→ Color_mode function has default value 'RGB', 'grayscale' should be explicitly mentioned.

→ All pre-trained models have to be worked on 3 channels (RGB) so if it is grayscale image (128 x 128) then it'll be 128x128x1, 128 x 128 x 1 and 128 x 128 x 1.

* TRANSFER LEARNING

→ Using the weights of a state-of-art model to your dataset.

* AUTOENCODERS

→ unsupervised problem statements.
→ when there are 100 features, autoencoders extract useful / important features.

→ 3 components:

① Encoder
② Latent space / code
③ Decoder

→ It is mainly used for restenction of image which is application of gene

Page No.
Date

→ Encoder works exactly like CNN till flatten layer.

→ Works on both structured & unstructured data.

→ Encoder has filters in terms of descending order to scale down the no of important features.

→ Latent layer has a hyperparameter of the fixed size to which you want the no of features to be reduced.

→ Encoder is NN to extract important features and reduce the dimensionality of the dataset.

→ But the difference b/w Encoder & PCA, PCA can only work on structured data while Encoder can work on both.

→ Decoder only reconstructs the data back to original size only by those with the most important features present in encoded part (latent space).

eg: you have 2 features: House address, House area, House color, etc. The most important is House address, so this can be reconstructed as Street, ~~area~~, local & etc.

→ 2 types : (input features → features in latent space)

① Over complete : $N < M$

② under complete : $N > M$

Over complete is inefficient because it lacks redundancy. The neurons learnt features can be repeated.

→ Encoder math :

H = encoded data

$H = X \times W_e$ → Encoder matrix

↳ input

→ Tied - state assumption

$W_e = W_d^T$ or vice - versa.

$f(w) = \text{argmin}_w \|X - \hat{X}\|^2$

↳ with respect to w.

$X' = H W_d \qquad W_e = W_d^T$

$H = X W_e$

$f(w) = \text{argmin}_w \|X - \hat{X}\|^2$

$= \|X -$

→ Computation

Autoencoders < variational autoencoder < GAN's