The smallest element of an image is called a **pixel**, or a picture element. It is basically a dot in the picture. An image contains multiple pixels arranged in rows and columns.

You will often see the number of rows and columns expressed as the image **resolution**. For example, an Ultra HD TV has the resolution of 3840x2160, meaning it is 3840 pixels wide and 2160 pixels high.

But a computer does not understand pixels as dots of color. It only understands numbers. To convert colors to numbers, the computer uses various color models.

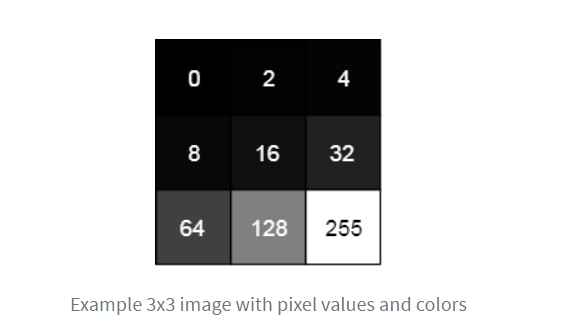
In color images, pixels are often represented in the RGB color model. RGB stands for **R**ed **G**reen **B**lue. Each pixel is a mix of those three colors. RGB is great at modeling all the colors humans perceive by combining various amounts of red, green, and blue.

Since a computer only understand numbers, every pixel is represented by three numbers

corresponding to the amounts of red, green, and blue present in that pixel

1. Grayscale(black and white) images

* Each pixel is represented by three numbers, corresponding to the amounts of red, green and blue present in that pixel.
* In many applications, the range of intensities is from 0 (black) to 255 (white). Everything between 0 and 255 is various shades of gray
* If each grayscale pixel is a number, an image is nothing more than a matrix (or table) of numbers:



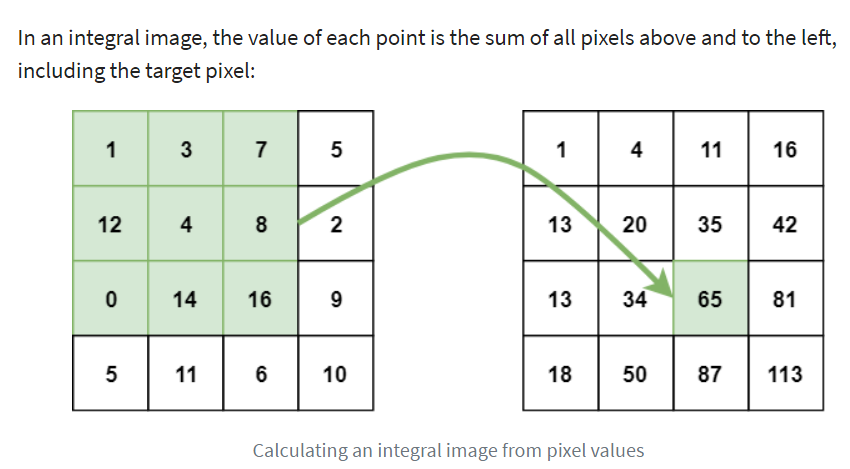
* BigNote: in color image, there are three such matrices representing the red, green and blue channels.

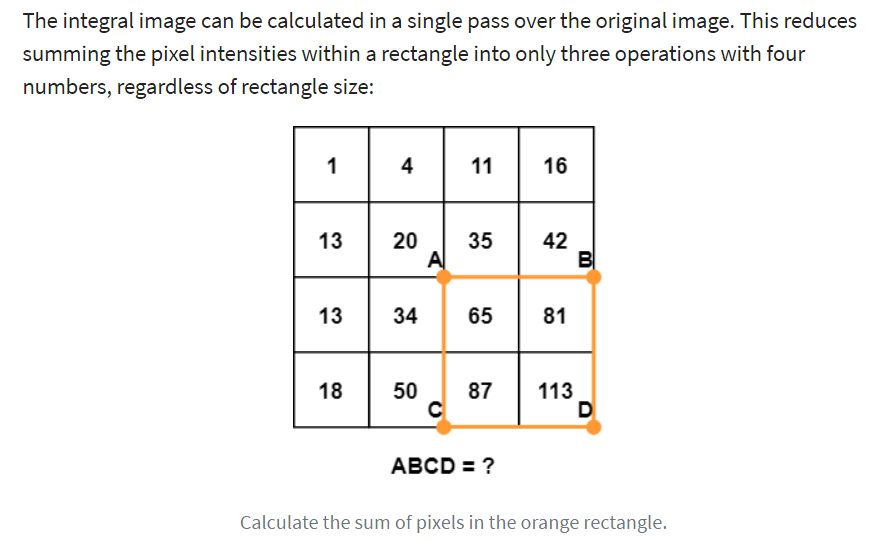
1. What are feature

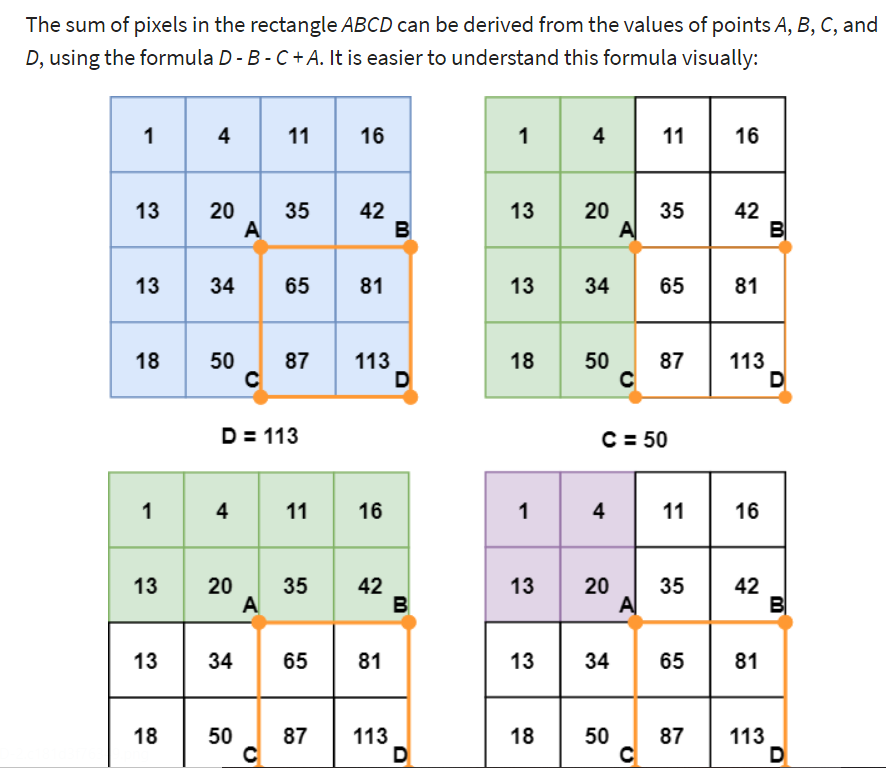
* A feature is a piece of information in an image that is relevant to solving a certain problem.
* Could be simple like a single pixel value, or more complex like edges, corners, and shapes. You can combine multiple simple features into a complex feature.

1. Harlike feature

* All human faces share some similarities. If you look at a photograph showing a person’s face, you will see, for example, that the eye region is darker than the bridge of the nose. The cheeks are also brighter than the eye region. We can use these properties to help us understand if an image contains a human face.
* The value of the feature is calculated as a single number: the sum of pixel values in the black area minus the sum of pixel values in the white area. For uniform areas like a wall, this number would be close to zero and won’t give you any meaningful information.
* <https://realpython.com/traditional-face-detection-python/>
* Simple solution: plus all value pixels in black area and white area.
* But this takes a lot of time using the limited resources of a computer.
* To tackle this problem, Viola and Jones used integral images.



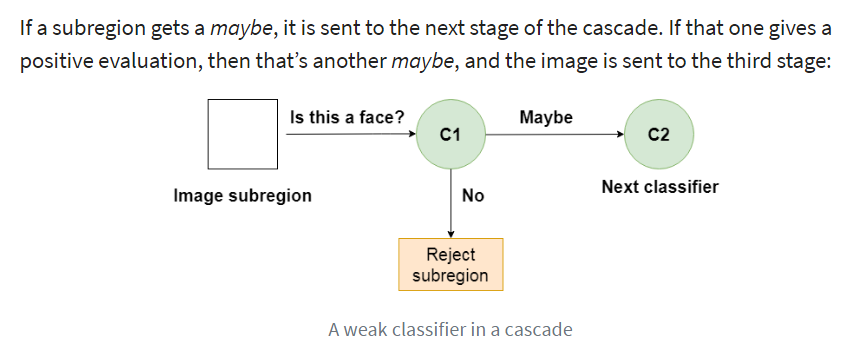


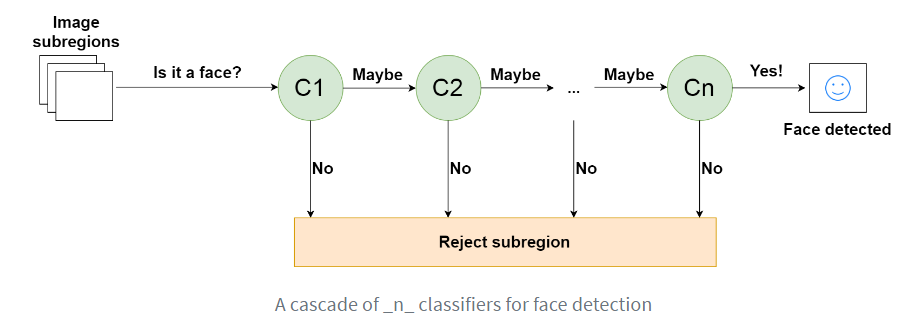


Now you have a simple way to calculate the difference between the sums of pixel values of two rectangles. This is perfect for Haar-like features!

1. AdaBoost

* To calculate the performance of a classifier, you evaluate it on all subregions of all the images used for training. Some subregions will produce a strong response in the classifier. Those will be classified as positives, meaning the classifier thinks it contains a human face.
* To solve it, Viola and Jones turned their strong classifier (consisting of thousands of weak classifiers) into a cascade where each weak classifier represents one stage. The job of the cascade is to quickly discard non-faces and avoid wasting precious time and computations.





* This is designed so that non-faces get discarded very quickly, which saves a lot of time and computational resources.  Since every classifier represents a feature of a human face, a positive detection basically says, “Yes, this subregion contains all the features of a human face.” But as soon as one feature is missing, it rejects the whole subregion.
* To accomplish this effectively, it is important to put your best performing classifiers early in the cascade. In the Viola-Jones algorithm, the eyes and nose bridge classifiers are examples of best performing weak classifiers.

1. Using a Viola-Jone Classifier

* Training a Viola-Jones classifier from scratch can take a long time. Fortunately, a pre-trained Viola-Jones classifier comes out-of-the-box with OpenCV! You will use that one to see the algorithm in action.