

Image Processing with Python

Introduction to Python

What is Python

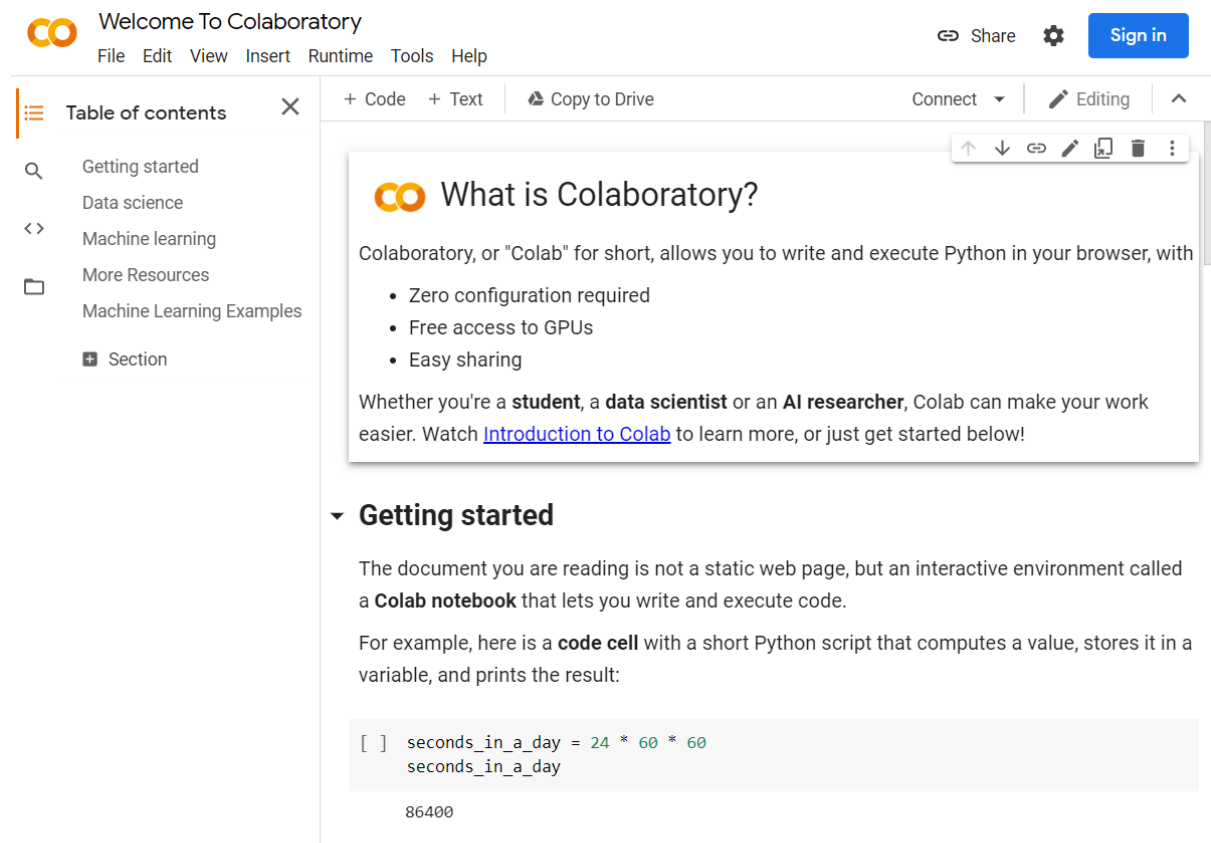
Python is a programming language that lets you work quickly and integrate systems more effectively. It's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Why should you know Python?

In academic and research we can use Python and its lots of libraries to do a wide range of things from basic calculation to complex statistical methods including machine learning. Since it is open-source, it is free to use.

What is Google Coalb?

Colaboratory is an online workspace for Python programming, which is developed by the Google Research team. It has several features that improve learning experience, such as Pre-Installed libraries, Stored on the cloud, Linked with Google drive, Collaborative coding, Integrated GPU and TPU, and It is free to use!



The screenshot displays the Google Colaboratory web interface. At the top, there's a 'Welcome To Colaboratory' header with a menu (File, Edit, View, Insert, Runtime, Tools, Help) and a 'Sign in' button. A left sidebar contains a 'Table of contents' with links to 'Getting started', 'Data science', 'Machine learning', 'More Resources', 'Machine Learning Examples', and a 'Section' button. The main area shows a tutorial titled 'What is Colaboratory?'. The tutorial text explains that Colab allows writing and executing Python in the browser, listing features like zero configuration, free GPU access, and easy sharing. It also mentions that Colab is suitable for students, data scientists, and AI researchers. Below the text, there's a 'Getting started' section that describes the Colab notebook as an interactive environment. It includes a code cell with a Python script to calculate the number of seconds in a day, which is shown as 86400.

Welcome To Colaboratory

File Edit View Insert Runtime Tools Help

Share Sign in

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Machine Learning Examples
- Section

What is Colaboratory?

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser, with

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
```

86400

Basic Data Structure

Type	Class	Explanation
None	NoneType	This is representing absence of a value.
		>> None
Numbers	Booleans	This is representing the truth value of an expression..
		>> Ture, False
	Integer	This is a positive or negative number without a decimal point.
		>> 1, 2, 3, -99
	Float	This is a positive or negative number with floating point precision.
		>> 1., .2, 3.4, -5.6
	Complex	This is containing 2 numbers as real and imaginary parts.
		>> 1 + 1j
Sequences	String	This is an immutable sequence of characters.
		>> 'Hello World'
	Tuple	This is an immutable sequence of elements.
		>> (True, 2, .3, '4', [5])
	List	This is a mutable sequence of elements.
		>> [True, 2, .3, '4', [5]]
Mappings	Dictionary	This is for storing key and value pairs.
		>> {'one': 1, 'two': 2, 'three', 3}

Arithmetic Operation

Operator	Name	Example	
+	Addition	>> 7 + 4	>> 11
-	Subtraction	>> 7 - 4	>> 3
*	Multiplication	>> 7 * 4	>> 28

/	Division	>> 7 / 4	>> 1.75
%	Modulus	>> 7 % 4	>> 3
**	Exponentiation	>> 7 ** 4	>> 2401
//	Floor division	>> 7 // 4	>> 1

Common Keywords

>> help('keywords')		To print out all keywords in Python		
constant values	True	False	None	
logical operation	and	or	not	
verify variable	is	in		
condition	if	elif	else	
start the loop	while	for		
loop's actions	pass	continue	break	
create function	def	lambda		
end function	return	yield		
create class	class			
error handling	try	except	else	finally
import package	import	as	from	

Built-in Functions

Function	Explanation
print()	To print out the argument.
help()	To print out the documentation of the argument.
type()	Return class type of the argument.
max()	Return item with the highest value.
min()	Return item with the lowest value.
abs()	Return the absolute number.

range()	Return the sequence of numbers.
len()	Return the number of items.
sum()	Return the summation of all items.
sorted()	Return the iterable with specific order.

Precedence of Operators

Operator	Name
()	Parentheses
**	Exponentiation
*, /, //, %	Multiplication, Division, Modulus, Floor division
+, -	Addition, Subtraction
&	Bitwise AND
^	Bitwise XOR
 	Bitwise OR
==, !=, >, >=, <, <=, is	Comparisons, Identity
not	Logical NOT
and	Logical AND
or	Logical OR

Exercise 1.) Calculate factorial of these numbers.

- a. 5
- b. 15
- c. 25

Exercise 2.) Remove prime numbers from these lists.

- a. [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
- b. [8830, 8831, 8832, 8833, 8834, 8835, 8836, 8837, 8838, 8839]
- c. [-37, 3/7, 7//3, '3', 7, 7.3, 3+7, 37, 73, 703]

Basic Image Processing

RGB Colorspace

RGB is the most common way to describe the chromaticity of the color. It consists of three values that represent the light intensity of each frequency. For example, R is referred to red color which have the wavelength peaking around 570 nm, G is green having peaking around 540 nm, and B is blue having peaking around 430 nm. By mixing these 3 values, we can generate any colors within the human perception. This color space is also used as the base unit of display monitors.

HSV Colorspace

HSV is an alternative way to explain colors. This color space can be modeled into the cylinder shape, which has the rainbow radial slice as the upper face, pure black as the lower face, and other colors are in between. This model was designed to reflect the way humans describe each color. It is separated into 3 components. H or hue represents color as angle. S or saturation represents expressiveness of color, it is covering from neutral to colorful. V or value represents lightness, it is covering from dark to bright.

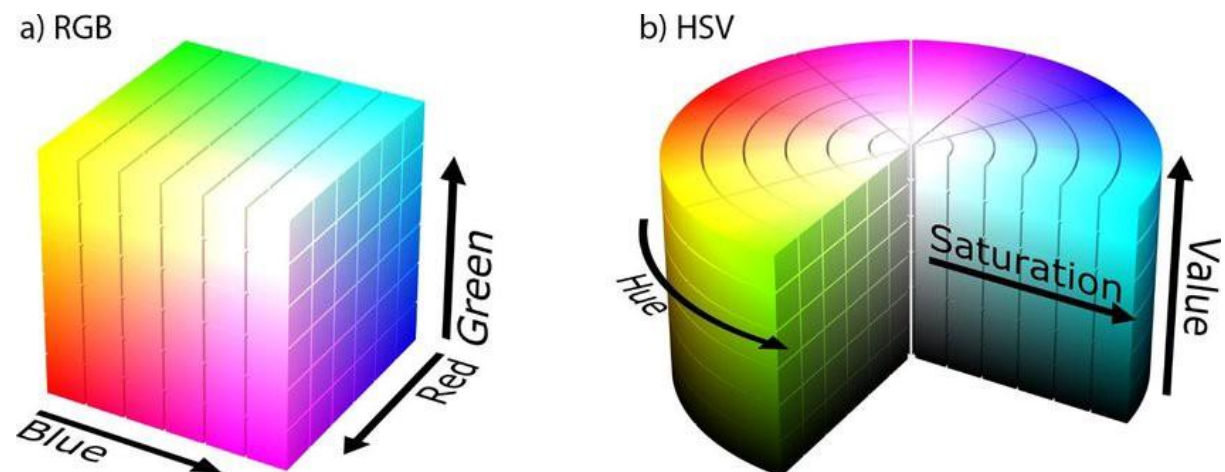


Image Thresholding

This is a technique in computer vision to determine image pixels property based on their intensity. The trick is straight forward. If a pixel value is higher than a threshold value, it is assigned one value, else it is assigned another value. This technique can be used for extracting regions of interest from the image as the mask.

Bitwise Operators

These are techniques for manipulating images. By applying pointwise logic operation between images, we can generate many applications, for example inverting an image, overlaying masks over an image, and removing the background.

Morphological Transformations

These are techniques for operating binary images based on its structure. By applying a kernel to the structure boundary, we can generate erosion, dilation, and noise removed images.

Common CV2 Functions

Function	Explanation
<code>np.zeros()</code>	To generate a Numpy array with member value = 0
<code>np.ones()</code>	To generate a Numpy array with member value = 1
<code>cv2.imread()</code>	To load an image to a Numpy array.
<code>cv2.cvtColor()</code>	To change the color space of an image.
<code>cv2.resize()</code>	To change the dimension of an image.
<code>cv2.split()</code>	To split an image to channels.
<code>cv2.merge()</code>	To merge channels to an image.
<code>cv2.threshold()</code>	To apply thresholding technique.
<code>cv2.bitwise_not()</code>	To operate logic NOT on an image.
<code>cv2.bitwise_and()</code>	To operate logic AND between 2 images.
<code>cv2.bitwise_or()</code>	To operate logic OR between 2 images.
<code>cv2.bitwise_xor()</code>	To operate logic XOR between 2 images.
<code>cv2.erode()</code>	To operate erosion morphological operation.
<code>cv2.dilate()</code>	To operate dilation morphological operation.
<code>cv2.morphologyEx()</code>	To operate morphological operations.
<code>cv2.findContours()</code>	To get contours of shapes in a binary image.

Exercise 3.) Enhance this image.

Exercise 4.) Turn these apples green!

Exercise 5.) Count Mitosis cells.

Assignment 1.) Classify Malaria infected cells.

Assignment 2.) Detect and classify pills in these images.