**ARTIFICIAL INTELLIGENCE**

**Natural Disasters Intensity Analysis & Classification Using AI**

|  |  |
| --- | --- |
| Date | 15-10-2022 |
| Team Id | PNT20222TMID45294 |
| Project Name | Natural Disaters Intensity Analysis And Classification Using AI |

**Apply Image Data Generator Functionality To Trainset And Testset:-**

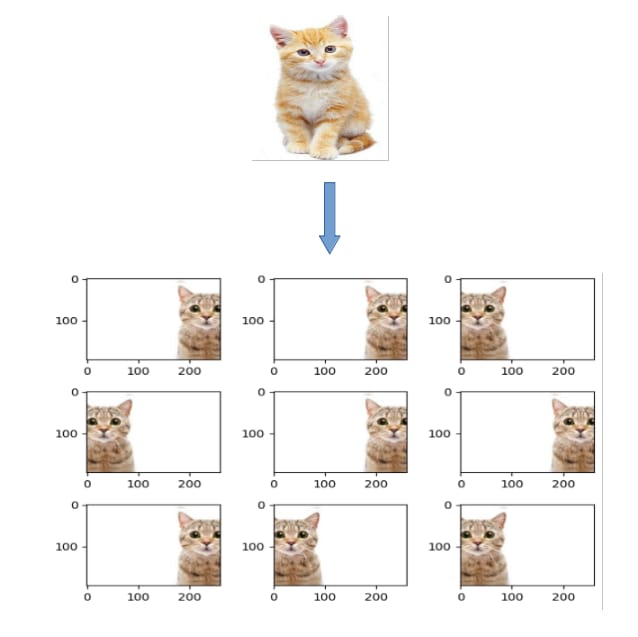
**Introduction:-**

**\*Keras Image DataGenerator is a gem! It lets you augment your images in real-time while your model is still training!**

**“A generator function is defined like a normal function, but whenever it needs to generate a value, it does so with the yield keyword rather than return.**

**\*If the body of a def contains yield, the function automatically becomes a generator function.”**

**“Generator are functions that yield result, suspend the operation and then resume the operation.”**

****

**Lets highlight some more points :-**

**1. Return sends a specified value back to its caller whereas Yield can produce a sequence of values.**

**2. Return basically exits from a function with a value and the local values within it is destroyed whereas yield is sort of exiting from the function but it remembers the state when it exits. When you do next from your function, it will resume from the point where it yielded.**

**3.The yield statement suspends function’s execution and sends a value back to caller, but retains enough state to enable function to resume where it is left off. When resumed, the function continues execution immediately after the last yield run. This allows its code to produce a series of values over time, rather them computing them all at once and sending them back like a list.**

**4. We should use yield when we want to iterate over a sequence, but don’t want to store the entire sequence in memory.**

**About Train Validation And Test Sets in AI:-**

****

**This is aimed to be a short primer for anyone who needs to know the difference between the various dataset splits while training Machine Learning models.**

**Training Data Set :-**

**\*The sample of data used to fit the model.**

**\*The actual dataset that we use to train the model (weights and biases in the case of a Neural Network). The model sees and learns from this data.**

**Test Data Set:-**

**\*The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.**

****

**A Visualization Of The Split**

**How To Split Data Set Into Train And Test Sets:-**

**This mainly depends on 2 things:**

**First, the total number of samples in your data and**

**second, on the actual model you are training.**

**\*Some models need substantial data to train upon, so in this case you would optimize for the larger training sets. Models with very few hyperparameters will be easy to validate and tune, so you can probably reduce the size of your validation set, but if your model has many hyperparameters, you would want to have a large validation set as well(although you should also consider cross validation). Also, if you happen to have a model with no hyperparameters or ones that cannot be easily tuned, you probably don’t need a validation set too!**

**\*All in all, like many other things in machine learning, the train-test-validation split ratio is also quite specific to your use case and it gets easier to make judge ment as you train and build more and more models.**

**Note on Cross Validation: -**

**\*Many a times, people first split their dataset into 2 — Train and Test. After this, they keep aside the Test set, and randomly choose X% of their Train dataset to be the actual Train set and the remaining (100-X)% to be the Validation set, where X is a fixed number(say 80%), the model is then iteratively trained and validated on these different sets. There are multiple ways to do this, and is commonly known as Cross Validation. Basically you use your training set to generate multiple splits of the Train and Validation sets. Cross validation avoids over fitting and is getting more and more popular, with K-fold Cross Validation being the most popular method of cross validation. Check this out for more.**