**HTS\_Assignment\_05**

**i. Text Summarization in Machine Learning:**

Text summarization refers to the technique of shortening long pieces of text. The intention is to create a coherent and fluent summary having only the main points outlined in the document.

Text summarization is the problem of creating a short, accurate, and fluent summary of a longer text document.

Automatic text summarization is a common problem in machine learning and natural language processing (NLP).

**TYPES OF NEURAL TEXT SUMMARIZATION:**

There are two main approaches to summarizing text documents; they are:

1. Extractive Methods.

2. Abstractive Methods.

*The different dimensions of text summarization can be generally categorized based on its input type (single or multi document), purpose (generic, domain specific, or query-based) and output type (extractive or abstractive).*

— A Review on Automatic Text Summarization Approaches, 2016.

Extractive text summarization involves the selection of phrases and sentences from the source document to make up the new summary. Techniques involve ranking the relevance of phrases in order to choose only those most relevant to the meaning of the source.

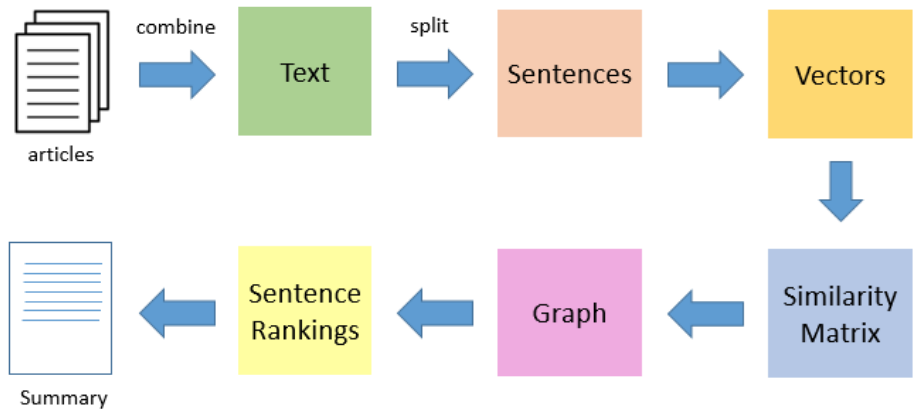
Abstractive text summarization involves generating entirely new phrases and sentences to capture the meaning of the source document. This is a more challenging approach, but is also the approach ultimately used by humans. Classical methods operate by selecting and compressing content from the source document.

*… there are two different approaches for automatic summarization: extraction and abstraction. Extractive summarization methods work by identifying important sections of the text and generating them verbatim; […] abstractive summarization methods aim at producing important material in a new way. In other words, they interpret and examine the text using advanced natural language techniques in order to generate a new shorter text that conveys the most critical information from the original text*

— Text Summarization Techniques: A Brief Survey, 2017.

Classically, most successful text summarization methods are extractive because it is an easier approach, but abstractive approaches hold the hope of more general solutions to the problem.

**TEXT SUMMARIZATION IN NLP:**



**ABSTRACTIVE APPROACHES:**

An abstractive approach is more advanced. On the basis of time requirements we exchange some sentences for smaller sentences with the same semantic approaches of our text data.

Here we generally use deep machine learning, that is **transformers, bi-directional transformers(BERT), GPT**, etc.

**EXTRACTIVE APPROACHES:**

Using an extractive approach we summarize our text on the basis of simple and traditional algorithms. For example, when we want to summarize our text on the basis of the frequency method, we store all the important words and frequency of all those words in the dictionary. On the basis of high frequency words, we store the sentences containing that word in our final summary. This means the words which are in our summary confirm that they are part of the given text.

TEXT SUMMARIZATION USING THE FREQUENCY METHOD

USING A PRE-TRAINED SUMMARIZER AND EVALUATING ITS OUTPUT

<https://www.analyticsvidhya.com/blog/2021/11/a-beginners-guide-to-understanding-text-summarization-with-nlp/>

What do we mean by pre-trained models:- These models have already been trained on large datasets. If a model is trained on huge amounts of data it will naturally predict better, however, the inability to collect large amounts of data and subsequently higher training time are some of the reasons why instead of training a model from scratch we could benefit by using a pre-trained model.

We would be using the [BBC News Summary dataset](https://www.kaggle.com/pariza/bbc-news-summary) for this article and [bert-extractive-summarizer](https://github.com/dmmiller612/bert-extractive-summarizer" \t "_blank) as the pre-trained model.

Below code, snippet includes loading the necessary libraries

!pip install bert-extractive-summarizer

!pip install spacy

!pip install transformers # > 4.0.0

!pip install neuralcoref

!python -m spacy download en\_core\_web\_md

After importing the above libraries and downloading the spacy model we would now call the summarizer and pass a sample text to view its output.

#from summarizer import Summarizer

model = Summarizer()

text = "Learning NLP involves understanding basic principles of machine learning which then need to be customized for words. With the advent of using transfer learning for NLP I think it hads made a huge progress in terms of its research"

As you can see in the below output the model does provide an appropriate summary given our input text.

Now let us use the same model on our BBC news dataset, the below snippet takes care of the same. As we have a total of 2225 input articles with an average length of 3000 words, to save execution time I have predicted the summary items only for the first 10 input articles.

from tqdm import tqdm

bert\_predicted\_summary = []

k = 0

for i in tqdm(df['text']):

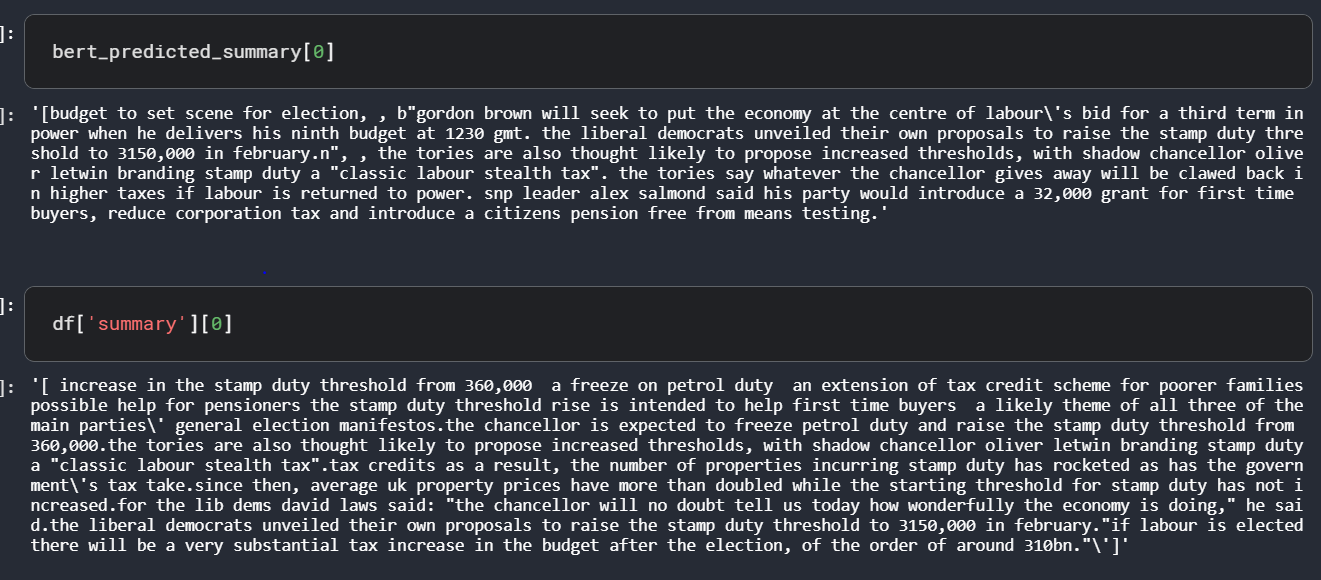
if k < 10:

x = model(str(i))

bert\_predicted\_summary.append(x)

k+=1

Below is the attached output, the first one is what the pre-trained model predicted and the second one is the actual summary provided in the dataset.



Using simple preprocessing techniques like removing newline characters(n) or end of sentence characters(b) is always recommended. As the popular saying goes garbage in is garbage out, so we need to clean our input before passing it to our model. I have used simple regular expressions for preprocessing the input, the code snippet for the same is as below.

path = '/kaggle/input/bbc-news-summary/bbc news summary/BBC News Summary/News Articles/'

for i in os.listdir(path):

for j in os.listdir(os.path.join(path+i)):

with open(os.path.join(path+i+'/'+j),'rb') as f:

article = f.readlines()

article = re.sub('b'','',str(article))

article = re.sub('[\nnt-\/]','',article)

article = re.sub('n'','',article)

article = re.sub('xc2xa','',article)

article = article.lower()

text.append(article)

type\_.append(i)

<https://www.assemblyai.com/blog/text-summarization-nlp-5-best-apis/>

In Natural Language Processing (NLP), Text Summarization models automatically shorten documents, papers, podcasts, videos, and more into their most important soundbites. The models are powered by advanced Deep Learning and Machine Learning research.

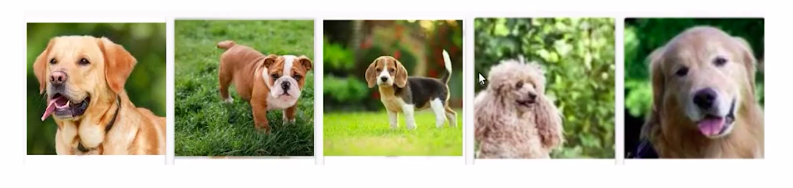
Product teams are integrating Text Summarization APIs and AI Summarization models into their AI-powered platforms to create summarization tools that automatically summarize calls, interviews, law documents, and more. These are sometimes referred to as AI summarizers.

**ii. Object Detection**

<https://www.analyticsvidhya.com/blog/2022/03/a-basic-introduction-to-object-detection/#h-training-data-for-object-detection>

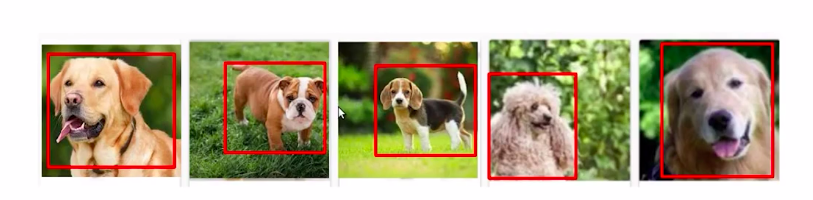
Object detection is a computer vision technique for locating instances of objects in images or videos. Humans can easily detect and identify objects present in an image.

Object Detection models allow users to identify objects of certain defined classes. Object detection models receive an image as input and output the images with bounding boxes and labels on detected objects.



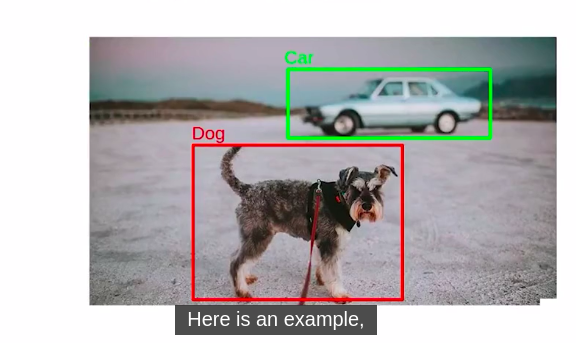
So instead of classifying, which type of dog is present in these images, we have to actually locate a dog in the image.  That is, I have to find out where is the dog present in the image? Is it at the center or at the bottom left? And so on. Now the next question comes into the human mind, how can we do that? So let’s start.

Well, we can create a box around the dog that is present in the image and specify the x and y coordinates of this box.



For now, consider that the location of the object in the image can be represented as coordinates of these boxes. So this box around the object in the image is formally known as a bounding box. Now, this becomes an ***image localization*** problem where we are given a set of images and we have to identify where is the object present in the image.

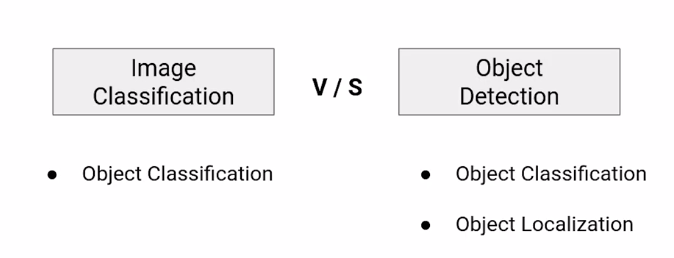
Note that here we have a single class. **what if we have multiple classes?** here is an example,



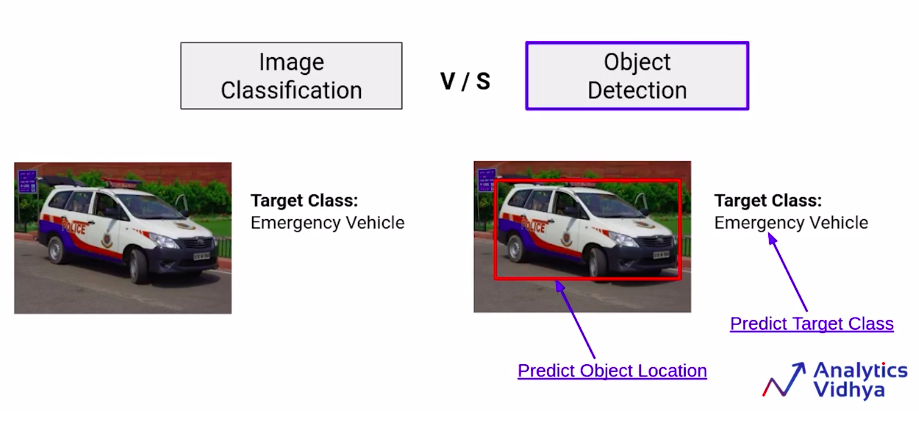
In this image, we have to locate the objects in the image but note that all the objects are not dogs. Here we have a dog and a car. So we not only have to locate the objects in the image but also classify the located object as a dog or Car. So this becomes an ***object detection problem***.

We will also discuss **image classification v/s object detection**.

In the case of ***object detection*** problems, we have to classify the objects in the image and also locate where these objects are present in the image. But the ***image classification*** problem had only one task where we had to classify the objects in the image.



So for the one example, the image is given below that we have been working on in the first case, we predict only the target class, and such tasks are known as ***image classification*** problems. While in the second case, along with predicting the target class, we also have to find the bounding box which denotes the location of the object. This is all



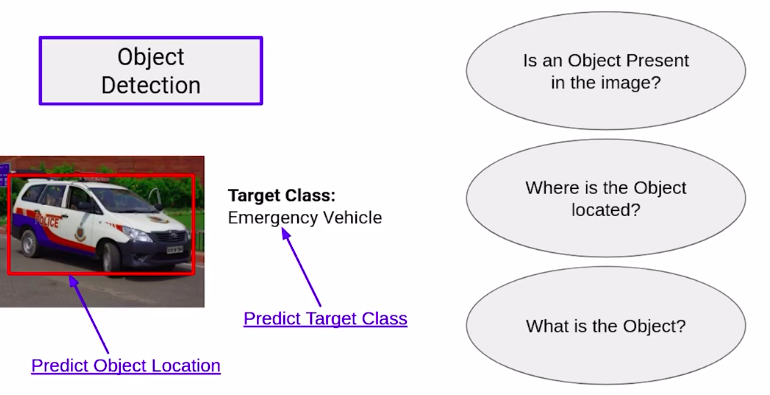
This is all about the ***object detection*** problem. So broadly we have ***three tasks for object detection problems:***

1. To identify if there is an object present in the image,

2. where is this object located,

3. what is this object?

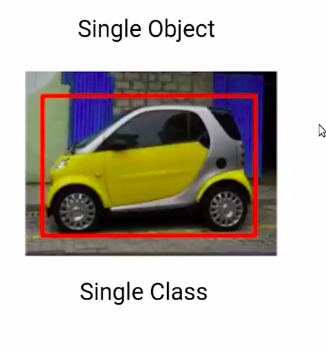
So we can see the below image.



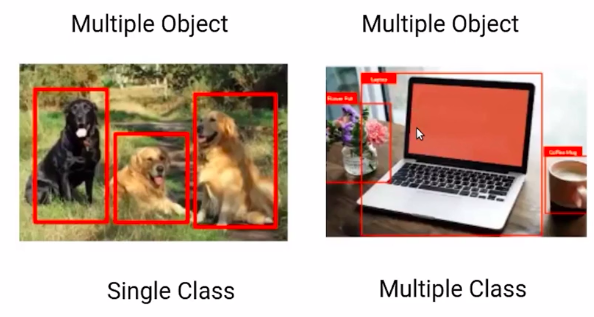
Specific to this example, we have an object in the image. We can create a bounding box around the object and this object is an emergency vehicle.

Now the object detection problem can also be divided into multiple categories.

First is the case when we have images that have only one object. That is we can have 1000 images in the data set, and all of these images will have only one object. And if all these objects belong to a single class, that is all the objects are cars, then this will be an ***image localization problem***. That is we already know what class these objects belong to, we only have to locate where these objects are present in the image.



Another problem could be where we are provided with multiple images, and within each of these images, we have multiple objects. Also, these objects can be of the same class, or another problem can be that these objects are of different classes.



So in case we have multiple objects in the image and all of the objects are of different classes. We would have to not only locate the objects but also classify these objects.

**Object Recognition with Deep Learning**

<https://machinelearningmastery.com/object-recognition-with-deep-learning/>

Image classification involves assigning a class label to an image, whereas object localization involves drawing a bounding box around one or more objects in an image. Object detection is more challenging and combines these two tasks and draws a bounding box around each object of interest in the image and assigns them a class label. Together, all of these problems are referred to as object recognition.

* Object recognition is refers to a collection of related tasks for identifying objects in digital photographs.
* Region-Based Convolutional Neural Networks, or R-CNNs, are a family of techniques for addressing object localization and recognition tasks, designed for model performance.
* You Only Look Once, or YOLO, is a second family of techniques for object recognition designed for speed and real-time use.

**What is Object Recognition?**

Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects in digital photographs.

***Image classification*** involves predicting the class of one object in an image. ***Object localization*** refers to identifying the location of one or more objects in an image and drawing abounding box around their extent. ***Object detection*** combines these two tasks and localizes and classifies one or more objects in an image.

When a user or practitioner refers to ***“object recognition”***, they often mean ***“object detection”***.

*… we will be using the term object recognition broadly to encompass both image classification (a task requiring an algorithm to determine what object classes are present in the image) as well as object detection (a task requiring an algorithm to localize all objects present in the image*

— ImageNet Large Scale Visual Recognition Challenge, 2015.

As such, we can distinguish between these three computer vision tasks:

**Image Classification:** Predict the type or class of an object in an image.

Input: An image with a single object, such as a photograph.

Output: A class label (e.g. one or more integers that are mapped to class labels).

**Object Localization:** Locate the presence of objects in an image and indicate their location with a bounding box.

Input: An image with one or more objects, such as a photograph.

Output: One or more bounding boxes (e.g. defined by a point, width, and height).

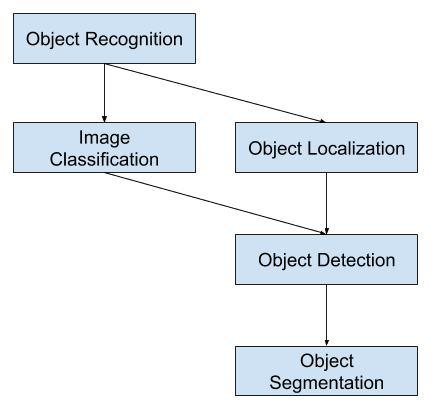
**Object Detection:** Locate the presence of objects with a bounding box and types or classes of the located objects in an image.

Input: An image with one or more objects, such as a photograph.

Output: One or more bounding boxes (e.g. defined by a point, width, and height), and a class label for each bounding box.

One further extension to this breakdown of computer vision tasks is ***object segmentation***, also called “object instance segmentation” or “semantic segmentation,” where instances of recognized objects are indicated by highlighting the specific pixels of the object instead of a coarse bounding box.

From this breakdown, we can see that object recognition refers to a suite of challenging computer vision tasks.



**iii. Image Segmentation**

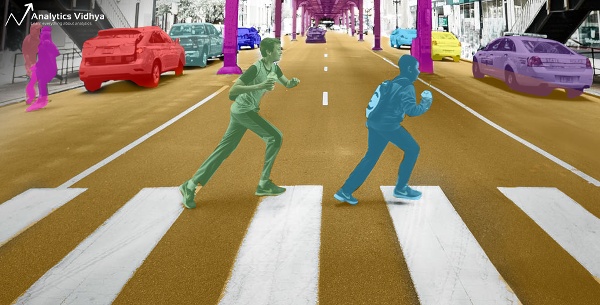
<https://www.analyticsvidhya.com/blog/2019/04/introduction-image-segmentation-techniques-python/>

<https://www.analyticsvidhya.com/blog/2019/07/computer-vision-implementing-mask-r-cnn-image-segmentation/?utm_source=blog&utm_medium=introduction-image-segmentation-techniques-python>

**Introduction**

What’s the first thing you do when you’re attempting to cross the road? We typically look left and right, take stock of the vehicles on the road, and make our decision. Our brain is able to analyze, in a matter of milliseconds, what kind of vehicle (car, bus, truck, auto, etc.) is coming towards us. Can machines do that?

The answer was an emphatic ‘no’ till a few years back. But the rise and advancements in [computer vision](https://courses.analyticsvidhya.com/courses/computer-vision-using-deep-learning-version2?utm_source=blog&utm_medium=image-segmentation-article) have changed the game. We are able to build computer vision models that can detect objects, determine their shape, predict the direction the objects will go in, and many other things. You might have guessed it – that’s the powerful technology behind self-driving cars!

[](https://cdn.analyticsvidhya.com/wp-content/uploads/2019/03/image-segmentation.png)

Now, there are multiple ways of dealing with computer vision challenges. The most popular approach I have come across is based on identifying the objects present in an image, aka, object detection. But what if we want to dive deeper? What if just detecting objects isn’t enough – we want to analyze our image at a much more granular level?

Here, I will introduce you to the concept of ***image segmentation***. It is a powerful **computer vision algorithm** that builds upon the idea of object detection and takes us to a whole new level of working with image data. This technique opens up so many possibilities

**What Is Image Segmentation?**

Let’s understand image segmentation using a simple example. Consider the below image:

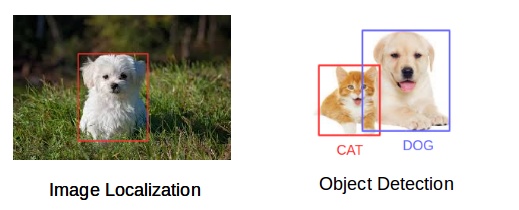


There’s only one object here – a dog. We can build a straightforward cat-dog classifier model and predict that there’s a dog in the given image. But what if we have both a cat and a dog in a single image?



We can train a multi-label classifier, in that instance. Now, there’s another caveat – we won’t know the location of either animal/object in the image.

That’s where ***image localization*** comes into the picture. It helps us to identify the location of a single object in the given image. In case we have multiple objects present, we then rely on the concept of [***object detection***](https://www.analyticsvidhya.com/blog/2018/10/a-step-by-step-introduction-to-the-basic-object-detection-algorithms-part-1/?utm_source=blog&utm_medium=introduction-image-segmentation-techniques-python)***(OD)***. We can predict the location along with the class for each object using OD.

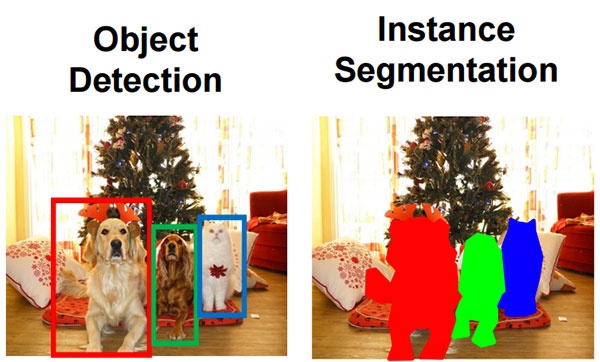


Before detecting the objects and even before classifying the image, we need to understand what the image consists of. Enter – Image Segmentation.

**How Does Image Segmentation Work?**

We can divide or partition the image into various parts called segments. It’s not a great idea to process the entire image at the same time as there will be regions in the image which do not contain any information. By dividing the image into segments, we can make use of the important segments for processing the image. That, in a nutshell, is how image segmentation works.

An image is a collection or set of different pixels. We group together the pixels that have similar attributes using image segmentation.



Object detection builds a bounding box corresponding to each class in the image. But it tells us nothing about the shape of the object. We only get the set of bounding box coordinates. We want to get more information – this is too vague for our purposes.

Image segmentation creates a pixel-wise mask for each object in the image. This technique gives us a far more granular understanding of the object(s) in the image.

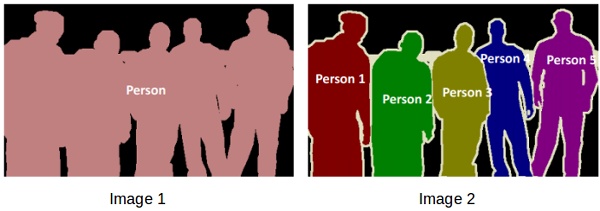
**What Is Image Segmentation Used For?**

There are many applications where Image segmentation is transforming industries:

* Healthcare
* Traffic Control Systems
* Self Driving Cars
* Locating objects in satellite images

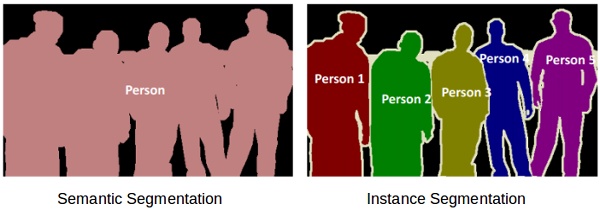
**Different Types of Image Segmentation**

We can broadly divide image segmentation techniques into two types. Consider the below images:



Both the images are using image segmentation to identify and locate the people present.

* In image 1, every pixel belongs to a particular class (either background or person). Also, all the pixels belonging to a particular class are represented by the same color (background as black and person as pink). This is an example of ***semantic segmentation***.
* Image 2 has also assigned a particular class to each pixel of the image. However, different objects of the same class have different colors (Person 1 as red, Person 2 as green, background as black, etc.). This is an example of ***instance segmentation***.



If there are 5 people in an image, semantic segmentation will focus on classifying all the people as a single instance. Instance segmentation, on the other hand, will identify each of these people individually.

**Approaches for image segmentation:**

Region-based Segmentation

Edge Detection Segmentation

Clustering-based Image Segmentation

Mask R-CNN

**iv. Image Classification**

<https://www.superannotate.com/blog/image-classification-basics>

Image classification is a supervised learning problem: define a set of target classes (objects to identify in images), and train a model to recognize them using labeled example photos. Early computer vision models relied on raw pixel data as the input to the model.

**Types of image classification**

Depending on the problem at hand, there are different types of image classification methodologies to be employed. These are binary, multiclass, multilabel, and hierarchical.

**‍Binary:** Binary classification takes an either-or logic to label images, and classifies unknown data points into two categories. When your task is to categorize benign or malignant tumors, analyze product quality to find out whether it has defects or not, and many other problems that require yes/no answers are solved with binary classification.‍

**Multiclass:** While binary classification is used to distinguish between two classes of objects, multiclass, as the name suggests, categorizes items into three or more classes. It's very useful in many domains like NLP (sentiment analysis where more than two emotions are present), medical diagnosis(classifying diseases into different categories), etc.‍

**Multilabel:** Unlike multiclass classification, where each image is assigned to exactly one class, multilabel classification allows the item to be assigned to multiple labels. For example, you may need to classify image colors and there are several colors. A picture of a fruit salad will have red, orange, yellow, purple, and other colors depending on your creativity with fruit salads. As a result, one image will have multiple colors as labels.‍

**Hierarchical:** Hierarchical classification is the task of organizing classes into a hierarchical structure based on their similarities, where a higher-level class represents broader categories and a lower-level class is more concrete and specific.

**Image classification vs. object detection**

Image classification, object detection, object localization are essential components of computer vision and image annotation, each with its own distinct nuances.

Image classification refers to assigning a specific label to the entire image. On the other hand, object localization goes beyond classification and focuses on precisely identifying and localizing the main object or regions of interest in an image. By drawing bounding boxes around these objects, object localization provides detailed spatial information, allowing for more specific analysis.



Object detection on the other hand is the method of locating items within and image assigning labels to them, as opposed to image classification, which assigns a label to the entire picture. As the name implies, object detection recognizes the target items inside an image, labels them, and specifies their position. One of the most prominent tools to perform object detection is the “bounding box” which is used to indicate where a particular object is located on an image and what the label of that object is. Essentially, object detection combines image classification and object localization.

Overall, the image classification pipeline looks something like this:

***Image pre-processing -> feature extraction -> object classification***