[1]: Tool: ChatGPT
Prompt: I would like to seed random data as I split an unbalanced dataset to contain the same number of entries for both classes. What Python libraries can I use?
Response: You can use Python's built-in random module to randomly sample or shuffle your data and ensure reproducibility by setting a seed. Here's how you can do it for balancing an unbalanced dataset:

- random.seed(42) ensures the same random selection every run.
- random.sample(population, k) selects k unique elements without replacement.
- random.shuffle(list) shuffles a list in-place.

[2]: Tool: ChatGPT
Prompt: I'm having an issue where I do not want to use too many epochs to train, and want to stop early once the validation accuracy begins to decrease. I know the learning rate is also influential in establishing when the stopping point is. Define 2 callback functions for a TF env that I can put as I train each model.
Response: Got it! You want two callbacks for TensorFlow/Keras: one to stop training early when validation accuracy stops improving, and another to adjust the learning rate when progress stalls. Here's a clean setup: EarlyStopping: stops training once val_accuracy stops improving for 5 epochs, preventing overfitting.

- ReduceLROnPlateau: lowers the learning rate by a factor of 0.5 if val_accuracy doesn't improve for 3 epochs, helping the model converge more finely.

This combination is very common in TF workflows because learning rate and early stopping work together: LR reductions can prevent the model from "getting stuck" before EarlyStopping halts training.

[3]: Tool: ChatGPT
Prompt: I need a quick way to flatten a 1D array, but I do not want to make a copy. There are already a lot of memory constraints on this Virtual Machine.
Response: In Python, .ravel() is a NumPy array method that flattens an array into a 1-dimensional array without making a copy if possible. It's very commonly used when you need a flat vector instead of a multi-dimensional array, for example in machine learning targets or reshaping images. If you need a copy, use flatten() instead (arr.flatten()), though I recommend trying .ravel() first.