Deep Generative Modeling

Generative Modeling

A generative model describes how a dataset is generated, in terms of a probabilistic model. By sampling from this model, we are able to generate new data.

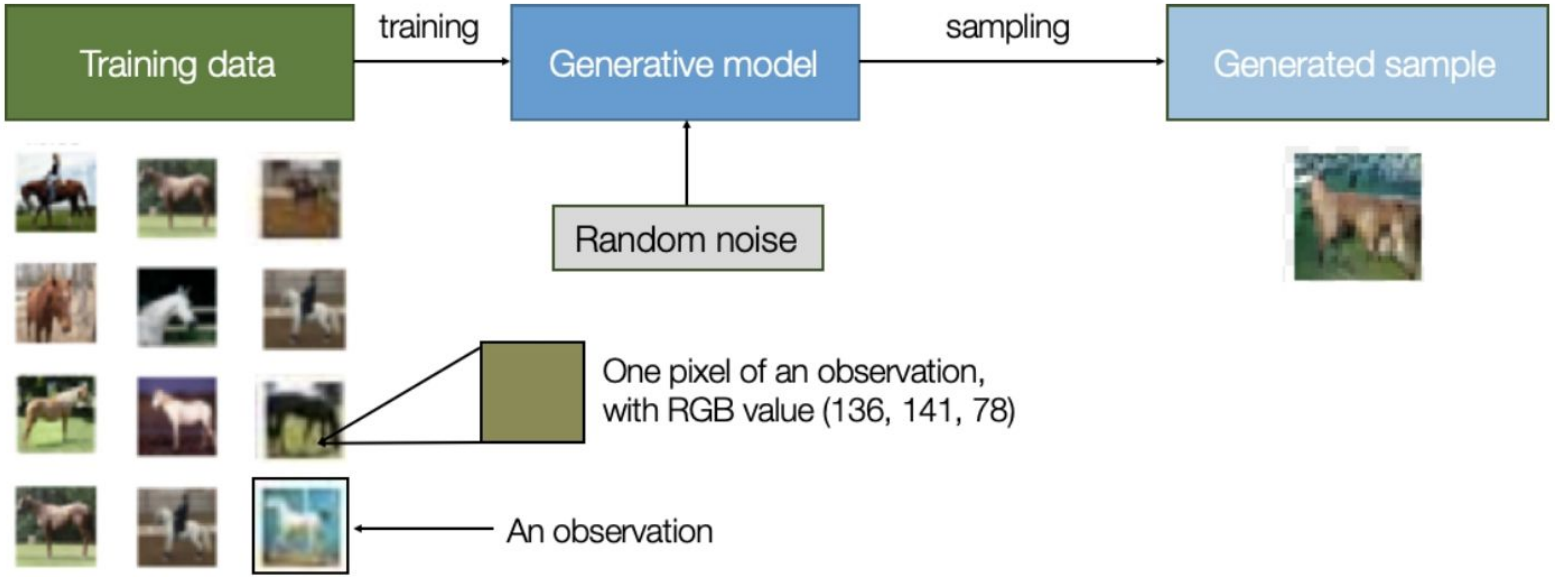


Figure 1-1. The generative modeling process

Discriminative Modeling

Discriminative modeling is synonymous with supervised learning, or learning a function that maps an input to an output using a labeled dataset.

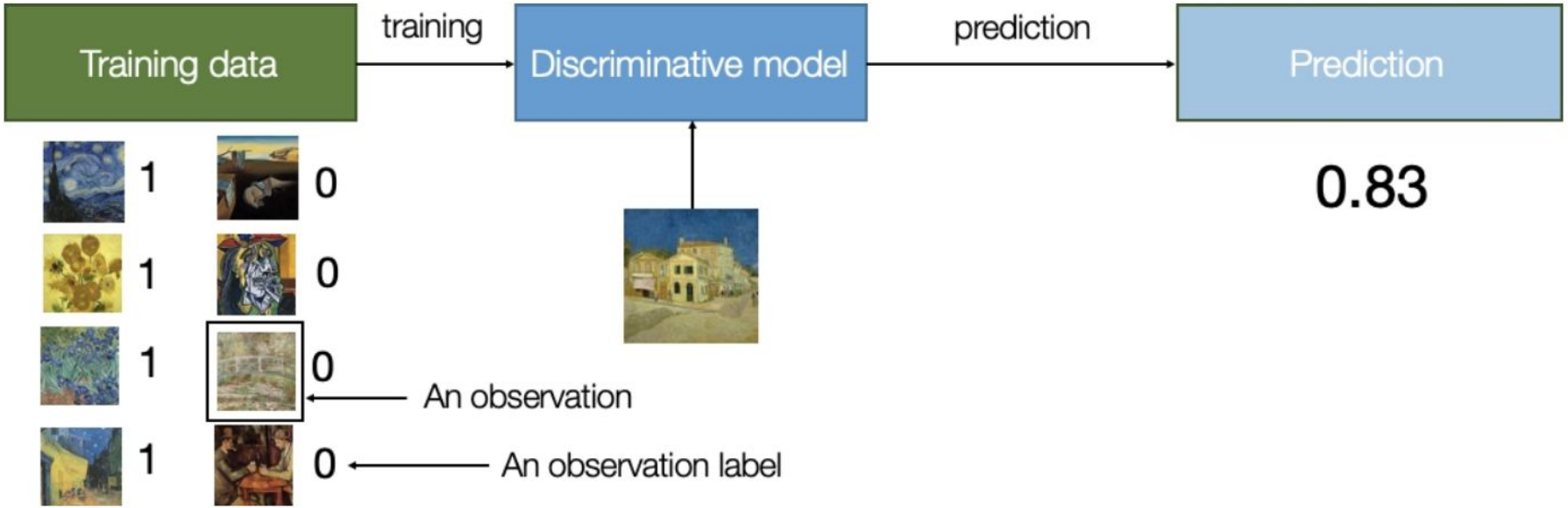


Figure 1-2. The discriminative modeling process

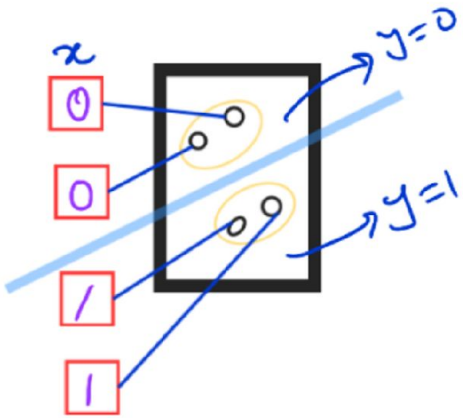
Generative Versus Discriminative Modeling

Discriminative modeling estimates p(y|x)—the probability of a label y given observation x.

Generative modeling estimates p(x)—the probability of observing observation x.

If the dataset is labeled, we can also build a generative model that estimates the distribution p(x|y).

P(y|x)



P(x)

P(x|y)

Discriminative

Generative

Parametric Modeling

P_{data}

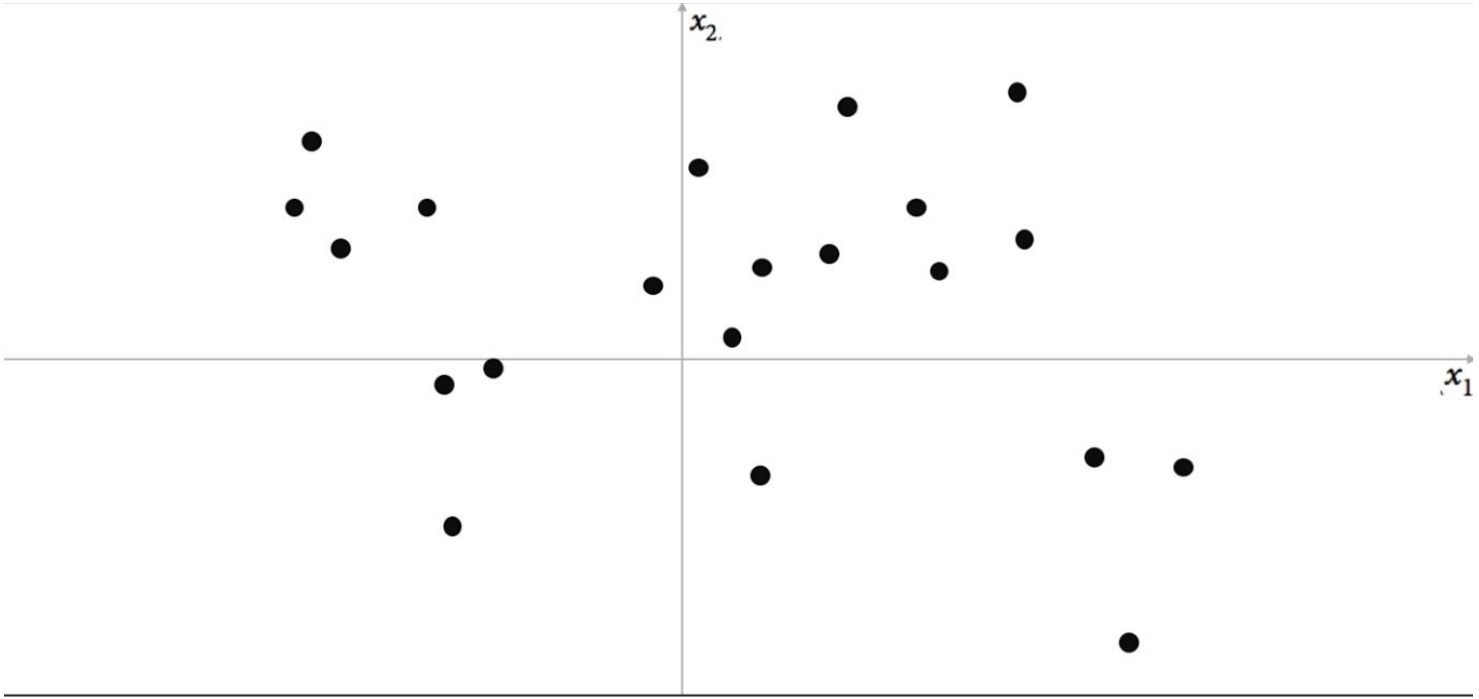
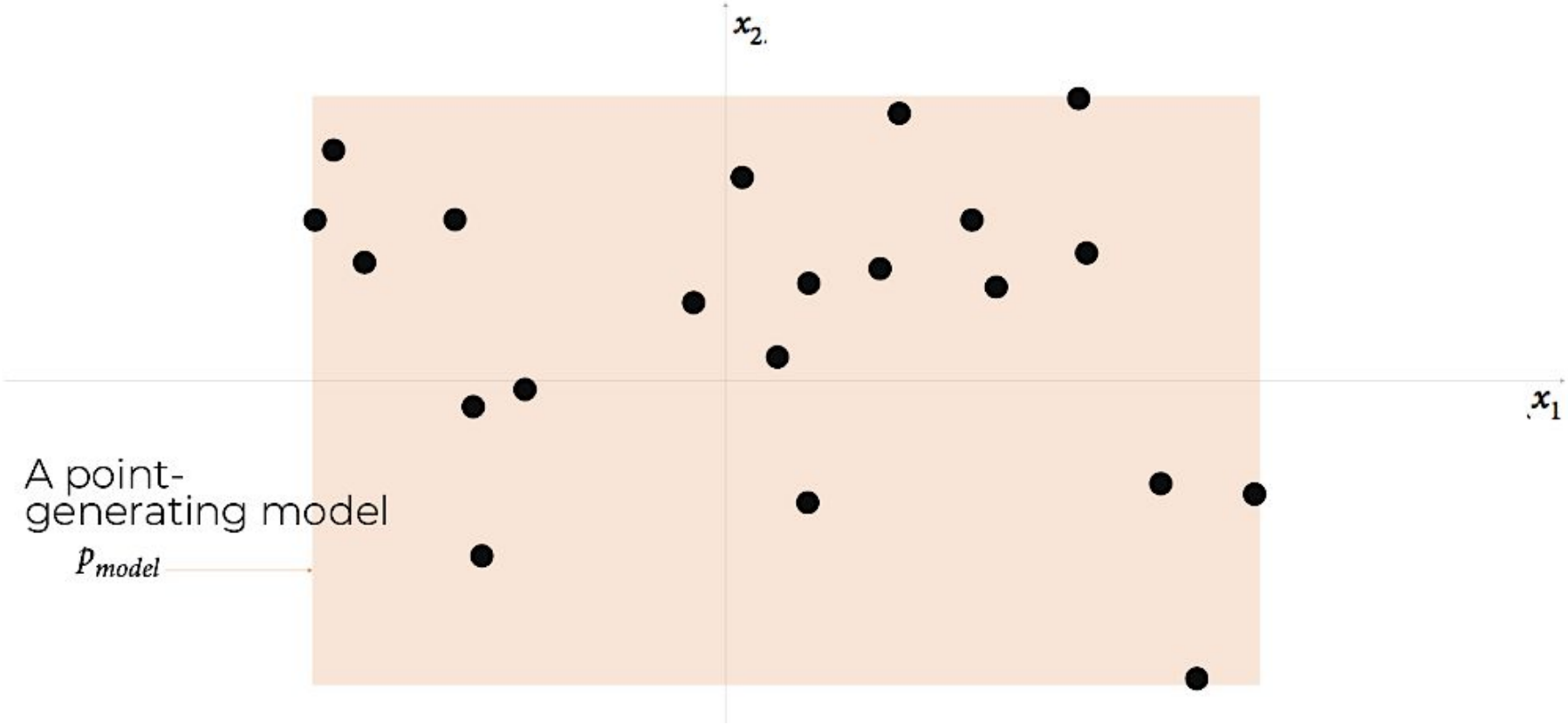


Figure 1-4. A set of points in two dimensions, generated by an unknown rule p_{data}

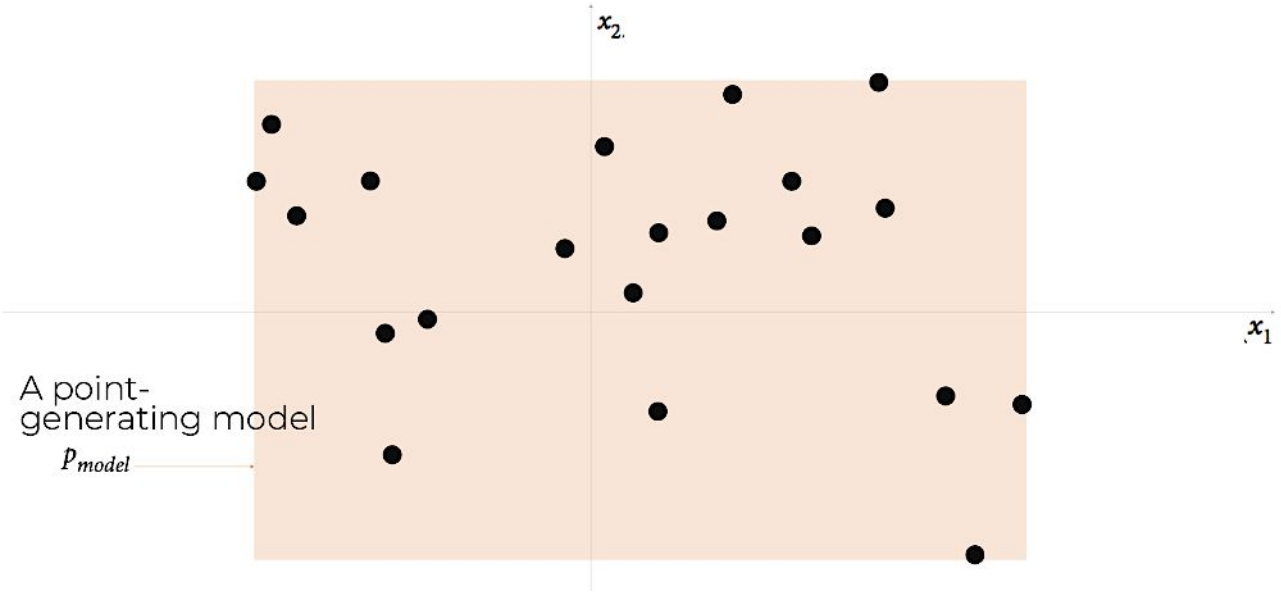
P_{model}



Parametric Modeling

$p_\theta(x)$: A parametric model, is a family of density functions that can be described using a finite number of parameters, θ.

Can we define P_{model} using a parametric function?



Parametric Modeling

- Here, there are four parameters: the coordinates of the bottom-left (θ_1,θ_2) and top-right (θ_3,θ_4) corners of the box.
- Thus, each density function $p_\theta(x)$ in this parametric model (i.e., each box) can be uniquely represented by four numbers,

$$\theta=(\theta_1, \theta_2, \theta_3, \theta_4).$$

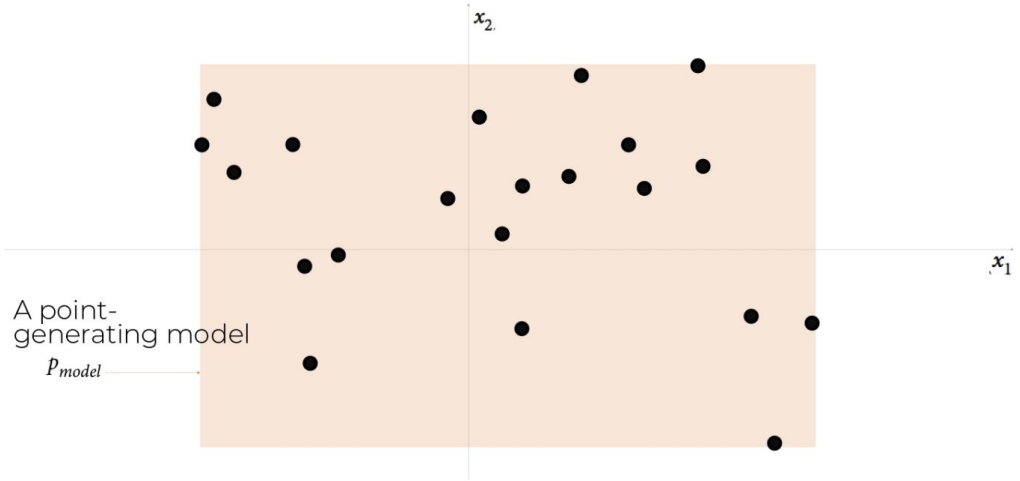


Figure 1-5. The orange box, p_{model}, is an estimate of the true data-generating distribution, p_{data}

Parametric Modeling

- The family of all possible boxes (infinite) you could draw is an example of a parametric model.
- Here, there are four parameters: the coordinates of the bottom-left (θ_1,θ_2) and top-right (θ_3,θ_4) corners of the box.
- Thus, each density function $p_\theta(x)$ in this parametric model (i.e., each box) can be uniquely represented by four numbers,

$$\theta=(\theta_1, \theta_2, \theta_3, \theta_4).$$

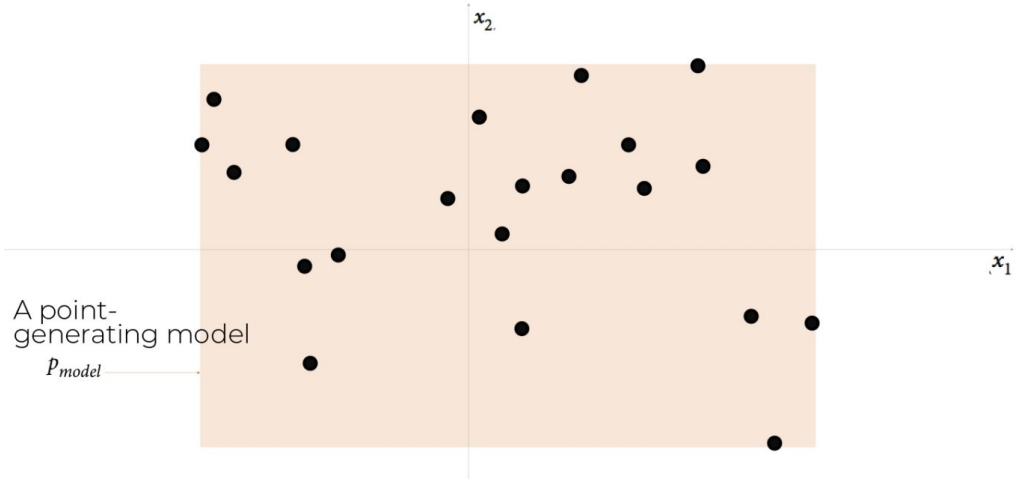


Figure 1-5. The orange box, p_{model}, is an estimate of the true data-generating distribution, p_{data}

THE GENERATIVE MODELING FRAMEWORK

- We have a dataset of observations $\mathbf{X}$.
- We assume that the observations have been generated according to some unknown distribution, p_{data}.
- A generative model p_{model} tries to mimic p_{data}. If we achieve this goal, we can sample from p_{model} to generate observations that appear to have been drawn from p_{data}.

THE GENERATIVE MODELING FRAMEWORK

- We have a dataset of observations **X**.
- We assume that the observations have been generated according to some unknown distribution, p_{data}.
- A generative model p_{model} tries to mimic p_{data}. If we achieve this goal, we can sample from p_{model} to generate observations that appear to have been drawn from p_{data}.
- We are impressed by p_{model} if:

  Rule 1: It can generate examples that appear to have been drawn from p_{data}.

  Rule 2: It can generate examples that are suitably different from the observations in **X**. In other words, the model shouldn't simply reproduce things it has already seen.

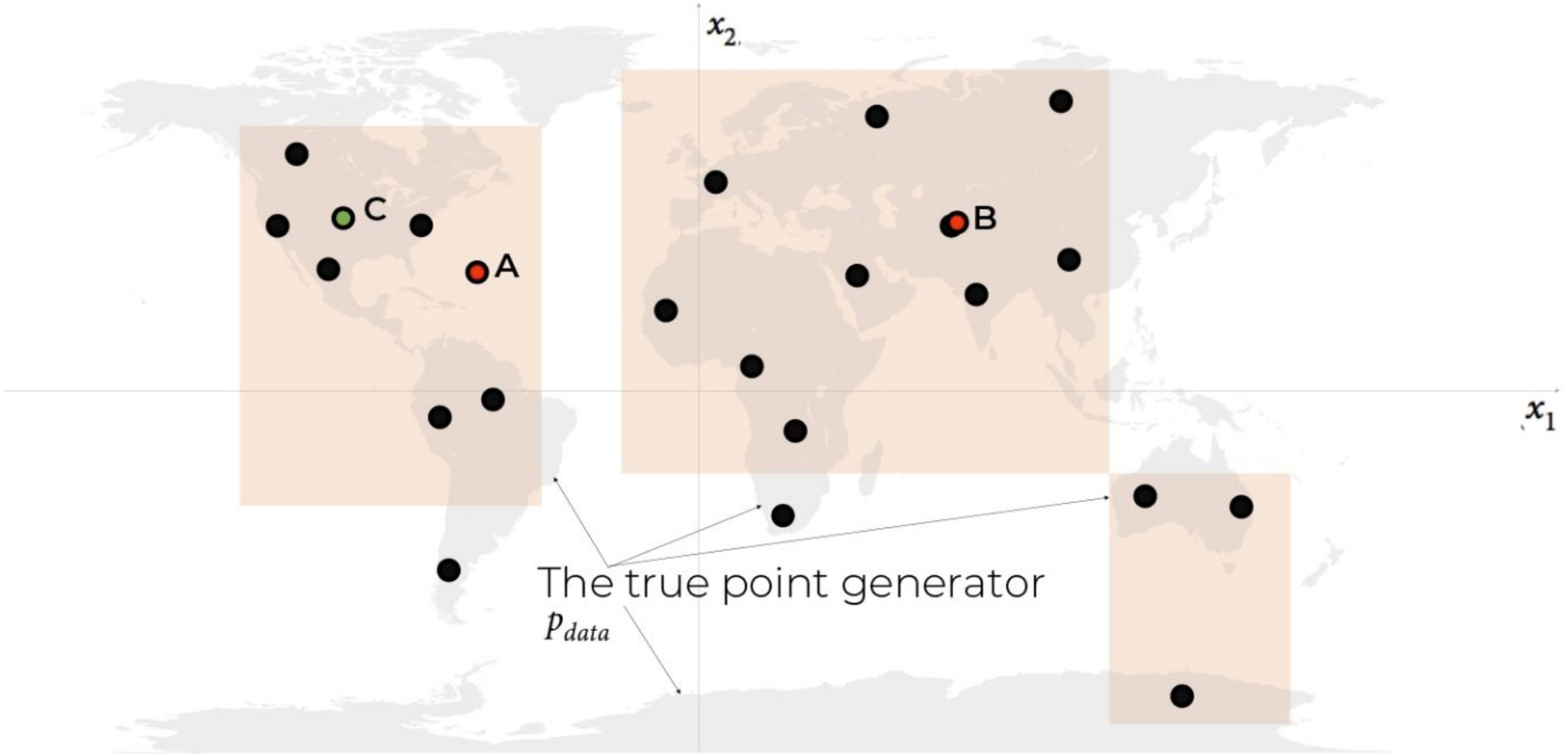


Figure 1-6. The orange box, p_{model}, is an estimate of the true data-generating distribution p_{data} (the gray area)

Your First Probabilistic Generative Model

Your First Probabilistic Generative Model

Figure 1-7. Headshots of 50 Wrodlers

Table 1-1. The first 10 observations in the Wrodler face dataset

accessoriesType	clothingColor	clothingType	hairColor	topType
Round	White	ShirtScoopNeck	Red	ShortHairShortFlat
Round	White	Overall	SilverGray	ShortHairFrizzle
Sunglasses	White	ShirtScoopNeck	Blonde	ShortHairShortFlat
Round	White	ShirtScoopNeck	Red	LongHairStraight
Round	White	Overall	SilverGray	NoHair

Wrodlers Dataset

The possible values for each feature include:

7 different hairstyles (topType):
NoHair, LongHairBun, LongHairCurly, LongHairStraight, ShortHairShortWaved, ShortHairShortFlat, ShortHairFrizzle

6 different hair colors (hairColor):
Black, Blonde, Brown, PastelPink, Red, SilverGray

3 different kinds of glasses (accessoriesType):
Blank, Round, Sunglasses

4 different kinds of clothing (clothingType):
Hoodie, Overall, ShirtScoopNeck, ShirtVNeck

8 different clothing colors (clothingColor):
Black, Blue01, Gray01, PastelGreen, PastelOrange, Pink, Red, White

points in the sample space?

Wrodlers Dataset

The possible values for each feature include:

7 different hairstyles (topType):
NoHair, LongHairBun, LongHairCurly, LongHairStraight, ShortHairShortWaved, ShortHairShortFlat, ShortHairFrizzle

6 different hair colors (hairColor):
Black, Blonde, Brown, PastelPink, Red, SilverGray

3 different kinds of glasses (accessoriesType):
Blank, Round, Sunglasses

4 different kinds of clothing (clothingType):
Hoodie, Overall, ShirtScoopNeck, ShirtVNeck

8 different clothing colors (clothingColor):
Black, Blue01, Gray01, PastelGreen, PastelOrange, Pink, Red, White

$7 \times 6 \times 3 \times 4 \times 8 = 4{,}032$

different combinations of these features, so there are 4,032 points in the sample space.

Table 1-1. The first 10 observations in the Wrodler face dataset

accessoriesType	clothingColor	clothingType	hairColor	topType
Round		ShirtScoopNeck	Red	ortHairShortFlat
Round	hite	Overall	SilverGray	Sho HairFrizzle
Sunglasses		ShirtScoopNeck	Blonde	ortHairShortFlat
Round	hite	ShirtScoopNeck	Red	LongHairStraight
Round	White	Overall	SilverGray	NoHair

P model ?

Probabilistic Generative Model

- The problem is that we do not know p_{data} explicitly—all we have to work with is the sample of observations X generated by p_{data}.

- The goal of generative modeling is to use these observations to build a p_{model} that can accurately mimic the observations produced by p_{data}.

- Let us now discuss Maximum Likelihood Estimation for synthesizing images using Wrodlers Dataset.

Likelihood

- The likelihood $\mathcal{L}(\theta \mid x)$ of a parameter set θ is a function that measures the plausibility of θ, given some observed point x.
- It is defined as follows:

$$\mathcal{L}(\theta \mid x) = p_\theta(x)$$

- That is, the likelihood of θ given some observed point x is defined to be the value of the density function parameterized by θ, at the point x.

Likelihood

- If we have a whole dataset X of independent observations then we can write:

$$\mathcal{L}(\theta|X)=\prod_{x\in X} p_{\boldsymbol{\theta}}(x)$$

- Since this product can be quite computationally difficult to work with, we often use the log-likelihood ℓ instead:

$$\ell(\theta|X)=\Sigma_{x\in X} \log p_{\boldsymbol{\theta}}(x)$$

Maximum Likelihood Estimation

- Maximum likelihood estimation is the technique that allows us to estimate θ^*.
- θ^* is the set of parameters θ of a density function $p_{\boldsymbol{\theta}}(x)$, that are most likely to explain some observed data X.

More formally:

$$\theta^* = \text{argmax}_\theta \, \mathcal{L}(\theta | X)$$

θ^* is also called the maximum likelihood estimate (MLE).

Multinomial distribution as P_{model}

Let's take a particular class of parametric model, known as a multinomial distribution. Then, maximum likelihood estimate θ_j^* of each parameter is given by:

$$\theta_j^* = \frac{n_j}{N}$$

Where, n_j is the number of times that combination j was observed in the dataset and $N = 50$ is the total number of observations.

Multinomial distribution as P_{model}

Let's take a particular class of parametric model, known as a multinomial distribution. Then, maximum likelihood estimate θ_j^* of each parameter is given by:

$$\theta_j^* = \frac{n_j}{N}$$

Where, n_j is the number of times that combination j was observed in the dataset and $N = 50$ is the total number of observations.

For example, the following combination appears twice in dataset:

(LongHairStraight, Red, Round, ShirtScoopNeck, White).

$$\textit{Therefore}: \ \theta_1^* = \frac{2}{50} = 0.04$$

Multinomial distribution as P_{model}

Multinomial distribution assumption can never generate anything that it hasn't already seen, since $\theta_j^* = 0$ for any combination that wasn't in the original dataset X.

Solution:

Multinomial distribution as P_{model}

Multinomial distribution assumption can never generate anything that it hasn't already seen, since $\theta_j^* = 0$ for any combination that wasn't in the original dataset X.

Solution: lets assign an additional pseudocount of 1 to each possible combination of features.
Under new solution model, our MLE for the parameters would be:

$$\theta_j^* = \frac{n_j + 1}{N + d}$$

But probability of observing a point not in original dataset is constant.

Naive Bayes as P_{model} (Generative model)

X

accessoriesType	clothingColor	clothingType	hairColor	topType
Round	White	ShirtScoopNeck	Red	ShortHairShortFlat
Round	White	Overall	SilverGray	ShortHairFrizzle
Sunglasses	White	ShirtScoopNeck	Blonde	ShortHairShortFlat
Round	White	ShirtScoopNeck	Red	LongHairStraight
Round	White	Overall	SilverGray	NoHair

$X = X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5$

$x = x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5$

Naive Bayes Assumption

The possible values for each feature include:

7 different hairstyles (topType):
NoHair, LongHairBun, LongHairCurly, LongHairStraight, ShortHairShortWaved, ShortHairShortFlat, ShortHairFrizzle

6 different hair colors (hairColor):
Black, Blonde, Brown, PastelPink, Red, SilverGray

3 different kinds of glasses (accessoriesType):
Blank, Round, Sunglasses

4 different kinds of clothing (clothingType):
Hoodie, Overall, ShirtScoopNeck, ShirtVNeck

8 different clothing colors (clothingColor):
Black, Blue01, Gray01, PastelGreen, PastelOrange, Pink, Red, White

Suppose in Wrodler dataset:

"*the choice of hair color has no impact on the choice of clothing type, and the type of glasses*".

Naive Bayes as $P_{\textbf{model}}$ (Generative model)

We make the naive assumption that each feature x_j is independent of every other feature x_k.

$$p(x_j \mid x_k) = p(x_j)$$

Chain rule of probability:

Naive Bayes as P_{model} (Generative model)

We make the naive assumption that each feature x_j is independent of every other feature x_k.

$$p(x_j \mid x_k) = p(x_j)$$

Chain rule of probability: Write the density function as a product of conditional probabilities as follows:

$$\begin{aligned} p(x) &= p(x_1, \ldots, x_K) \\ &= p(x_K \mid x_1, \ldots, x_{K-1})\, p(x_1, \ldots, x_{K-1}) \end{aligned}$$

Where **K** is the total number of features.

Naive Baye

We now apply the Naive Bayes assumption to simplify the last line:

$$p(x) = \prod_{k=1}^{K} p\left(x_k\right) \qquad \text{Where } \mathbf{K} \text{ is the total number of features.}$$

The problem is reduced to estimating the parameters $\theta_{kl} = p\left(x_k = l\right)$ for each feature separately.

Multiplying these to find the probability for any possible combination.

$$\mathbf{X} = \quad \mathbf{X}_1 \qquad \mathbf{X}_2 \qquad \mathbf{X}_3 \qquad \mathbf{X}_4 \qquad \mathbf{X}_5$$

$$P(X) = P(X_1, X_2, X_3, X_4, X_5)$$

$$\mathbf{X} = \quad \mathbf{X}_1 \qquad \mathbf{X}_2 \qquad \mathbf{X}_3 \qquad \mathbf{X}_4 \qquad \mathbf{X}_5$$

$$\begin{aligned} P(X) &= P(X_1, X_2, X_3, X_4, X_5) \\ &= P(X_1) \cdot P(X_2) \cdot P(X_3) \cdot P(X_4) \cdot P(X_5) \\ &= P(X_1 = x_1) \cdot P(X_2 = x_2) \cdot P(X_3 = x_3) \cdot P(X_4 = x_4) \cdot P(X_5 = x_5) \end{aligned}$$

Naive Baye in Wrodler dataset

The maximum likelihood estimates $\theta^{\star}_{kl}$ are as follows:

$$\theta^{\star}_{kl} = \frac{n_{kl}}{N}$$

Where n_{kl} is the number of times that the feature k takes on the value l in the dataset and $\mathrm{N} = 50$ is the total number of observations.

To calculate the probability of the model generating some observation x, we simply multiply together the individual feature probabilities.

Naive Bayes

For example:

p(LongHairStraight, Red, Round, ShirtScoopNeck, White)

= p(LongHairStraight) × p(Red) × p(Round) × p(ShirtScoopNeck) × p(White)

= 0.46 × 0.16 × 0.44 × 0.38 × 0.44 = = 0.0054

Naive Bayes (Any advantage?)

- Combination that doesn’t appear in the original dataset?

- Sample probabilities?

Naive Bayes

- The combination(LongHairStraight, Red, Round, ShirtScoopNeck, White) doesn’t appear in the original dataset, but our model still allocates it a nonzero probability (0.0054), so it is still able to be generated.

- Also, it has a higher probability of being sampled than, say, (LongHairStraight, Red, Round, ShirtScoopNeck, White), because white clothing appears more often than blue clothing in the dataset.

Naive Bayes as P_{model}

Figure 1-8. Ten new Wrodl styles, generated using the Naive Bayes model

Planet Pixel

each observation now has 32 × 32 = 1,024 features

Figure 1-9. Fashions on Planet Pixel

Planet Pixel

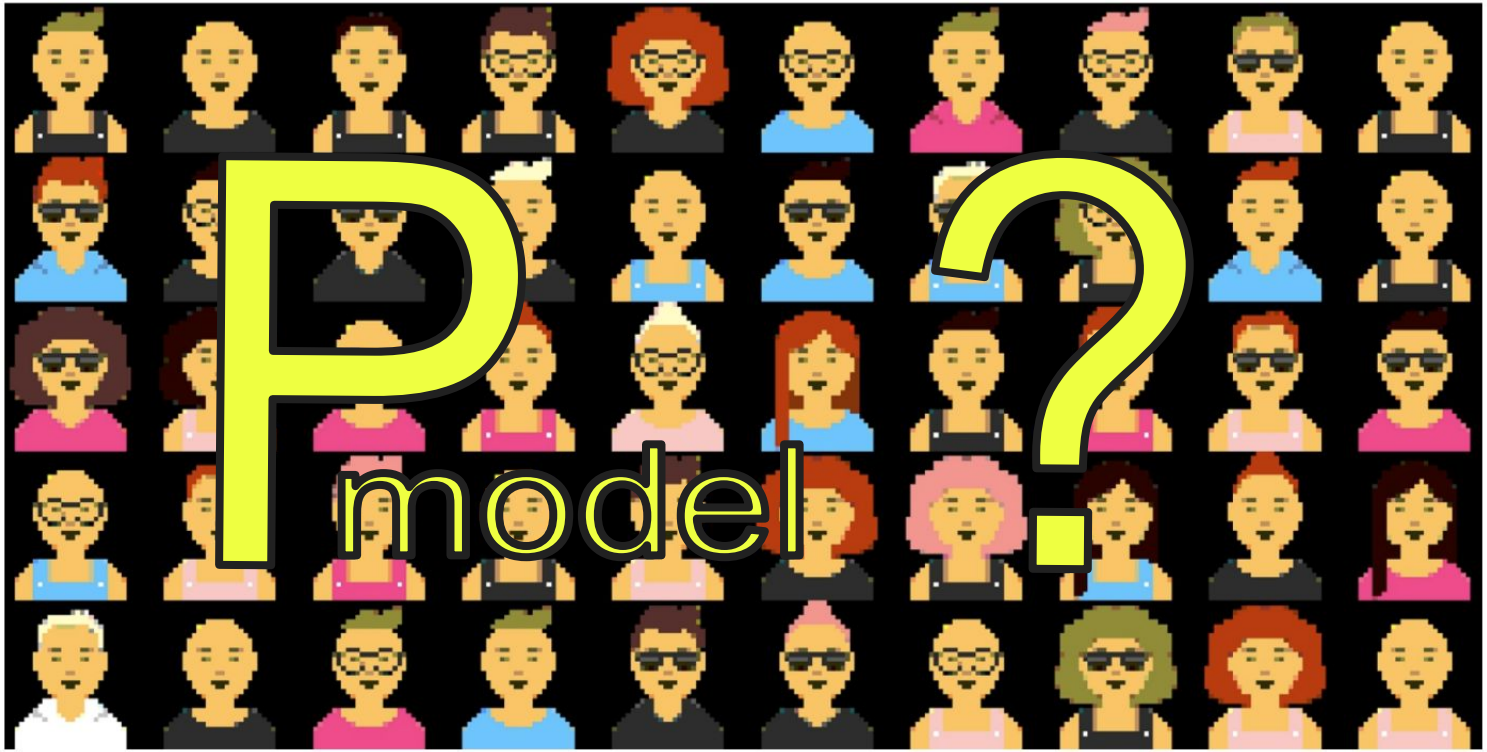


Figure 1-9. Fashions on Planet Pixel

Can we use Naive Bayes as $P_{\mathbf{model}}$ for Planet Pixel dataset ?

Figure 1-9. Fashions on Planet Pixel

Table 1-3. The values of pixels 458–467 from the first 10 observations on Planet Pixel

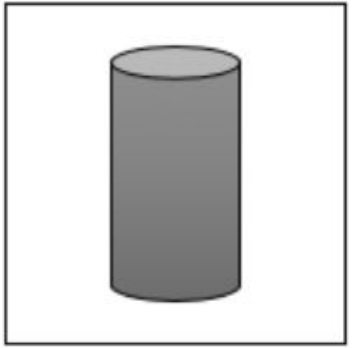
face_id	px_458	px_459	px_460	px_461	px_462	px_463	px_464	px_465	px_466	px_467
0	49	14	14	19	7	5	5	12	19	14
1	43	10	10	17	9	3	3	18	17	10
2	37	12	12	14	11	4	4	6	14	12
3	54	9	9	14	10	4	4	16	14	9
4	2	2	5	2	4	4	4	4	2	5
5	44	15	15	21	14	3	3	4	21	15

Naive Bayes as $P_{\mathbf{model}}$ for Planet Pixel dataset

Figure 1-10. Ten new Planet Pixel styles, generated by the Naive Bayes model

The Challenges of Generative Modeling

Height and width can describe the Tin box →



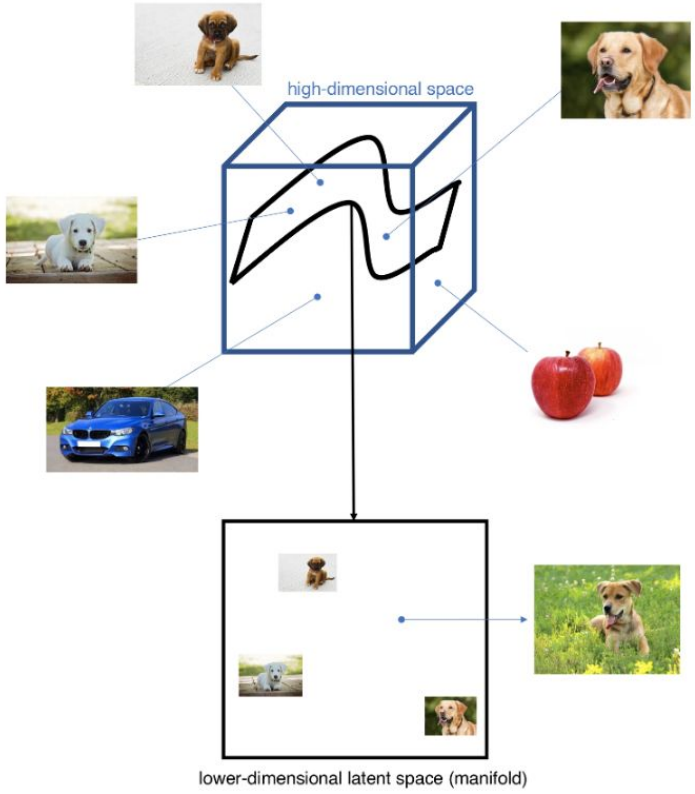
- How does the model generated observations from the high-dimensional space of images?

Low dimensional latent space having height and width as representations

Features like hair color, clothing type can describe the image →

The Challenges of Generative Modeling

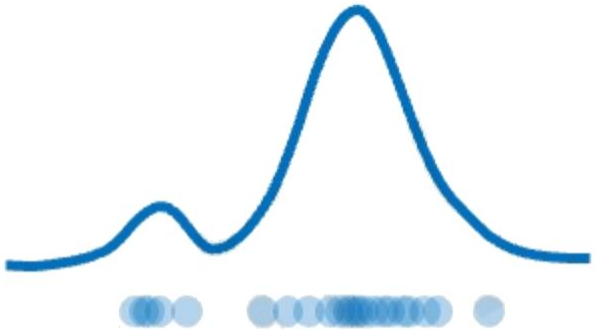
- How does the model generated observations from the high-dimensional space of images?
  - **Representation Learning**



Generative Modeling

Goal: Take as input training samples from some distribution and learn a model that represents that distribution

Density Estimation



samples

Sample Generation

Input samples

Training data ~ $P_{data}(x)$

Generated samples

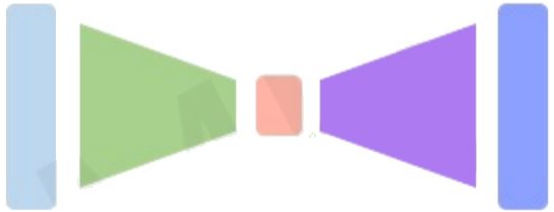
Generated ~ $P_{model}(x)$

How can we learn $P_{model}(x)$ similar to $P_{data}(x)$?

Deep Generative Models

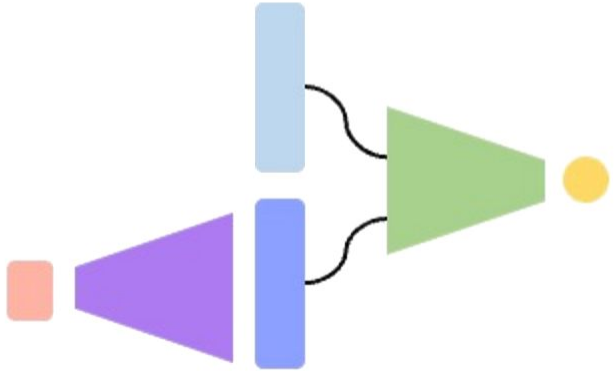
Autoencoders and Variational Autoencoders (VAEs)

Learn lower-dimensional **latent space** and **sample** to generate input reconstructions



Generative Adversarial Networks (GANs)

Competing **generator** and **discriminator** networks



What if we just want to sample?

Idea: don't explicitly model density, and instead just sample to generate new instances.

Problem: want to sample from complex distribution – can't do this directly!

Solution: sample from something simple (e.g., noise), learn a transformation to the data distribution.