I have 10k rows and 14 columns, out of which 11 are numerical and 3 are categorical(Surname, Geography, Gender)

There were no duplicate values.

I dropped RowNumber, CustomerID and Surname as they will be irrelevant in making my predictions.

Did OHE for Geography and Gender which were the remaining cat6egorical columns.

callback = EarlyStopping(

    monitor="val\_loss",

    min\_delta=0.00001,

    patience=20,

    verbose=1,

    mode="auto",

    baseline=None,

    restore\_best\_weights=False

)

explain each

This code defines an EarlyStopping callback function in Keras, which helps prevent overfitting in your neural network training. Here's a breakdown of each argument:

**monitor="val\_loss"**

* This argument specifies the metric to monitor during training. Here, it's set to "val\_loss", indicating the validation loss. The EarlyStopping callback will track the validation loss after each epoch.

**min\_delta=0.00001**

* This argument defines the minimum amount of improvement the validation loss needs to show before it's considered a real change. Here, it's set to a very small value, 0.00001. This means the validation loss needs to decrease by at least 0.00001 to be considered an improvement. Values close to zero allow for very small improvements to trigger stopping.

**patience=20**

* This argument defines the number of epochs to wait for an improvement in the monitored metric (validation loss in this case) before stopping training. Here, it's set to 20. This means if the validation loss doesn't improve (decrease by at least min\_delta) for 20 consecutive epochs, the training will be stopped.

**verbose=1**

* This argument controls how much information the callback prints during training. Here, it's set to 1, which means the callback will print a message when training is stopped. Other options include 0 for no messages and 2 for more verbose output.

**mode="auto"**

* This argument specifies the direction of change in the monitored metric that triggers early stopping. Here, it's set to "auto", which means the callback will automatically infer the direction based on the name of the metric. Since it's monitoring validation loss (generally minimized), it will stop training when the validation loss stops decreasing. You could also set it to "min" (stop on minimum) or "max" (stop on maximum) for other metrics.

**baseline=None**

* This argument allows you to set a baseline value for the monitored metric. Training would stop if the metric gets worse (higher for loss) than this baseline value, regardless of improvement. Here, it's set to None, meaning no baseline is used.

**restore\_best\_weights=False**

* This argument controls whether to restore the model weights from the epoch with the best performance on the monitored metric. Here, it's set to False. This means the model weights at the stopping point won't be restored. You might set it to True if you want to use the model with the best validation loss even though training stopped early.