Question:
learning_rate: learning rate to be used for all updates, defaults to 0.001
momentum: momentum/gamma values for momentum based gradient descent, defaults to 0.5
num_hidden: number of hidden layers
sizes: comma separated values, number of units in each hidden layer
activation: non-linearity to be used after each layer, defaults to relu
loss: loss function to optimize, defaults to "ce" i.e cross entropy
opt: optimizer to be used for learning, defaults to "adam". Could be any one of "gd", "momentum", "adam", "nag"
batch_size: size of minibatch, default is 20
num_epochs: number of epochs to run for, defaults to 10
anneal: a boolean argument. If set to "true", validation loss of the current epoch is compared with the previous epoch and if found to be greater than the latter, the epoch is restarted and any weight updates during the epoch are discareded.
path_save_dir: path to the folder where the final model is stored
expt_dir: path to the folder where the logs are stored
path_to_train: path to the training data .csv file
path_to_val: path to the validation dataset .csv file
path_to_test: path to the test dataset .csv file
to_pretrain: boolean variable, if set to "true", weigths from the epoch determined by state from a previous training session are loaded
epoch_to_restore: epoch from which weights need to be loaded
to_test: boolean variable, if set to "true", weights determined by "state" are loaded and predictiosn produced for the test data

Notes:
Used Xavier initialization for weights so that the variance of the outputs of each layer is equal to the variance of the inputs it helps to prevent the gradients from vanishing or exploding during training.
Implemented the following optimizers:

PROF

1. Gradient Descent
2. Momentum
3. Adam
4. Nesterov Accelerated Gradient

Took the data from scikit-learn datasets