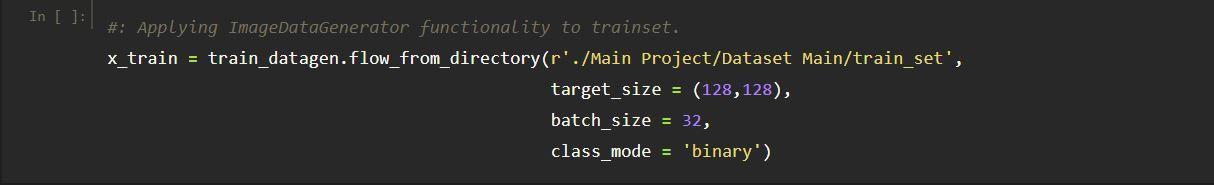
**APPLYING IMAGE DATA GENERATOR FUNCTIONALITY TO TRAINSET AND TESTSET**

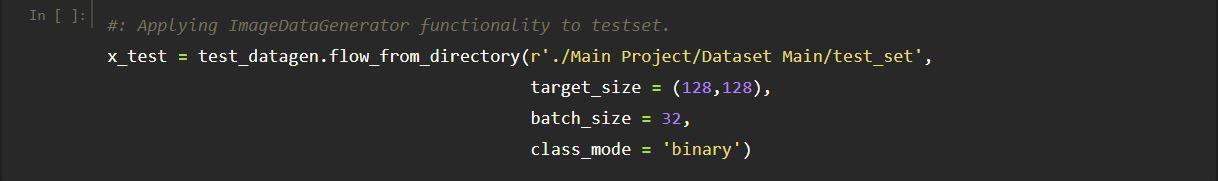
The ImageDataGenerator class has three methods flow ( ), flow\_from\_directory ( ), and flow\_from\_dataframe ( ) to read the images from a big numpy array and folders containing images.

flow\_from\_directory ( ) expects at least one directory under the given directory path.

**Task1:**Applyflow\_from\_directory ( ) method for Train folder.



**Task 2:**Now will apply the flow\_from\_directory ( ) method for test folder.



* The directory must be set to the path where your training folders are present.
* The target\_size is the size of your input images, every image will be resized to this size.
* batch\_size: No. of images to be yielded from the generator per batch.
* “batch\_size” in both train and test generators is to some number that divides your total number of images in your train set and train set respectively.
* class\_mode: Set “binary” if you have only two classes to predict, if not set to “categorical”.

Keras has DataGenerator classes available for different data types. However as I mentioned earlier, this post will be about images and for this data [ImageDataGenerator](https://keras.io/api/preprocessing/image/#imagedatagenerator-class) is the corresponding class.

I will be explaining the process using code because I believe that this would lead to a better understanding. There are six aspects that I would be covering.

1. Creating a data generator
2. Few useful data generator properties
3. Visualizing data generator tensors for a quick correctness test
4. Training using the data generator
5. Predicting using the data generator
6. Training, validation and test set creation

* ImageDataGenerator class allows you to randomly rotate images through any degree between 0 and 360 by providing an integer value in the rotation\_range argument.
* Keras provides a data generator for image datasets. This is available in tf.keras.preprocessing.image as ImageDataGenerator class.
* The advantage of using ImageDataGenerator is that it will generate batches of data with augmentation.
* To load images from a local directory, use image\_dataset\_from\_directory() method to convert the directory to a valid dataset to be used by a deep learning model. image\_size and batch\_size parameters specify the size of an image and the number of dataset batches respectively.