Deep Generative models. — Generative-AI (Gen-AI)

1. Generative models

1) put machine learning model into two categories.

① generative models: models to learn the hidden distribution of the data

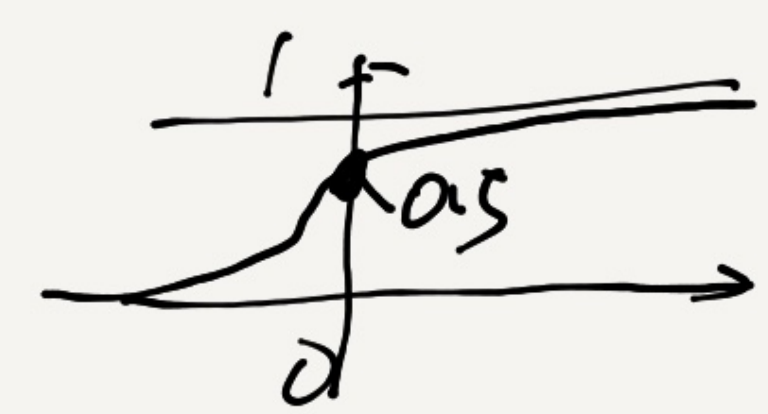
Gassian mixture model (GMM).

$$GMM = \prod_{i=1}^{K} w_i \cdot \underbrace{G_i(x; \mu_i, \delta_i^2)}_{\text{Gaussian component}}$$

K Gaussian distributions. G_1, G_2, ..., G_K

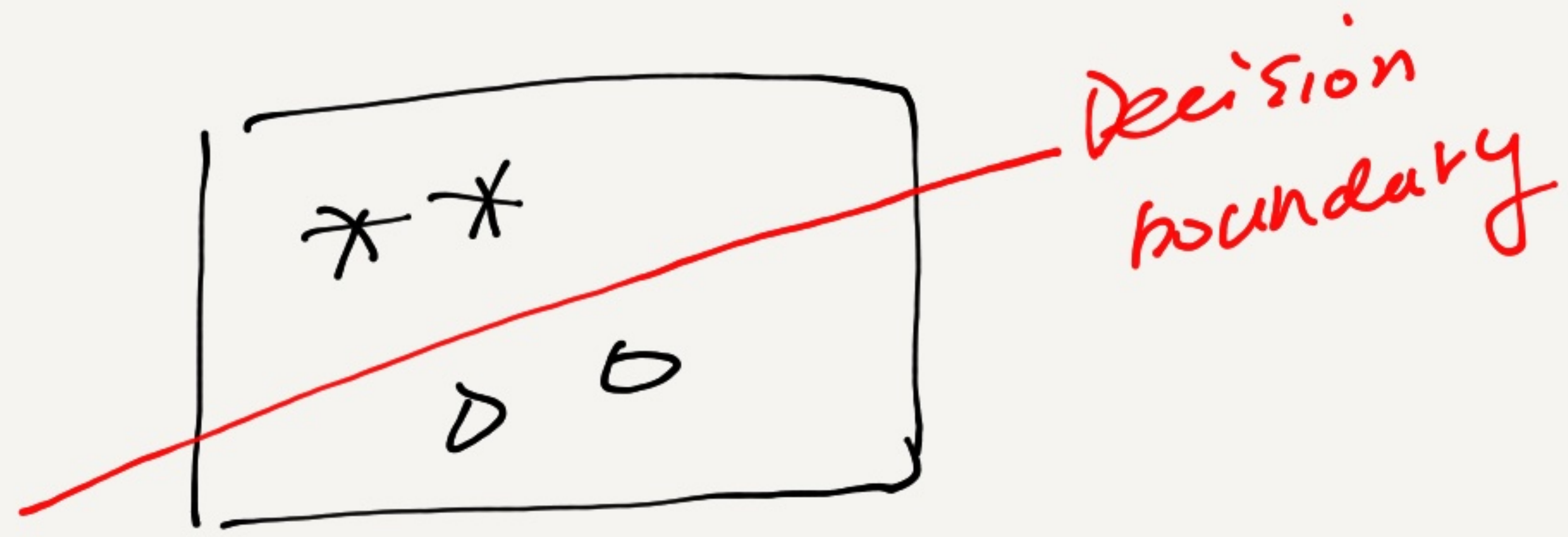
② Discriminative models: models to classify data samples and focuse on the differences of data samples.

logistic regression: binary classifier → sigmoid $(w^T x + w_0)$



2) compare the two categories

Discriminative models



k-nearest neighbors (KNN)

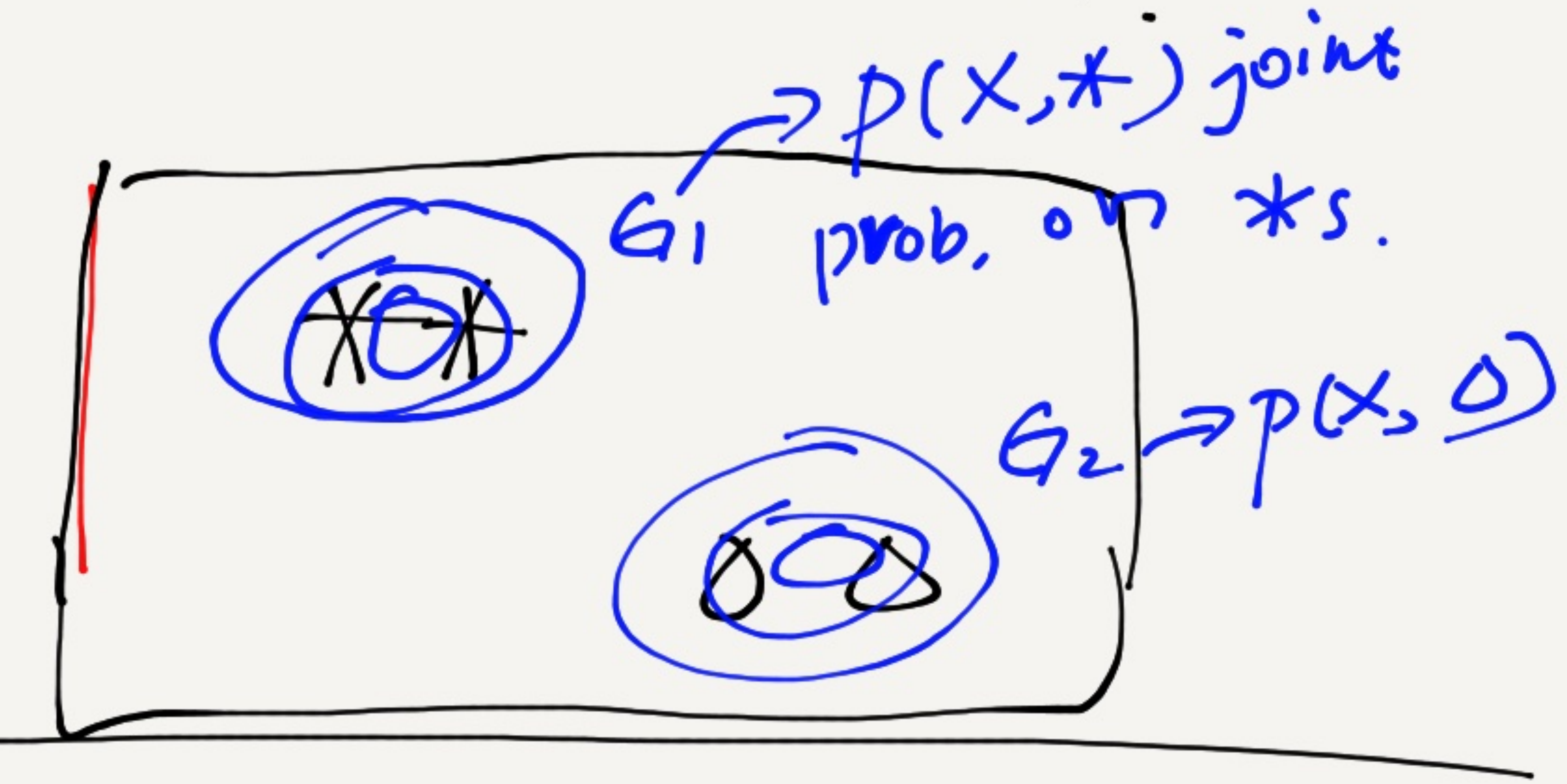
logistic regression

support vector machines (SVMs)

random forest

Conditional random fields (CRF)

Generative models



GMM

Hidden Markov model (HMM)

Bayesian Network.

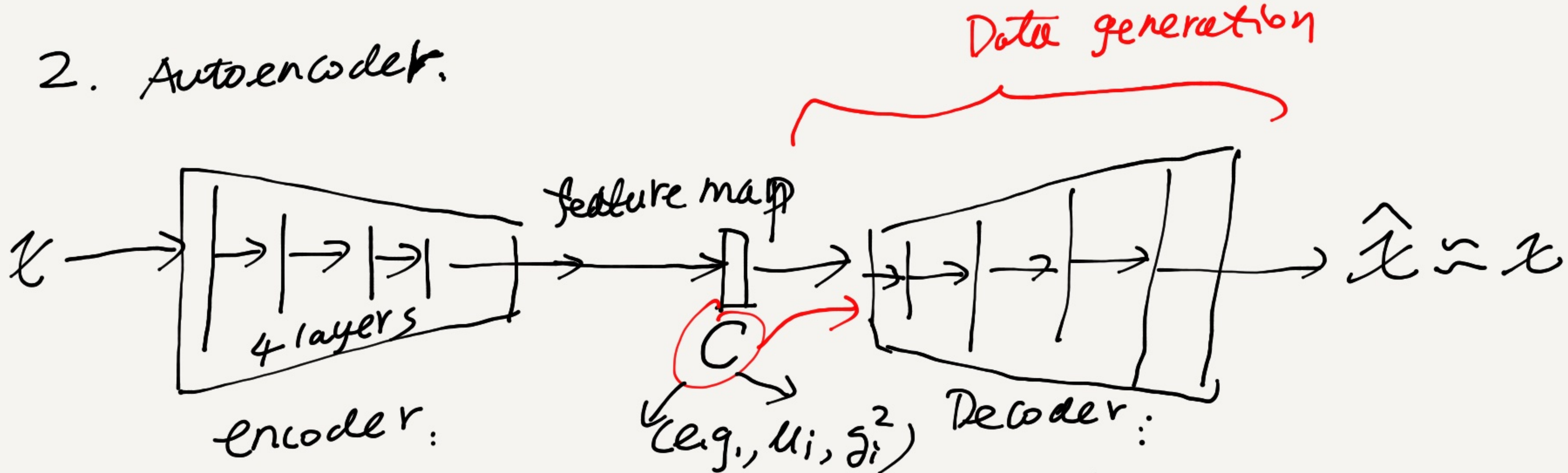
Auto-encoder (NN)

Generative adversarial Net (GAN)
2014

Diffussion models (DL-based

DM is most popular generative model)

2. Autoencoder.



Encoder:

① shorter layers have less hidden nodes

② encoder extract important features from x and reduce the dimensionality,

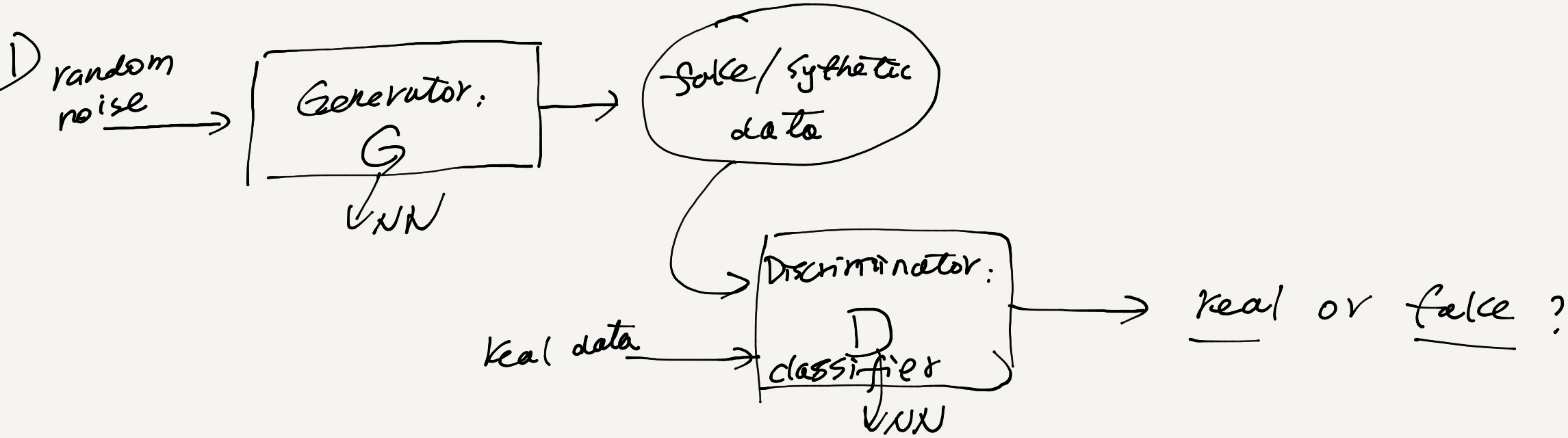
Decoder:

① tall layers have more hidden nodes.

② decoder recover data using the feature map: C.

If we get an autoencoder trained, the encoder captures the key features that control the data generation, (e.g., mean, std in Gassian function) and decoder learns to generate data using only the key features. (c) We can use the encoder to generate new data by creating new C.

3. GAN architecture. 2014.



Generator: learns to generate fake data that can fool the discriminator.

Discriminator: trained to accurately distinguish real from fake.

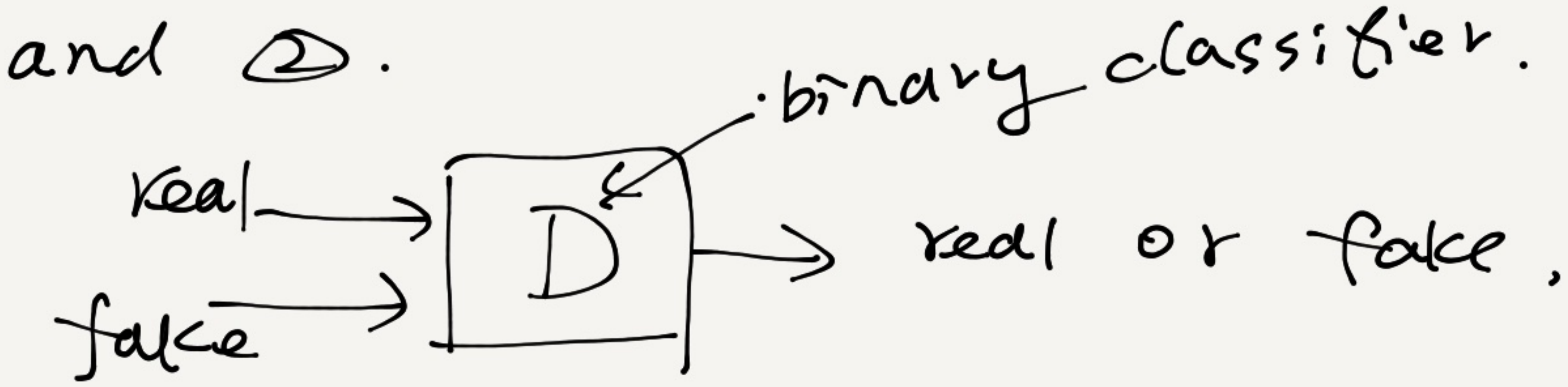
2) Train GAN

① Alternating training

① train D for one or more epochs. (fix G)

② train G for one or more epochs (fix D)

③ repeat ① and ②.



train D:

- prepare the taining set for D: real data + fake data (G)
- Calculate the loss function, e.g., binary cross-entropy.
- Apply BP algorithm to calculate the gradients and update NN weights (GD)

② Train G.

noise $z \longrightarrow \boxed{\begin{matrix} G \\ NN \end{matrix}} \xrightarrow{G(z)}$ fake data.

Sample random noise z (if we input conditions → conditional GAN)

produce the output of $G(z)$ → fake data

input $G(z)$ to D to generate $D(G(z))$ = "fake" or "real"

Calculate loss of D.

Apply GD-based approach to update weights only for G.

Loss function for GAN

min-max version.

$$-E_x[\log(D(\underbrace{x}_{\text{real data}}))] + E_x[\log(1-D(\underbrace{G(z)}_{\text{fake data}}))]$$

G.
Generator is to maximize the loss function

D is to minimize the loss.