map input data with the output data.
- The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher.
- The example of supervised learning is **spam filtering**.



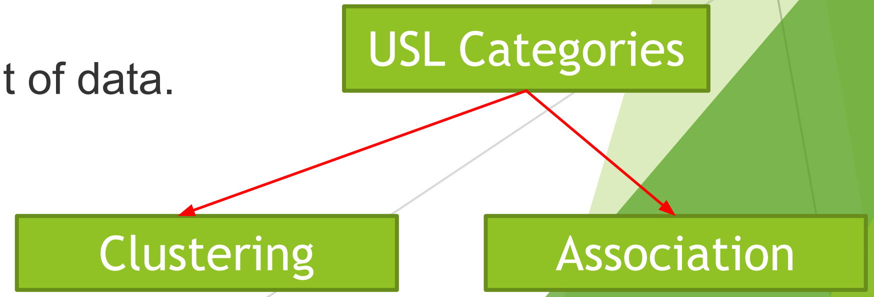
SL Categories

Classification

Regression

Unsupervised Learning

- Unsupervised learning is a learning method in which a machine learns without any supervision.
- The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision.
- The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.
- In unsupervised learning, we don't have a predetermined result.
- The machine tries to find useful insights from the huge amount of data.



USL Categories

Clustering

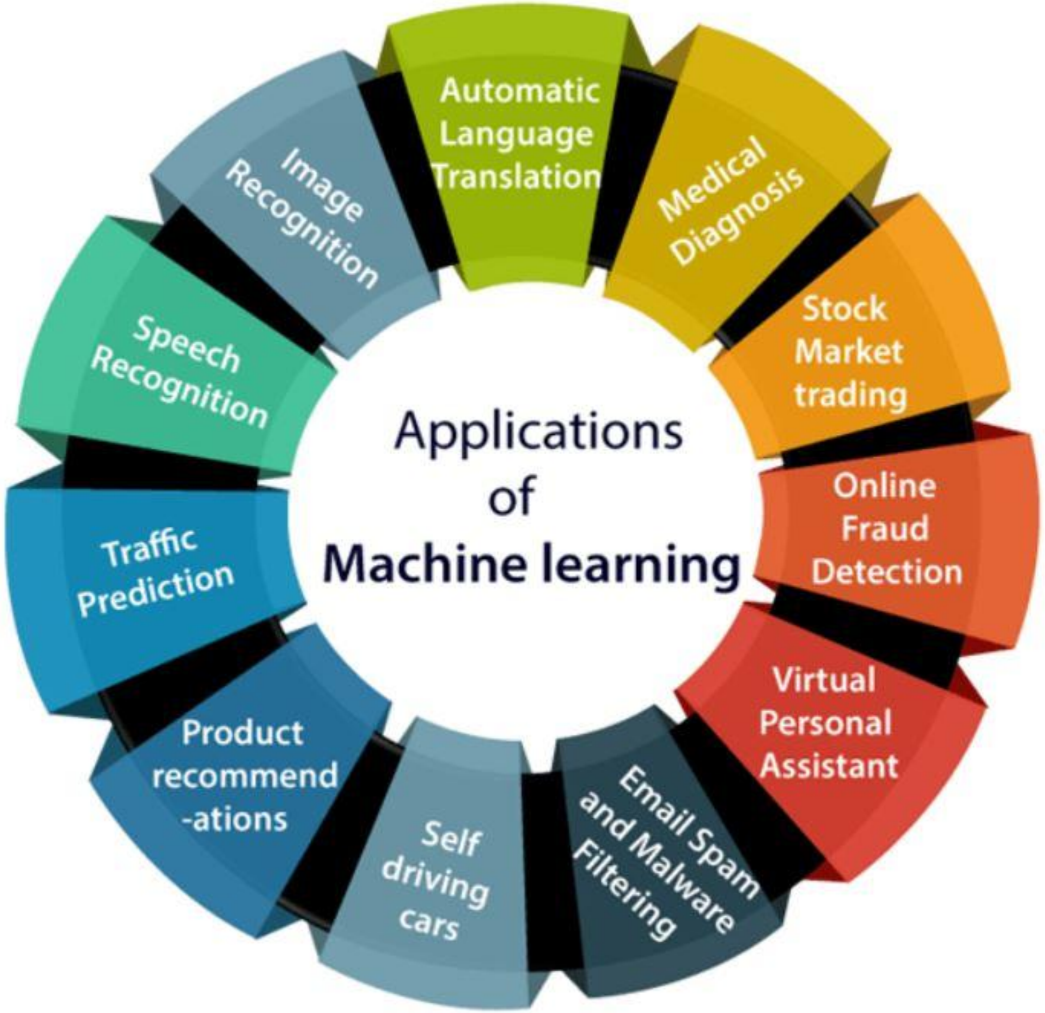
Association

Reinforcement Learning

- Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.
- The agent learns automatically with these feedbacks and improves its performance.
- In reinforcement learning, the agent interacts with the environment and explores it.
- The goal of an agent is to get the most reward points, and hence, it improves its performance.
- The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.

Applications

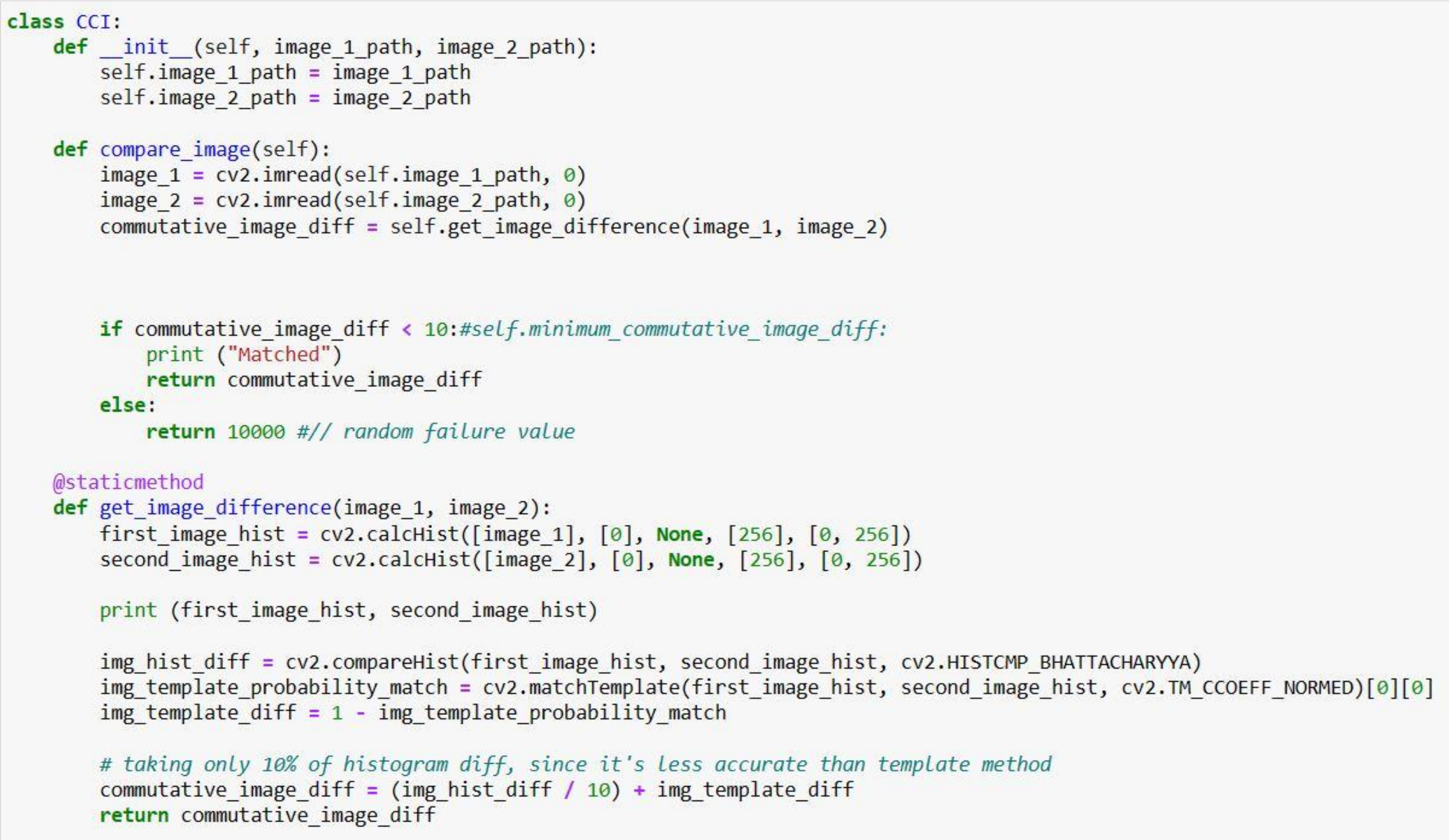
- We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc.



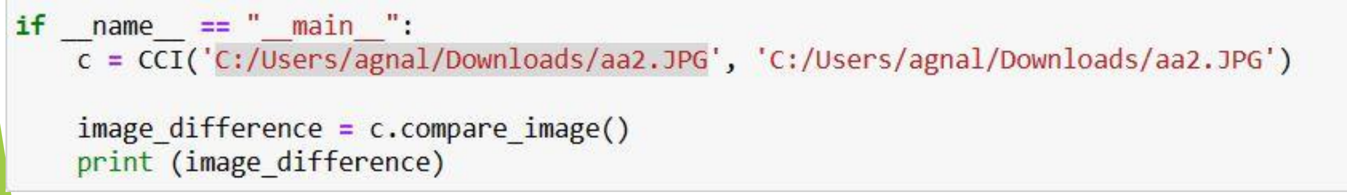
Applications of Machine learning
Automatic Language Translation
Medical Diagnosis
Stock Market trading
Online Fraud Detection
Virtual Personal Assistant
Email Spam and Malware Filtering
Self driving cars
Product recommend -ations
Traffic Prediction
Speech Recognition
Image Recognition

Image Recognition

- Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc.
- The popular use case of image recognition and face detection is, **Automatic friend tagging suggestion**:
- Facebook provides us a feature of auto friend tagging suggestion.
- Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's **face detection** and **recognition algorithm**.
- It is based on the Facebook project named "**Deep Face**," which is responsible for face recognition and person identification in the picture.



```python
class CCI:
    def __init__(self, image_1_path, image_2_path):
        self.image_1_path = image_1_path
        self.image_2_path = image_2_path

    def compare_image(self):
        image_1 = cv2.imread(self.image_1_path, 0)
        image_2 = cv2.imread(self.image_2_path, 0)
        commutative_image_diff = self.get_image_difference(image_1, image_2)



        if commutative_image_diff < 10:#self.minimum_commutative_image_diff:
            print ("Matched")
            return commutative_image_diff
        else:
            return 10000 #// random failure value

    @staticmethod
    def get_image_difference(image_1, image_2):
        first_image_hist = cv2.calcHist([image_1], [0], None, [256], [0, 256])
        second_image_hist = cv2.calcHist([image_2], [0], None, [256], [0, 256])

        print (first_image_hist, second_image_hist)

        img_hist_diff = cv2.compareHist(first_image_hist, second_image_hist, cv2.HISTCMP_BHATTACHARYYA)
        img_template_probability_match = cv2.matchTemplate(first_image_hist, second_image_hist, cv2.TM_CCOEFF_NORMED)[0][0]
        img_template_diff = 1 - img_template_probability_match

        # taking only 10% of histogram diff, since it's less accurate than template method
        commutative_image_diff = (img_hist_diff / 10) + img_template_diff
        return commutative_image_diff
```

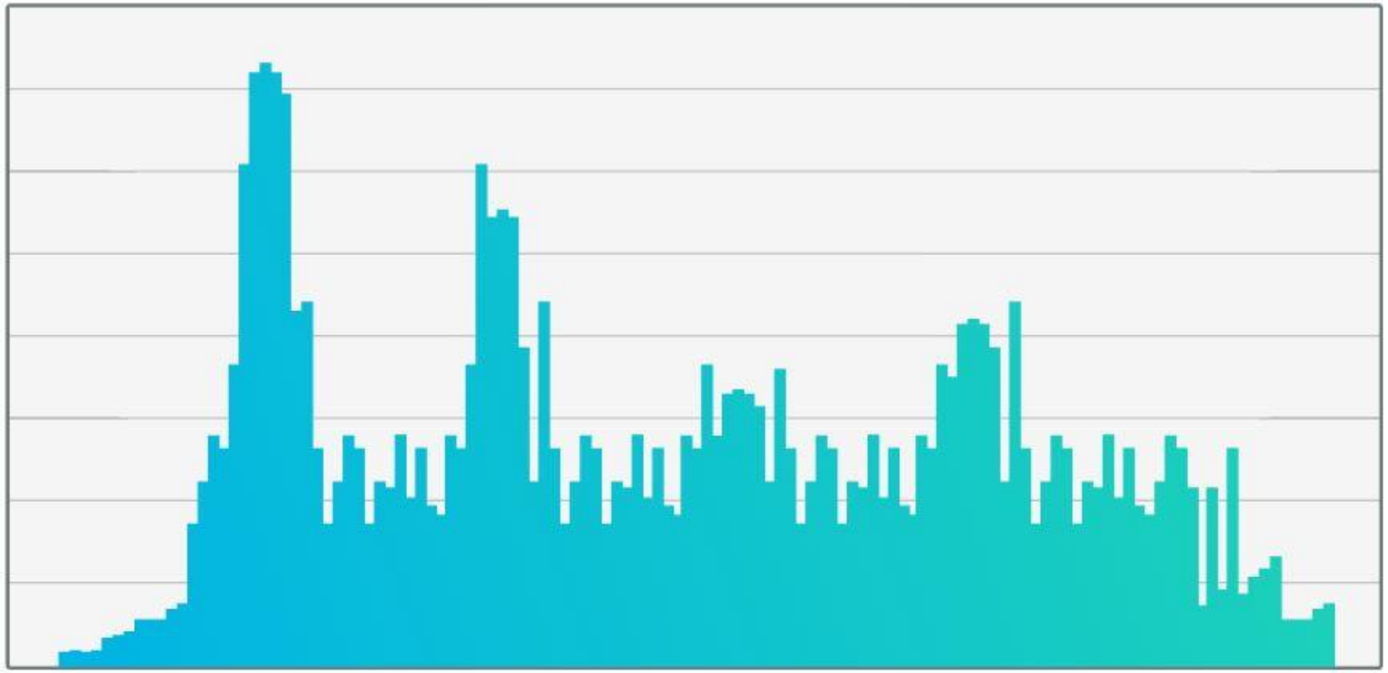


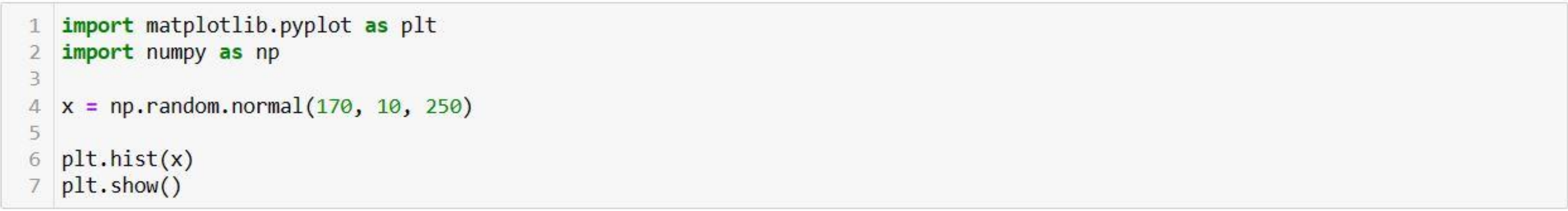
```python
if __name__ == "__main__":
    c = CCI('C:/Users/agnal/Downloads/aa2.JPG', 'C:/Users/agnal/Downloads/aa2.JPG')

    image_difference = c.compare_image()
    print (image_difference)
```

What is a Histogram Chart?

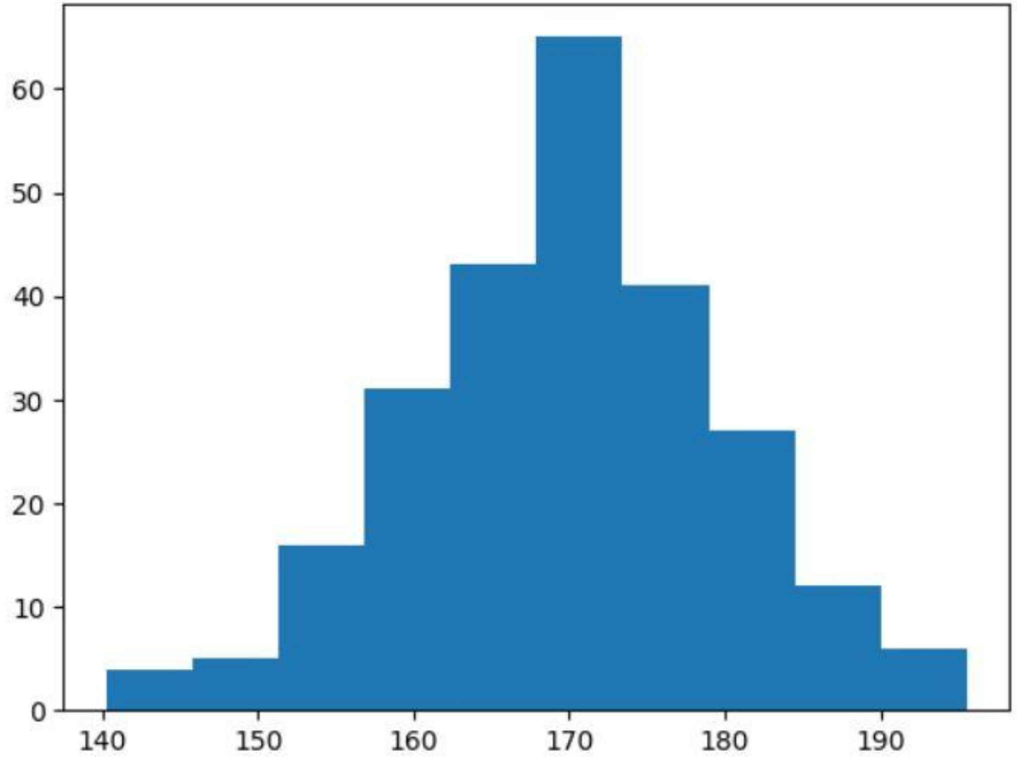
A **histogram** is a graph used to represent the frequency distribution of a few data points of one variable.

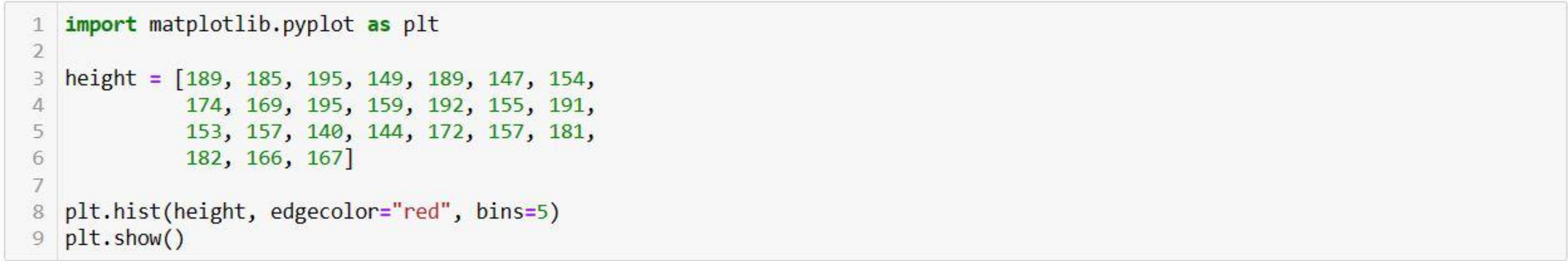
Histograms often classify data into various “bins” or “range groups” and count how many data points belong to each of those bins.



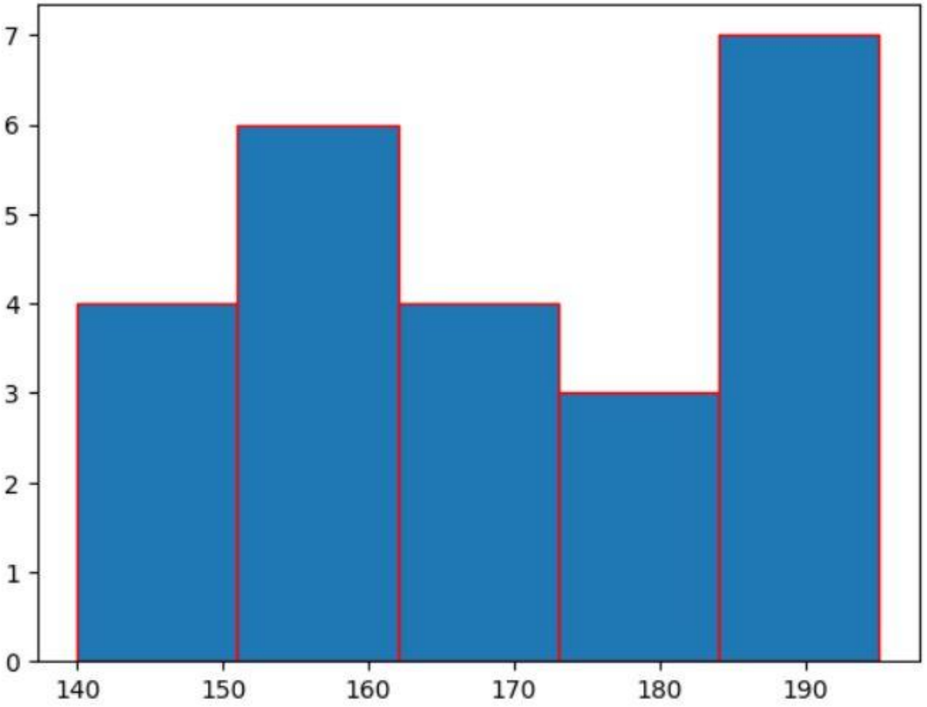


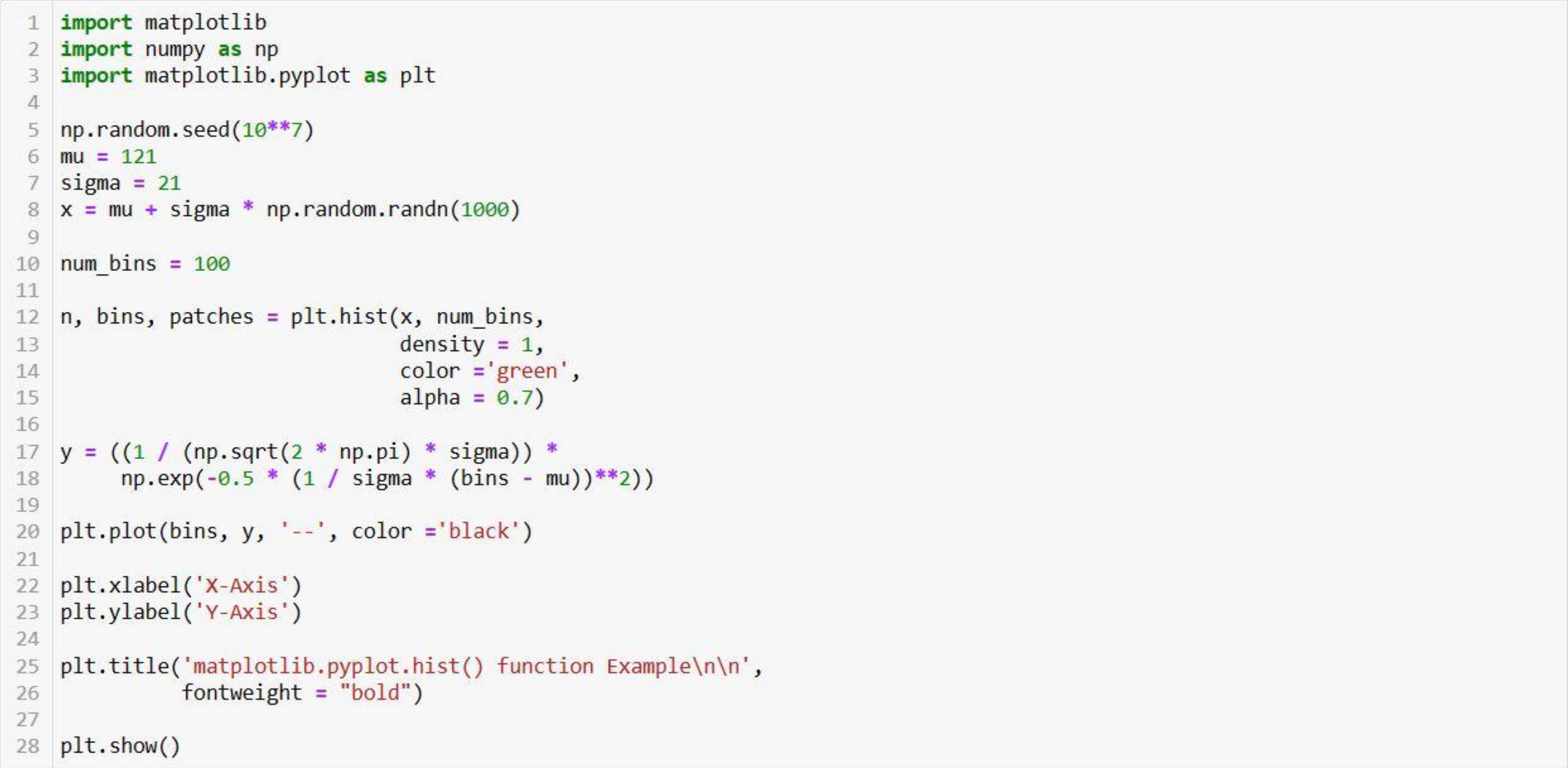
```python
import matplotlib.pyplot as plt
import numpy as np

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

plt.hist(x)
plt.show()
```





```python
import matplotlib.pyplot as plt

height = [189, 185, 195, 149, 189, 147, 154,
          174, 169, 195, 159, 192, 155, 191,
          153, 157, 140, 144, 172, 157, 181,
          182, 166, 167]

plt.hist(height, edgecolor="red", bins=5)
plt.show()
```





```python
import matplotlib
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(10**7)
mu = 121
sigma = 21
x = mu + sigma * np.random.randn(1000)

num_bins = 100

n, bins, patches = plt.hist(x, num_bins,
                            density = 1,
                            color ='green',
                            alpha = 0.7)

y = ((1 / (np.sqrt(2 * np.pi) * sigma)) *
     np.exp(-0.5 * (1 / sigma * (bins - mu))**2))

plt.plot(bins, y, '--', color ='black')

plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.title('matplotlib.pyplot.hist() function Example\n\n',
          fontweight = "bold")

plt.show()
```

compare the histograms for the notion of similarity

This comparison is possible because we can classify a number of things around us based on color.

various single number evaluation metrics that tell how well two histograms match with each other.

The histogram comparison methods can be classified into two categories

- Bin-to-Bin comparison
- Cross-bin comparison

Bin-to-Bin comparison methods include L1, L2 norm for calculating the bin distances or bin intersection, etc.

These methods assume that the histogram domains are aligned but this condition is easily violated in most of the cases due to change in lighting conditions, quantization, etc.

Cross bin comparison methods are more robust and discriminative but this can be computationally expensive.

To circumvent this, one can reduce the cross bin comparison to bin-to-bin.

Cross bin comparison methods include Earthmoving distance (EMD), quadratic form distances (taking into account the bin similarity matrix), etc.

```
cv2.compareHist(H1, H2, method)
```

Here, H1 and H2 are the histograms we want to compare and the “method” argument specifies the comparison method.

OpenCV provides several built-in methods for histogram comparison as shown below

- **HISTCMP_CORREL**: Correlation
- **HISTCMP _CHISQR**: Chi-Square
- **HISTCMP _CHISQR_ALT**: Alternative Chi-Square
- **HISTCMP _INTERSECT**: Intersection
- **HISTCMP _BHATTACHARYYA**: Bhattacharyya distance
- **HISTCMP _HELLINGER**: Synonym for CV_COMP_BHATTACHARYYA
- **HISTCMP _KL_DIV**: Kullback-Leibler divergence

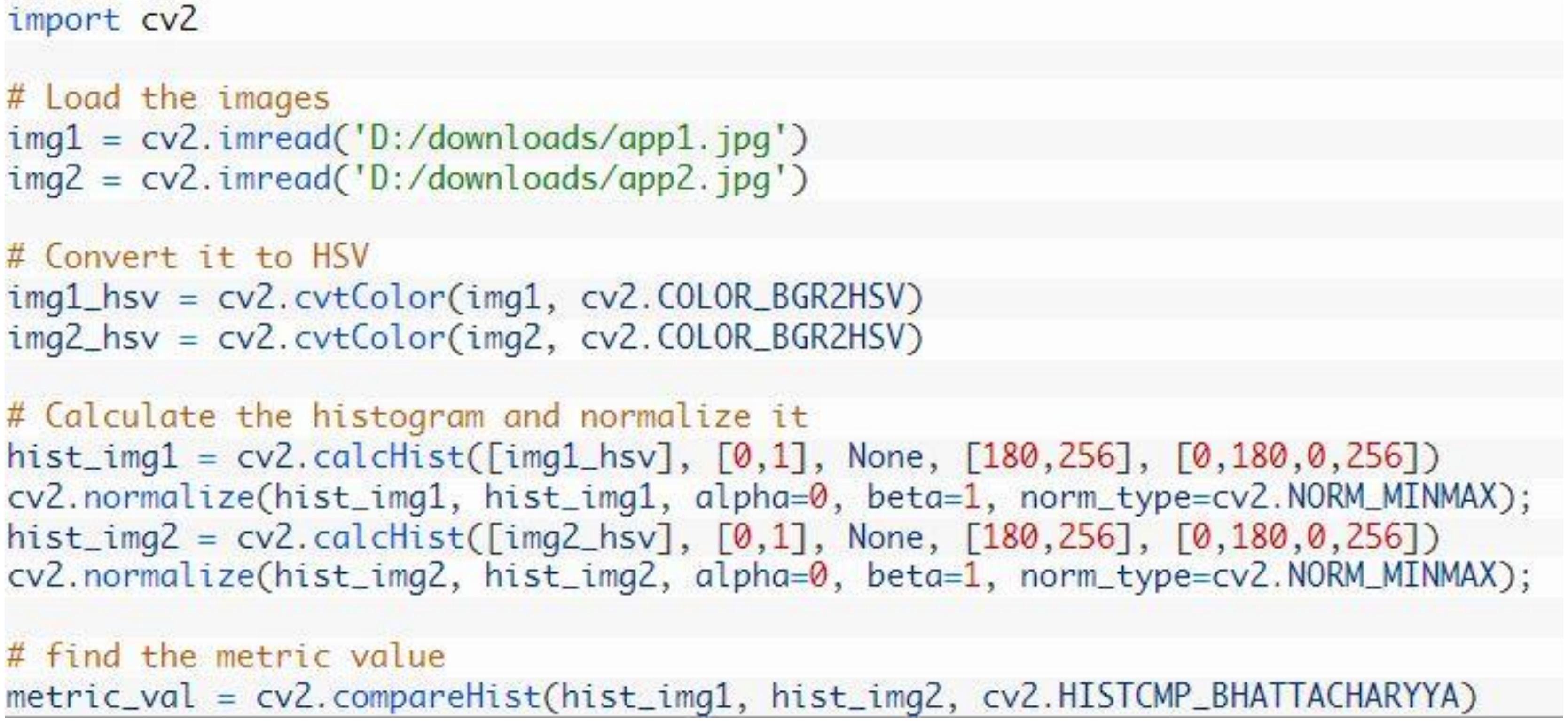
For the **Correlation** and **Intersection** methods, the higher the metric, the more accurate the match.

While for **chi-square** and **Bhattacharyya**, the lower metric value represents a more accurate match.

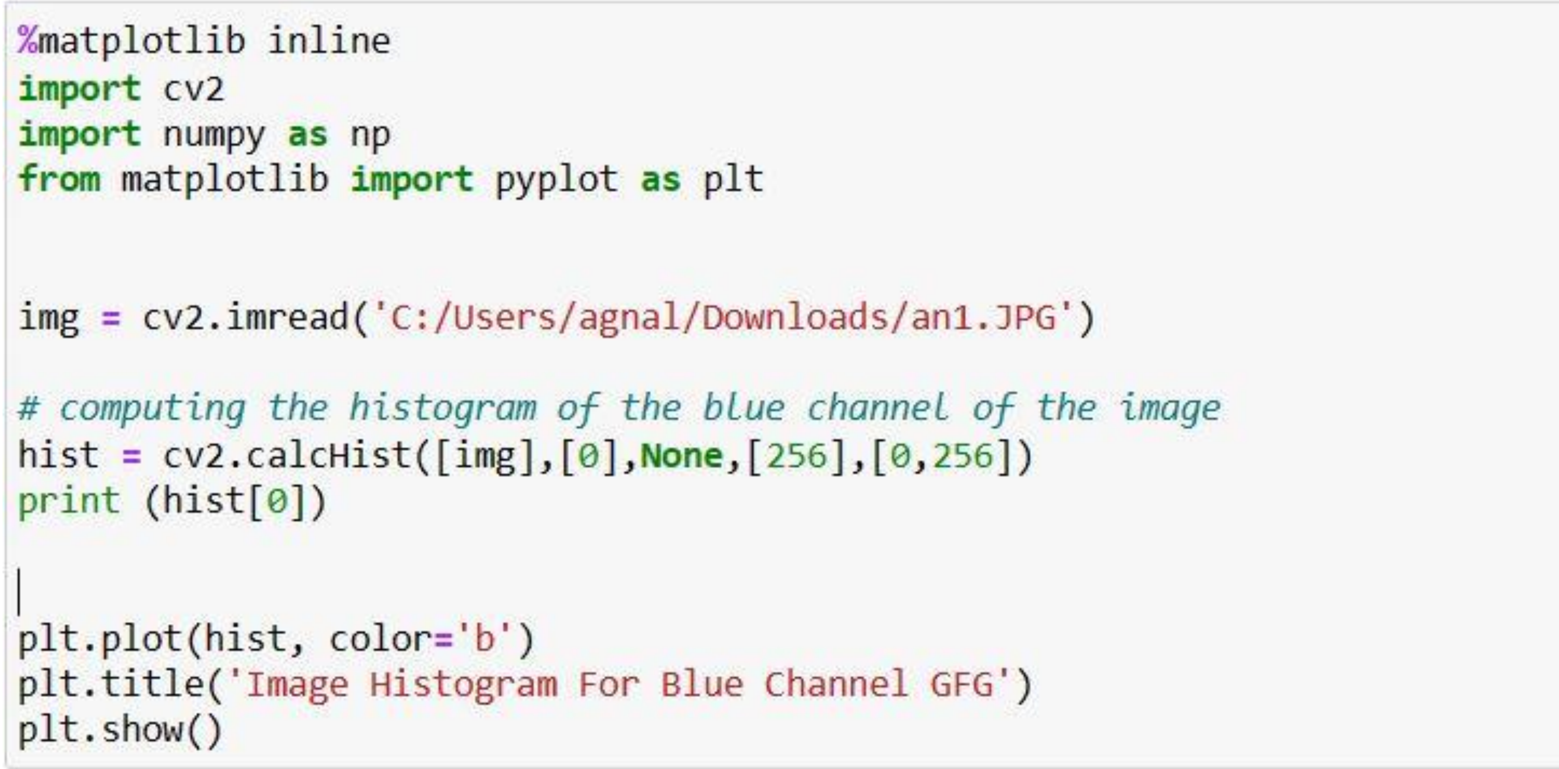
Now, let's take an example to understand how to use this function.

Here, we will compare the two images as shown below.

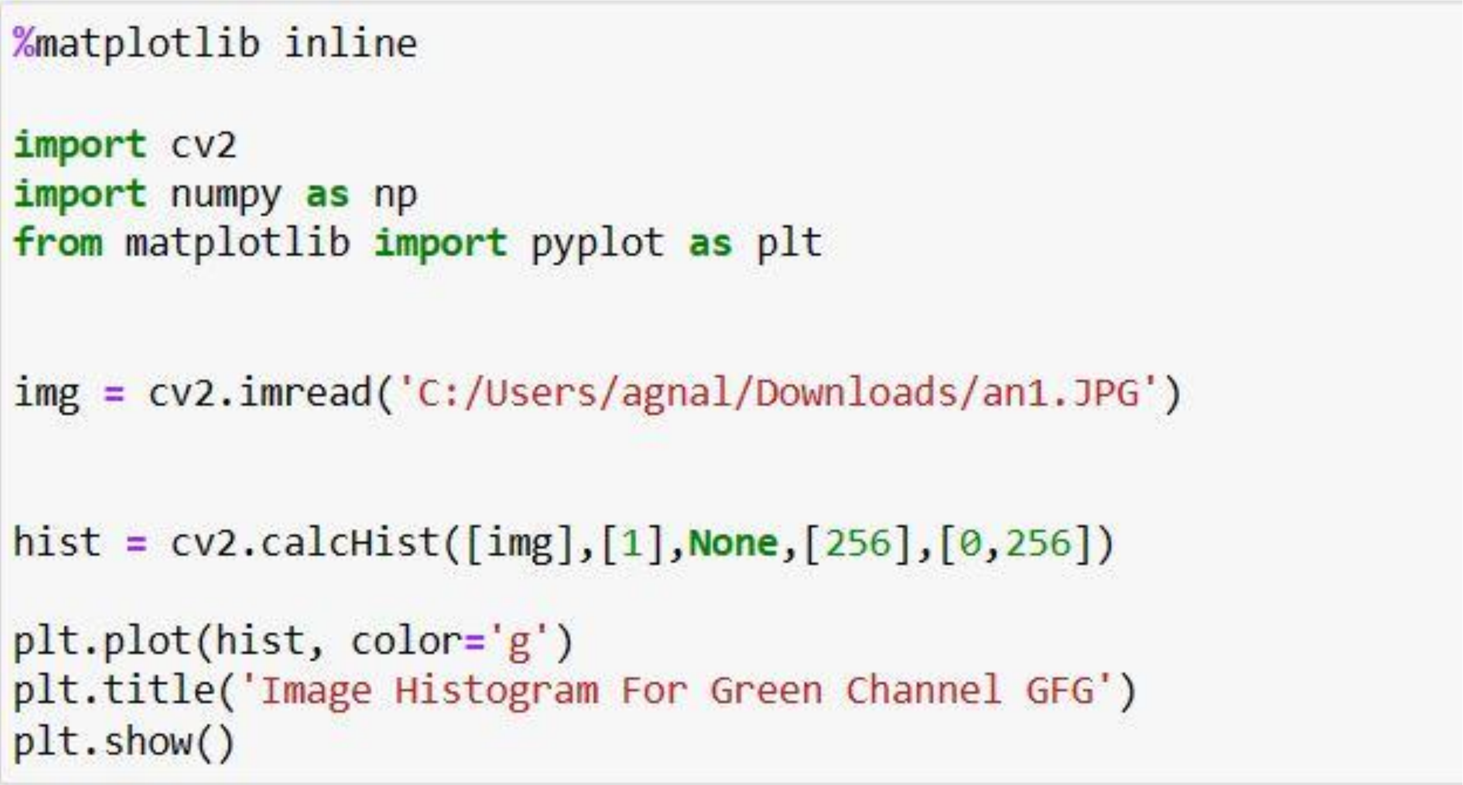
- Load the images
- Convert it into any suitable color model
- Calculate the image histogram (2D or 3D histograms are better) and normalize it
- Compare the histograms using the above function



```
import cv2

# Load the images
img1 = cv2.imread('D:/downloads/app1.jpg')
img2 = cv2.imread('D:/downloads/app2.jpg')

# Convert it to HSV
img1_hsv = cv2.cvtColor(img1, cv2.COLOR_BGR2HSV)
img2_hsv = cv2.cvtColor(img2, cv2.COLOR_BGR2HSV)

# Calculate the histogram and normalize it
hist_img1 = cv2.calcHist([img1_hsv], [0,1], None, [180,256], [0,180,0,256])
cv2.normalize(hist_img1, hist_img1, alpha=0, beta=1, norm_type=cv2.NORM_MINMAX);
hist_img2 = cv2.calcHist([img2_hsv], [0,1], None, [180,256], [0,180,0,256])
cv2.normalize(hist_img2, hist_img2, alpha=0, beta=1, norm_type=cv2.NORM_MINMAX);

# find the metric value
metric_val = cv2.compareHist(hist_img1, hist_img2, cv2.HISTCMP_BHATTACHARYYA)
```



```
%matplotlib inline
import cv2
import numpy as np
from matplotlib import pyplot as plt


img = cv2.imread('C:/Users/agnal/Downloads/an1.JPG')

# computing the histogram of the blue channel of the image
hist = cv2.calcHist([img],[0],None,[256],[0,256])
print (hist[0])


plt.plot(hist, color='b')
plt.title('Image Histogram For Blue Channel GFG')
plt.show()
```



```
%matplotlib inline

import cv2
import numpy as np
from matplotlib import pyplot as plt


img = cv2.imread('C:/Users/agnal/Downloads/an1.JPG')


hist = cv2.calcHist([img],[1],None,[256],[0,256])

plt.plot(hist, color='g')
plt.title('Image Histogram For Green Channel GFG')
plt.show()
```