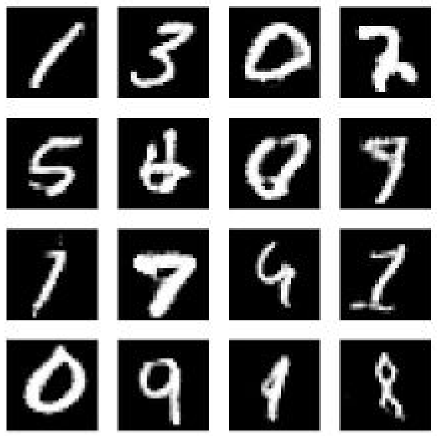
TASK - GANS, and RL FOR STOCK PREDICTION

Hey guys, now this is a slightly complicated problem statement when compared to the others. Hence, a slightly more challenging task as well :P. As the name suggests, we will try to make a simple GAN model for generating DIGITS from the MNIST Dataset.

So how is my data going to look like, basically the **MNIST Digit Dataset** composed of handwritten digits.



Simple GAN output for the MNIST DIGIT DATASET

A basic outline of how to approach the problem.

- So what is a GAN - Generative Adversarial Networks? **It is an architecture comprising of two networks pitting against each other (adversarial) to generate synthetic datasets that can pass for real data.** Now why GANS? Often we tend to not have a lot of data that can be used to train a model to generate images. **Hence the GAN network works on the principle that one of the models is trained to differentiate between real and fake data, and the other model is trained to fool this "discriminator" by "generating" synthetic data that cannot be differentiated from real data.**

- Now the task here is to implement a simple GAN architecture. So as you already know a simple GAN consists of a Discriminator and a Generator. **Initially, the Discriminator is trained to classify between real images and unreal images. Next, the generator model is created in such a way that the output of the generator (synthesized images) is passed onto the discriminator. In a simple GAN model, the input is going to be random numbers of a specific size. (These can be a gaussian distribution, etc).** Right now you don't need to worry about what these inputs mean and it actually doesn't matter for the model as it is just trying to generate realistic images.

- **The simplest way to train the generator is to connect it with the discriminator. This way the weights of the generator are trained based on the performance of the discriminator. Hence the better the discriminator gets, the better the generator tries to fool the model by synthesizing real images.**

- **In every iteration, we first start by training the discriminator to classify real and fake images better. Next, we fix the weights of the discriminator and train the generator by giving it the randomly created data as input and force the discriminator to predict it as 1 [real data]. Since the discriminator was already trained to distinguish between real and fake data, it would output very low probabilities for being real. Hence the error is high and the weights are updated. Then the discriminator model is set to be trainable once again and retrained to distinguish between real and fake data and the process continues. After a sufficient number of iterations, the discriminator can no longer realize the difference between the synthesized data and real data.**

- I know this is very difficult to process and hence please try to refer to some articles as well for the same. A simple GAN model uses convolution layers and upsampling layers to generate the images and the discriminator uses a similar set of convolution layers to downsample the given image and later passes it on to dense layers. I am not going to bore you with the architecture and I expect you to explore the same.

- **Now I know for a fact that training a GAN even to generate digits is going to be very challenging and computationally expensive.**

Hence I have already written some code for you!! (Almost all of it). Now I expect you to fill the blank spots with which I have marked which is to basically test your understanding of the code and the architecture itself. I have commented my code well and in case you still find some difficulties make sure to contact me for the same. Also, all the places where I have left the code blank still have a description of what is expected to be filled.

References for the architecture:
https://medium.com/@liyin2015/dcgan-79af14a1c247
https://towardsdatascience.com/dcgans-deep-convolutional-generative-adversarial-networks-c7f392c2c8f8 - this has code too!!!!

IMPORTANT NOTE:
I know the simple GAN to generate digits has a lot of code available on the internet and hence you must make sure **not to make complete changes** to my code. However, I will appreciate it if you find some redundant code or ways to make it better (faster and cleaner). This is to ensure that you gain a proper understanding of the architecture!!

Make sure to use a code editor or IDE to view the code as then you can identify the incomplete portions differentiated by color from the rest of the code!!

Please do let me know in case you face any issues.

All the Best