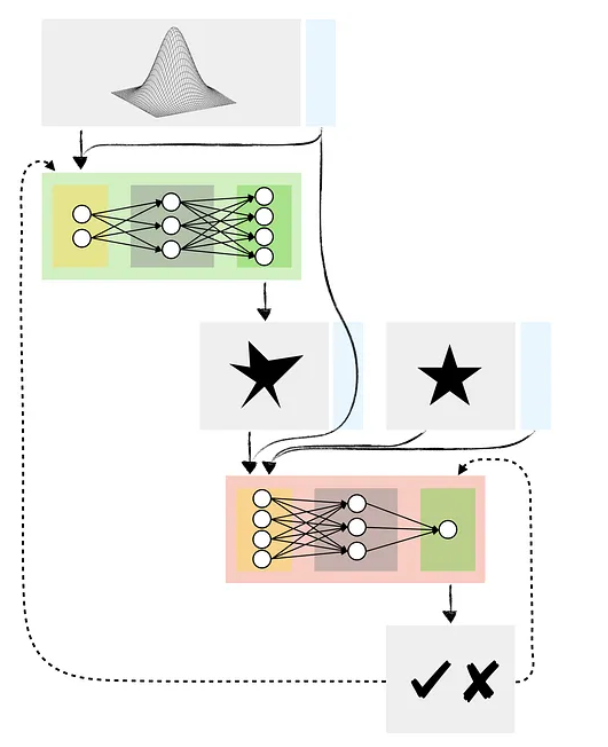
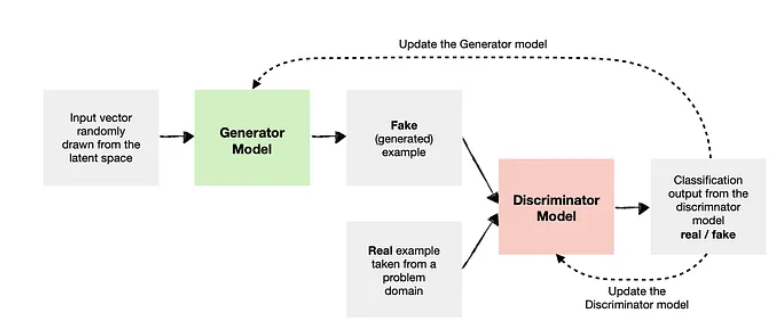
**cGAN: Conditional Generative Adversarial Network — How to Gain Control Over GAN Outputs**



Have you experimented with Generative Adversarial Networks (GANs) yet? If so, you may have encountered a situation where you wanted your GAN to generate a **specific type of data** but did not have sufficient control over GANs outputs.

For example, assume you used a broad spectrum of flower images to train a GAN capable of producing fake pictures of flowers. While you can use your model to generate an image of a random flower, you cannot instruct it to create an image of, say, a tulip or a sunflower.

Conditional GAN (cGAN) allows us to condition the network with additional information such as class labels. It means that during the training, we pass images to the network with their actual labels (rose, tulip, sunflower etc.) for it to learn the difference between them. That way, we gain the ability to ask our model to generate images of specific flowers.

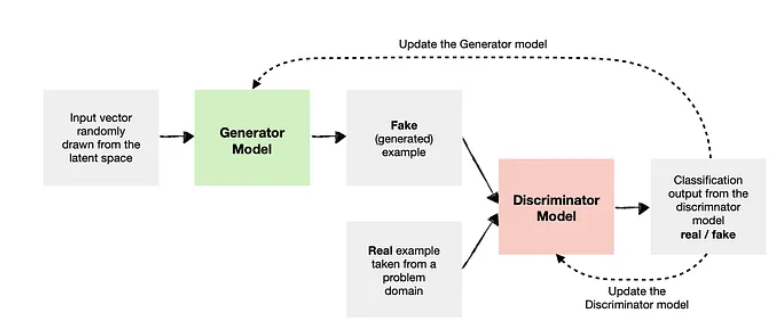


As you can see, we have two main components:

* **Generator Model**— generates new data (i.e., fake data) similar to that of the problem domain.
* **Discriminator Model** — tries to identify whether the provided example is fake (comes from a generator ) or real (comes from the actual data domain).

In the case of a Conditional GAN, **we want to condition both** the Generator and the Discriminator so they know which type they are dealing with.

For example: Say we use our GAN to create synthetic data containing house prices in London and Madrid. To make it conditional, we need to tell the Generator which city to generate the data for each time. We also need to inform the Discriminator whether the example passed to it is for London or Madrid.



Note that we can condition GANs on many types of inputs. For example, we could also condition the network on other images where we want to create a GAN for image-to-image translation (e.g., turning the daytime image into a nighttime one).

## **Conditional Deep Convolutional GAN (cDCGAN)**

As with the earlier flower example, we may want to condition a [Deep Convolution GAN](https://towardsdatascience.com/deep-convolutional-gan-how-to-use-a-dcgan-to-generate-images-in-python-b08afd4d124e) so we can ask the model to generate a specific type of image.

Below is a model architecture diagram for a Conditional DCGAN. Note that the high-level architecture is essentially the same as in the previous example, except the Generator and Discriminator contain additional layers, such as [Convolutions](https://towardsdatascience.com/convolutional-neural-networks-explained-how-to-successfully-classify-images-in-python-df829d4ba761) and [Transposed Convolutions](https://towardsdatascience.com/transposed-convolutional-neural-networks-how-to-increase-the-resolution-of-your-image-d1ec27700c6a).

