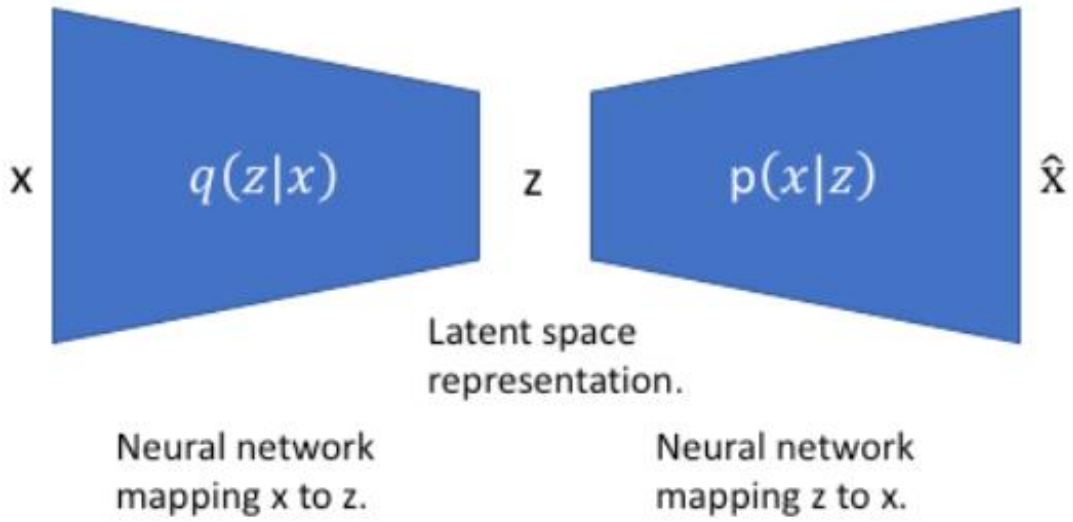
VAE: Variational Autoencoder:

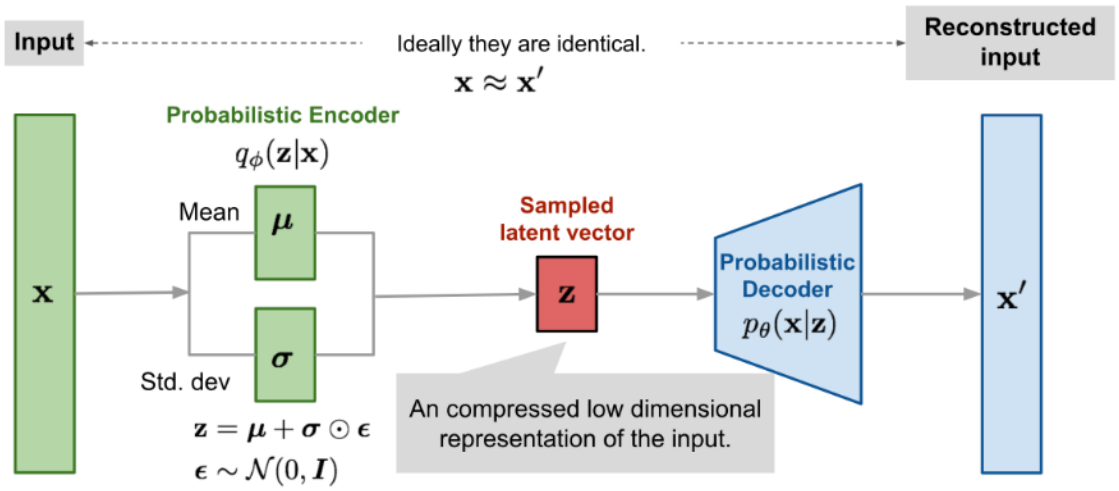
https://lilianweng.github.io/posts/2018-08-12-vae/#vq-vae-and-vq-vae-2

https://www.jeremyjordan.me/variational-autoencoders/

https://www.youtube.com/watch?v=qJeaCHQ1k2w

- **Objective/ Goal of VAE: Trying to find distribution $q_\varphi(z|x)$ from where we sample $z \sim q_\varphi(z|x)$ to generate new sample x' from $p_\theta(x|z)$**





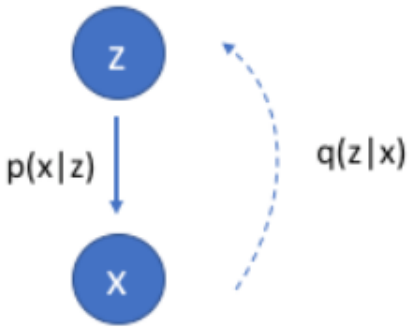
$$L_{\text{VAE}}(\theta, \phi) = -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) + D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}))$$

2. **Architecture of VAE**

- **Encoder** $q_\phi(z|x)$:
  - Maps the input x to a latent distribution $q_\phi(z|x)$, which is typically modeled as a Gaussian distribution.
- **Latent Variable Sampling** $z \sim q_\phi$:
  - z represents compressed information from x.
  - Sampling allows z to capture variability in the data.
- **Decoder** $p_\theta(x|z)$:
  - Reconstructs or generates samples x' based on the latent variable z.

3. **Learning Process**

- **Optimization Variables:**
  - ϕ: Parameters of the encoder (probabilistic mapping to latent space).
  - θ: Parameters of the decoder (probabilistic mapping from latent space back to the data).
  - Both are learned through **backpropagation** using the VAE loss function.
- **Loss Function:**
  - Balances two terms:
    1. **Reconstruction Loss:** Ensures x' is similar to x (e.g., log-likelihood).
    2. **KL Divergence:** Regularizes $q_\phi(z|x)$ to be close to a prior distribution (commonly standard Gaussian).



4. Graphical Model

- Highlights the probabilistic relationships:
  - z: Latent variable (hidden).
  - x: Observed variable (input).
  - Arrows show how z is sampled using $q_\phi(z|x)$, and x' is generated using $p_\theta(x|z)$.

We'd like to use our observations to understand the hidden variable.

5. Steps Summarized:

1. Input x is fed into the encoder to compute $q_\phi(z|x)$.
2. Sample z from $q_\phi(z|x)$.
3. Use z as input to the decoder $p_\theta(x|z)$ to generate x'.
4. Optimize the loss function to align x' with x while regularizing $q_\phi(z|x)$.

1. Goal of VAE

- The objective of VAEs is to learn a **distribution $q_\phi(z|x)$** of some **latent variables** z. From which we can sample **$z \sim q_\phi(z|x)$ to generate x' from $p_\theta(x|z)$**
- Instead of just encoding and decoding data deterministically (as in typical autoencoders), VAEs aim to:
  - Encode input **x** into a latent space **z** such that **z** represents a probability distribution (usually Gaussian).
  - Generate new samples by sampling from this latent space distribution **$z \sim q_\phi(z|x)$**.

- Use a decoder $p_\theta(x|z)$ to reconstruct samples **x′**.

Expectation of a random variable

$$E_x[f(x)] = \int xf(x)dx$$

Chain rule of probability

$$P(x, y) = P(x|y)P(y)$$

Bayes' Theorem

$$P(x \mid y) = \frac{P(y|x)P(x)}{P(y)}$$

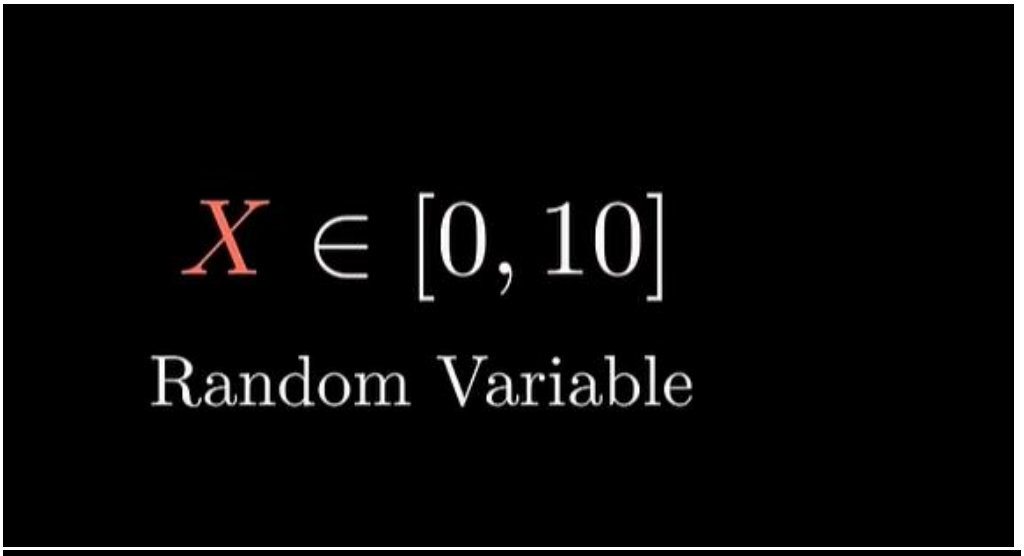
Kullback-Leibler Divergence

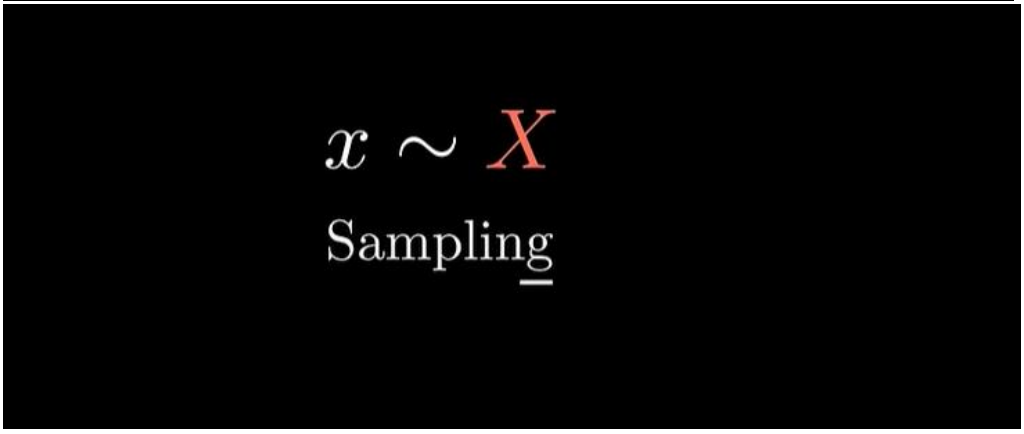
$$D_{KL}(P\|Q) = \int p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$$

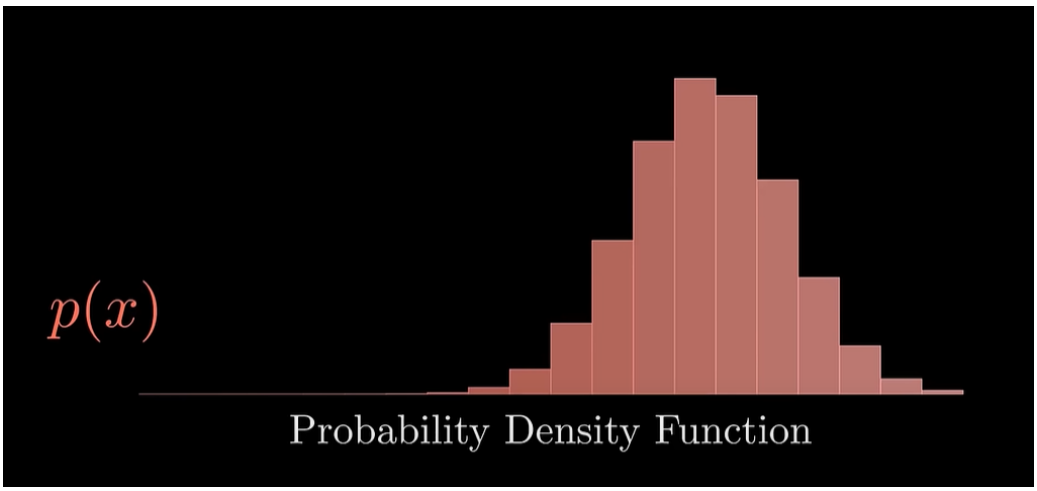
Properties:

- Not symmetric.
- Always ≥ 0
- It is equal to 0 if and only if $P = Q$

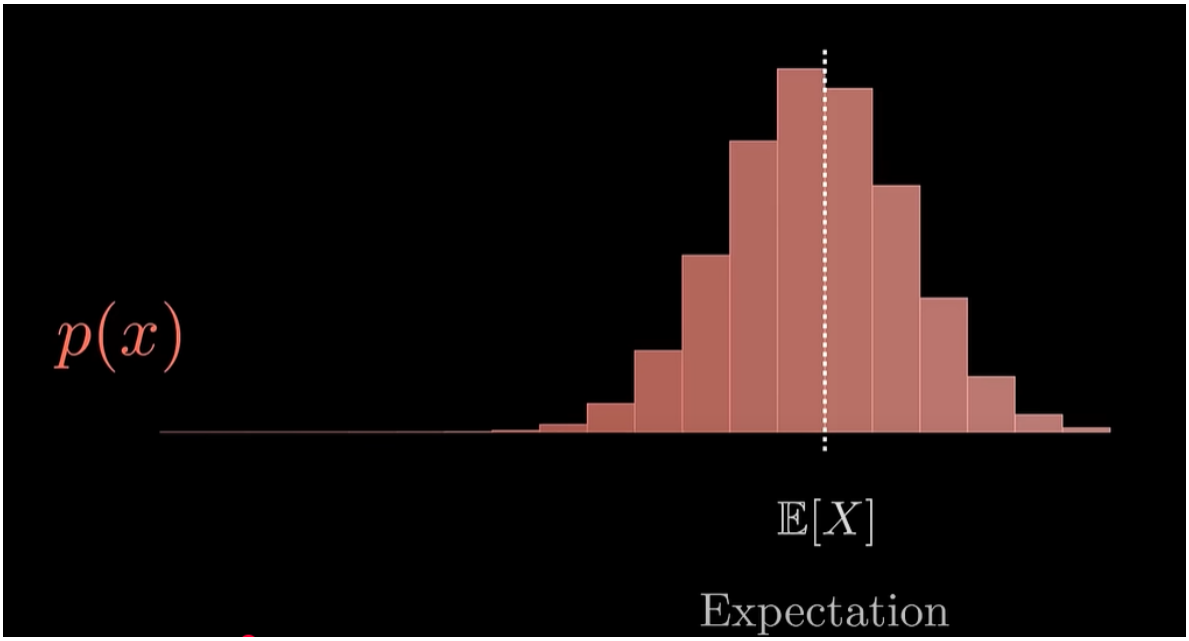
https://www.youtube.com/watch?v=qJeaCHQ1k2w



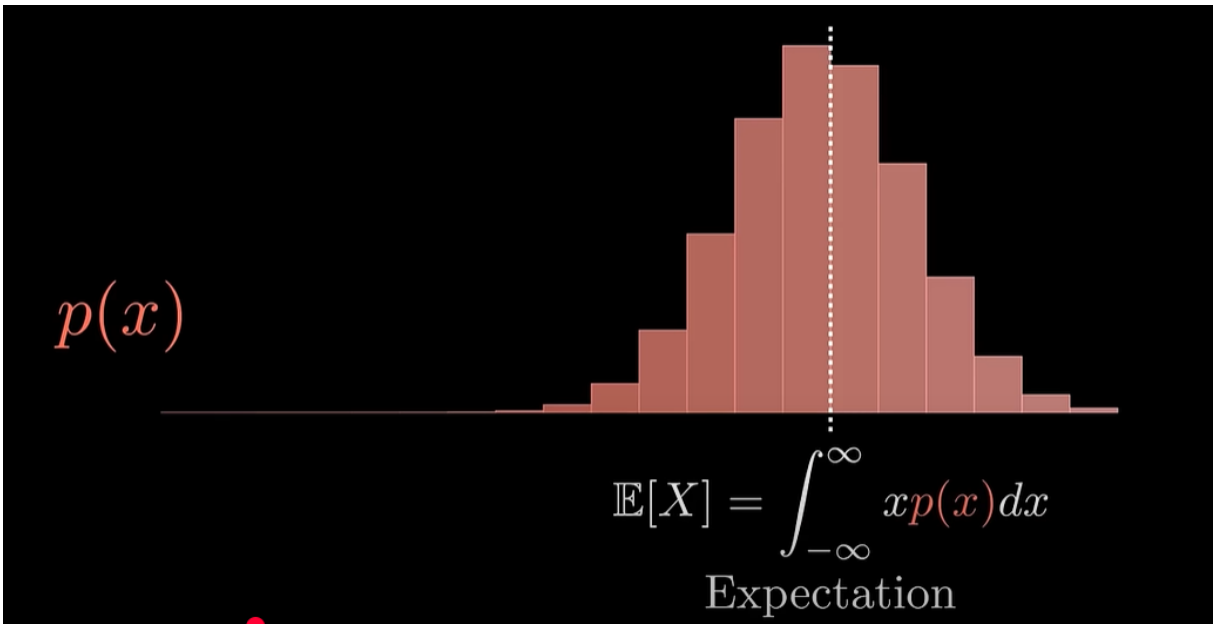




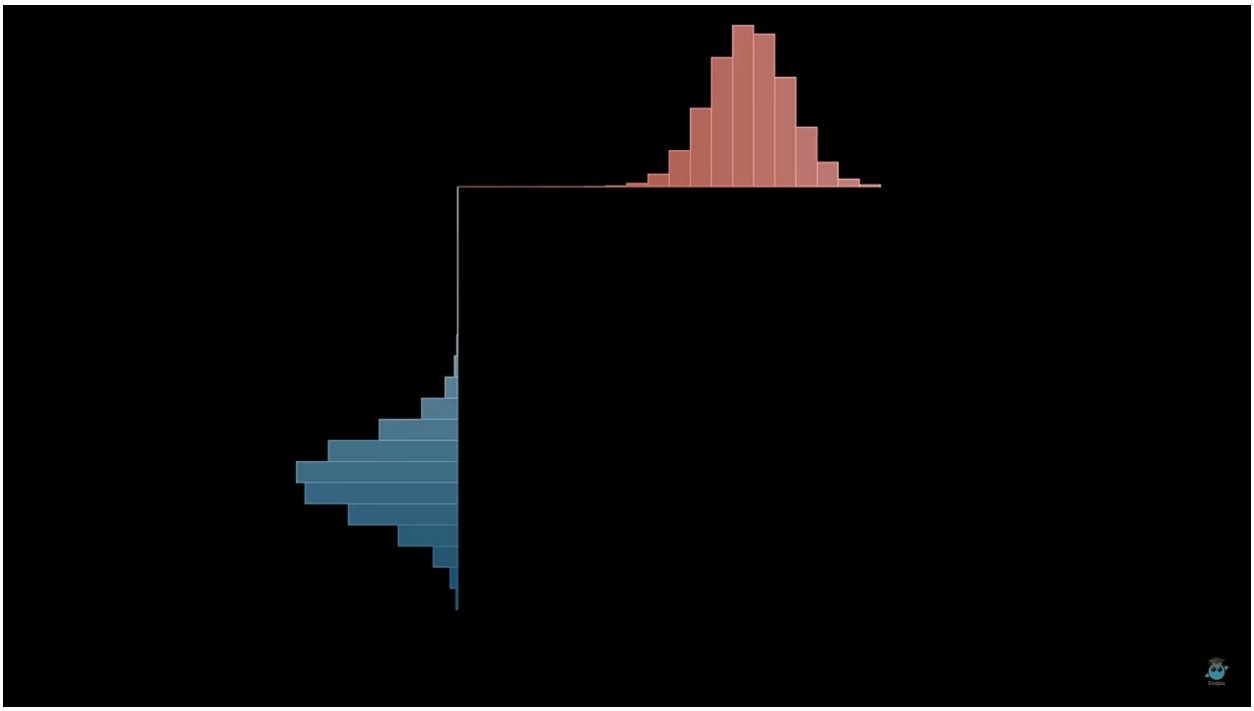
P(x) is the probability of any value of X

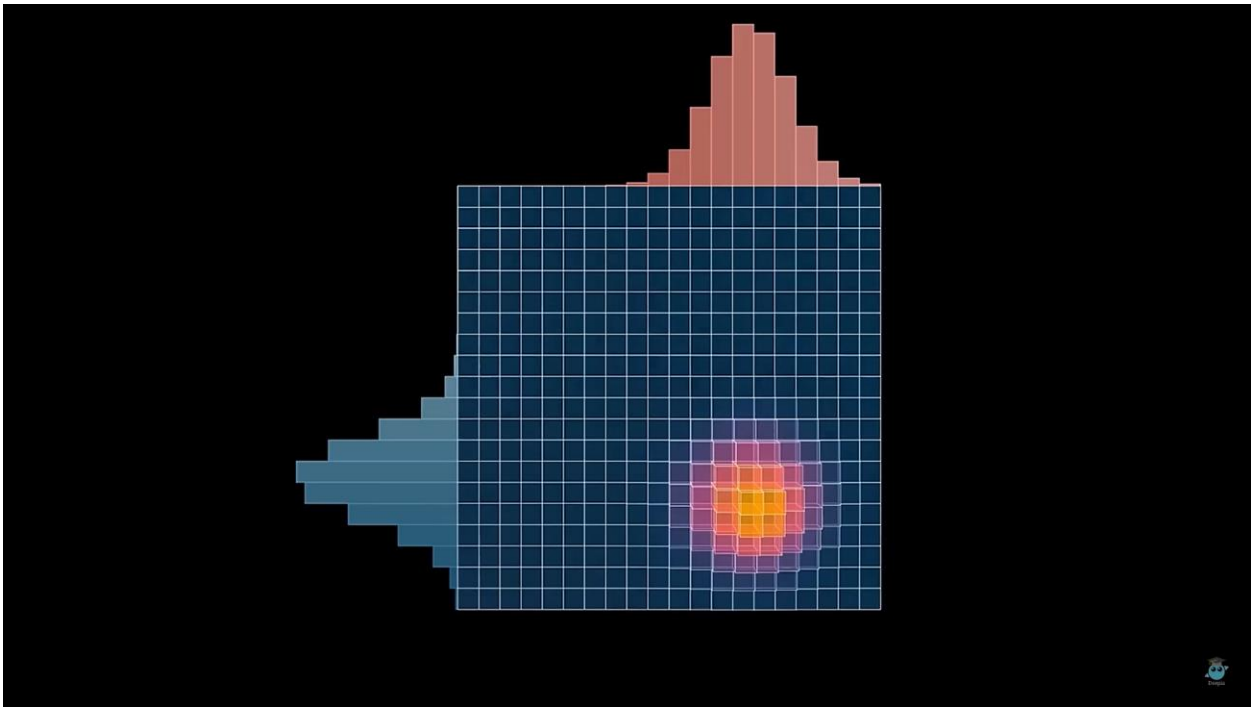


The interesting quantity is E[X] is the expectation of X,. It the average value of X when we sampling from X. and is computed by the following formulation. CAN show the previous Lecture example



Let consider two random variable X and Z, and their joint probability



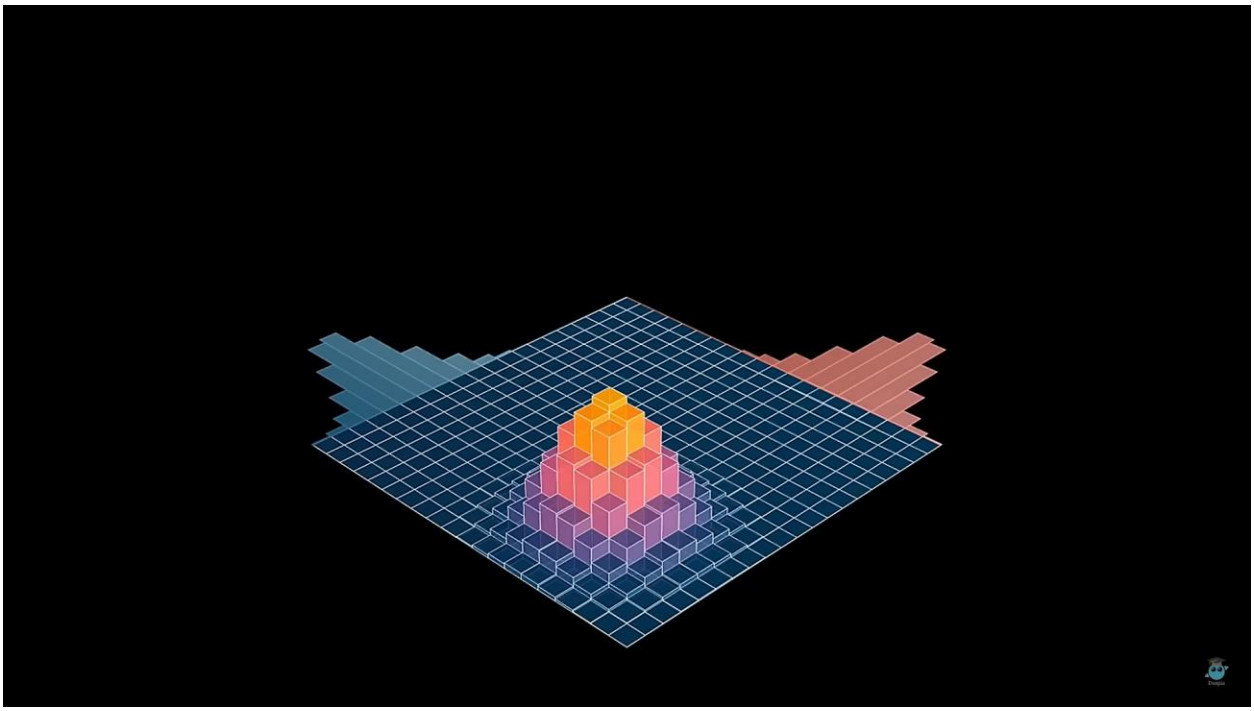


In 3D, we can see the joint probability of X and Z. it shows **the each pair of events occurs together**.

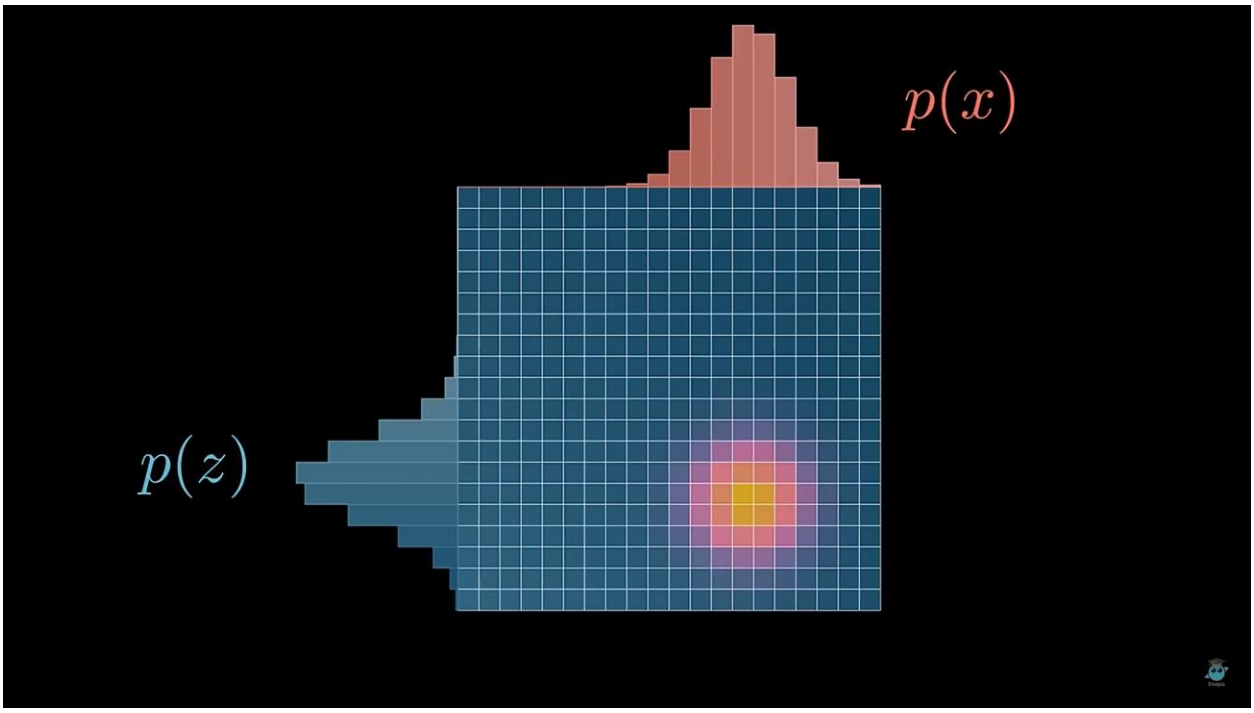
[show examples of joint distribution.

If X = {H, T}, and Z = {1,2,3,4,5,6}

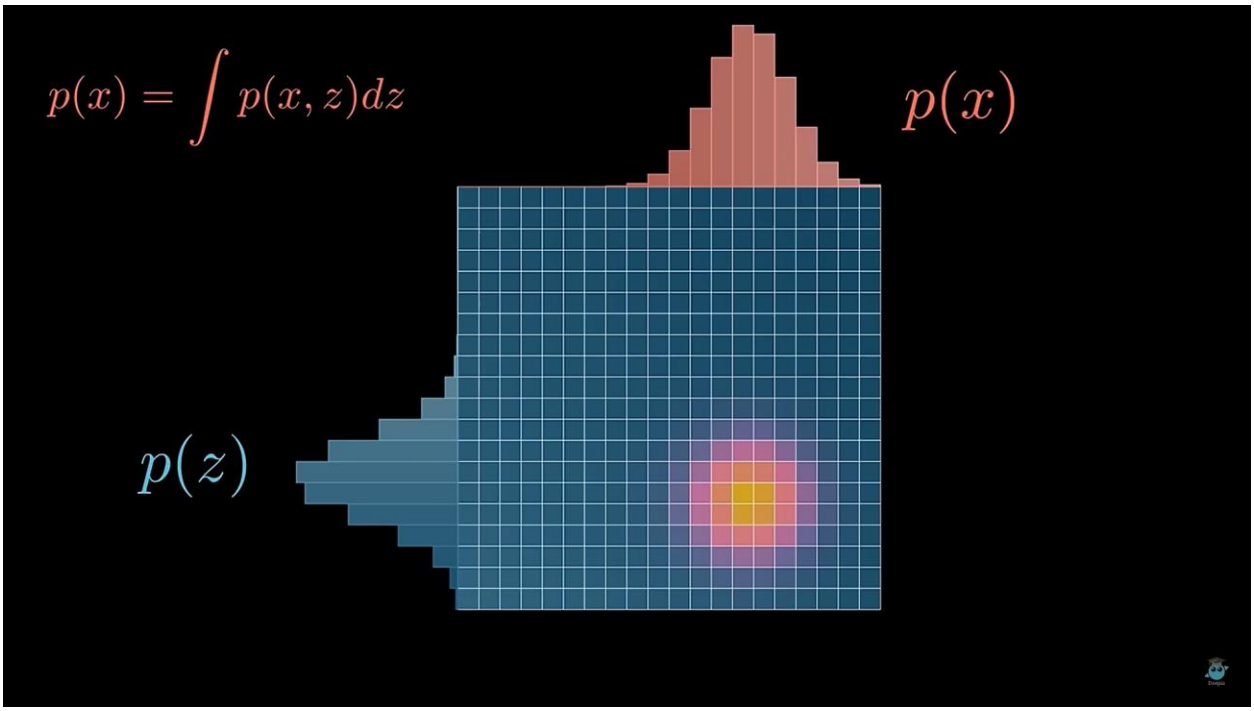
P(x = H, Y= 3) = 1/ 2 * 1/6 = 1/12]



At the same time each Random variable has its own probability distribution called **the marginal distribution.**



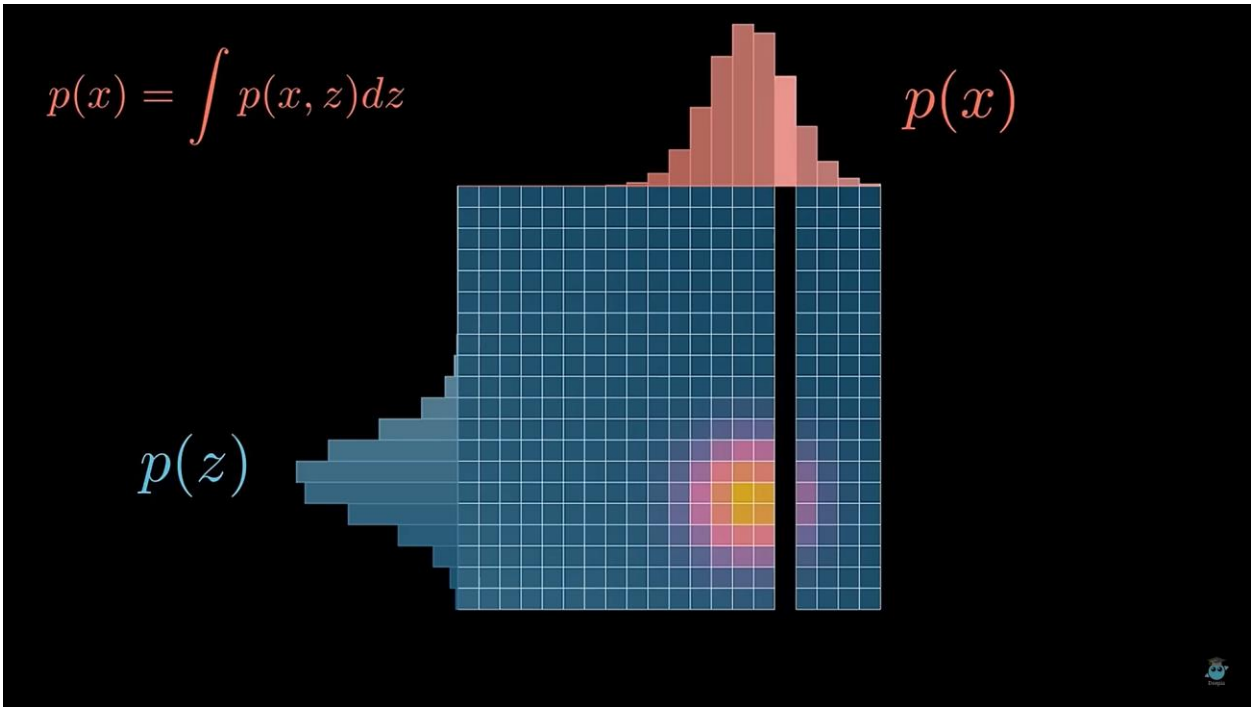
Interesting property of joint distribution is to find p(x) and p(z) called the marginal distribution.



Example define the find the marginal distribution p(x) with respect to other variable z by integrating the joint distrubion of p(x,z) with respect to dz

P(x=3) = find the joint probabity over the all value of z

.



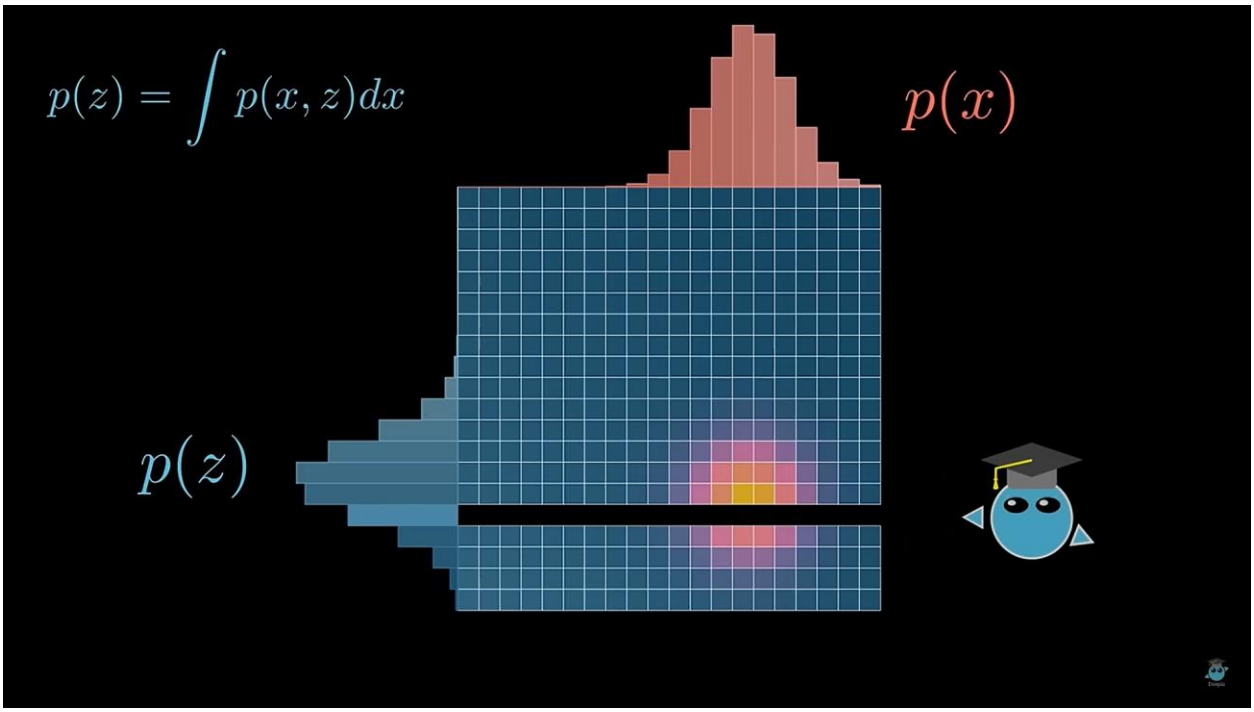
[show examples of joint distribution.

P(X) = sum the joint probability over all the possible value of Z.

If X = {H, T}, and Z = {1,2,3,4,5,6}

P(X)= P(x=H) = P(X= H)* P(Y = 1) + P(X= H)* P(Y = 2) + P(X= H)* P(Y = 3) + P(X= H)* P(Y = 6)

P(X) = 1/2 *1/6 + 1/2 *1/6 + 1/2 *1/6 + 1/2 *1/6 + 1/2 *1/6 + 1/2 *1/6 = 6*1/12 = 1/2

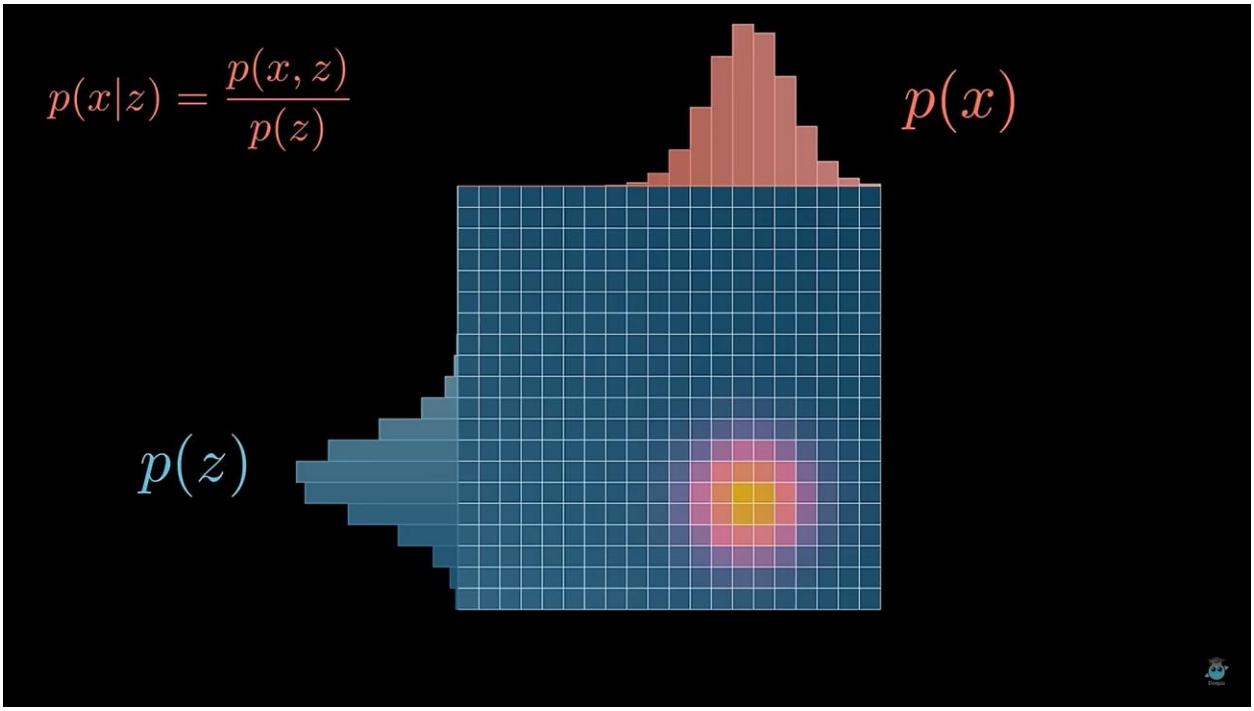


P(Z) = sum the joint probability over all the possible value of x.

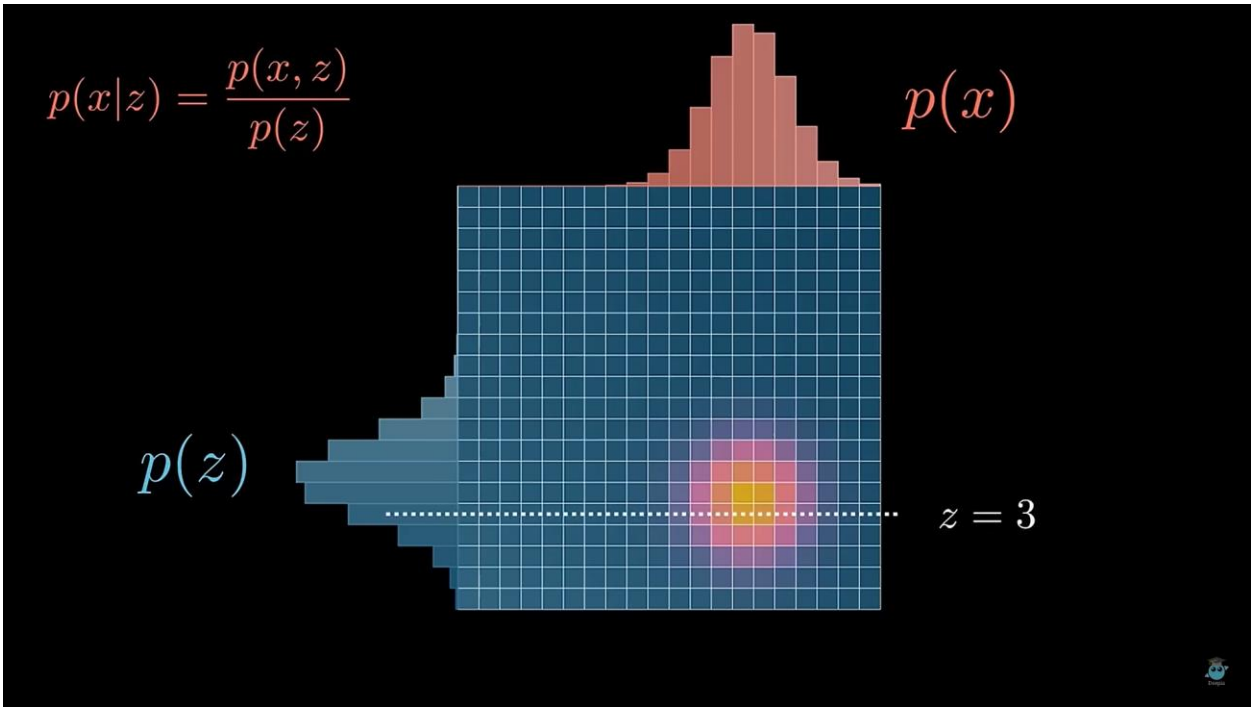
If X = {H, T}, and Z = {1,2,3,4,5,6}

P(Z)= P(z=1) = P(X= H)* P(z = 1) + P(X= T)* P(Y = 1) = ½* 1/6 + ½* 1/6 = 1/6

Conditional probability: it is defined as the joint distribution and normalizing it with the marginal distribution.



It is the slice of joint distribution and normalized it with the marginal distribution.



It is slice of joint probability for fix z= 3, p dividing

P(Z) = sum the joint probability over all the possible value of x.

If X = {H, T}, and Z = {1,2,3,4,5,6}

P(Z)= P(z=1) = P(X= H)* P(z = 1) + P(X= T)* P(Y = 1) = ½* 1/6 + ½* 1/6 = 1/6

As P(x /z = 1) = p(x, z=1)/ p(z=1)

p(x=H, z=1) = p(x=H)*p(z=1) = ½* 1/6 = 1/12

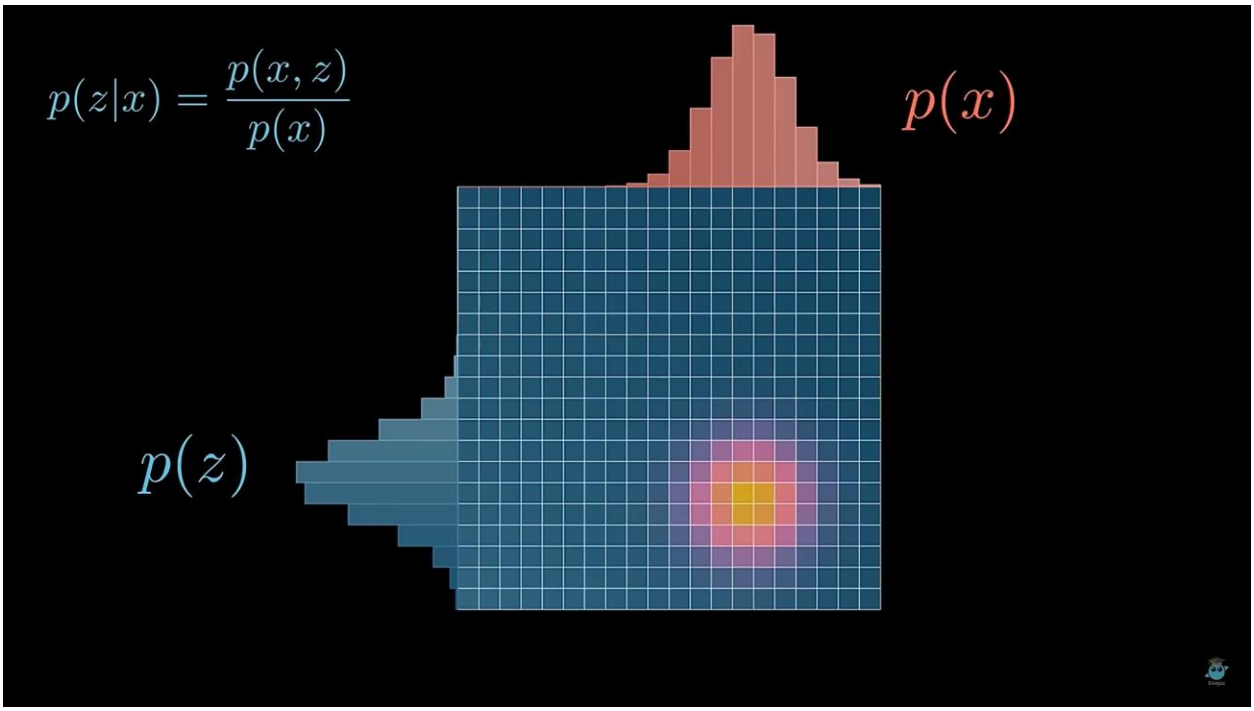
p(x=T, z=1) = p(x=T)*p(z=1) = ½* 1/6 = 1/12

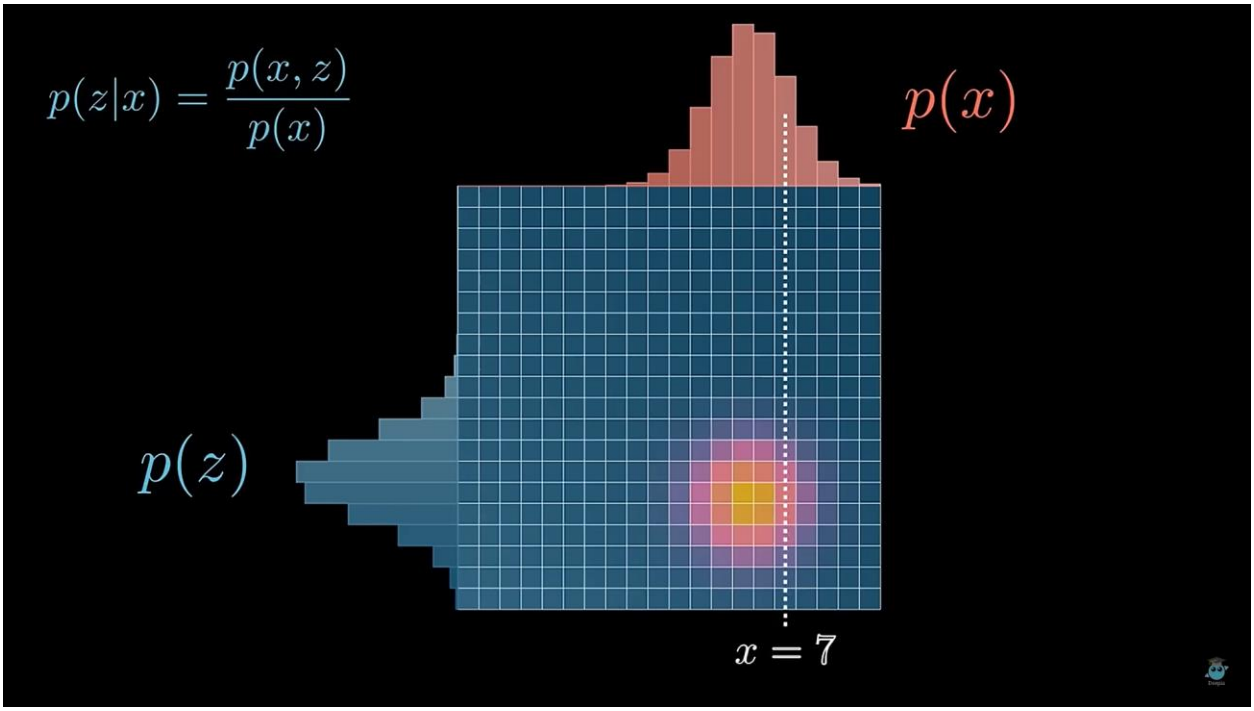
therefore

p(x=H/Z=1) = p(x =H, z =1) / p(z=1) = 1/12 *6 = ½

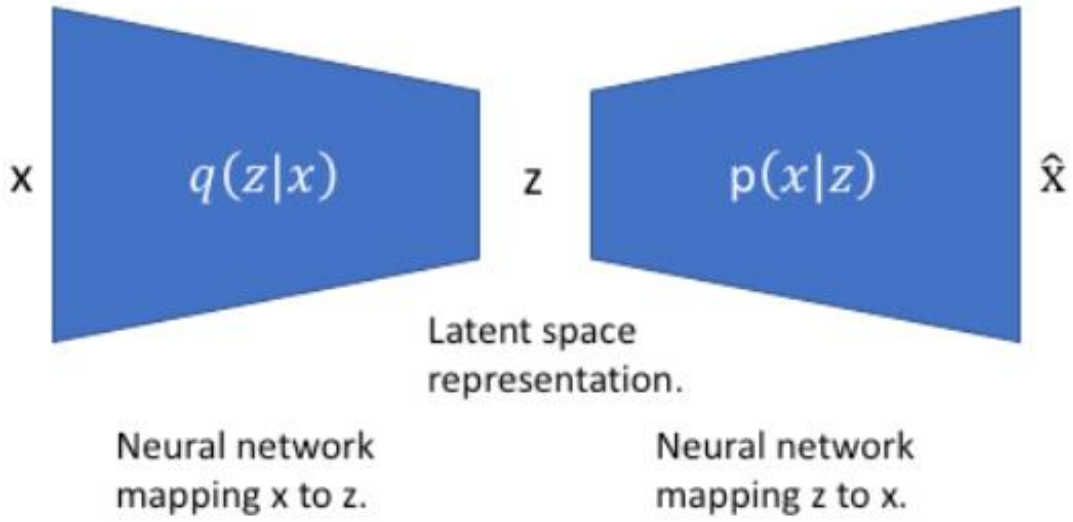
p(x=T/Z=1) = p(x =T, z =1) / p(z=1) = 1/12 *6 = ½

This makes sense because the coin toss is independent of the die roll, and thus the outcomes of the coin toss (H or T) remain equally likely even when we know Z=3.





Show the presence of z when we sample the data from x.



In the case of a Variational Autoencoder (VAE), **directly finding P(Z|X) using the formula** P(Z|X)=P(X,Z)/P(X) is extremely difficult due to several reasons:

Why $P(Z|X) = \frac{P(X,Z)}{P(X)}$ Is Hard to Compute in VAEs

1. **Marginal Probability $P(X)$ is Intractable:**
   - The denominator $P(X)$ requires integrating over all possible values of the latent variable Z, i.e.:

$$P(X) = \int P(X, Z)\, dZ.$$

   - If Z is high-dimensional (e.g., a 10-dimensional latent space), the integration becomes computationally infeasible due to the **curse of dimensionality**.
2. **Complex Form of $P(X, Z)$:**
   - In a VAE, $P(X, Z) = P(Z)P(X|Z)$, where:
     - $P(Z)$: The prior on the latent variable Z, typically a multivariate Gaussian.
     - $P(X|Z)$: The likelihood, often modeled by a complex neural network (the decoder in the VAE).
   - Computing $P(X|Z)$ involves evaluating the decoder network for each possible value of Z, which is computationally expensive.

The Problem Setup:

1. We are trying to compute the marginal probability $P(X)$:

$$P(X) = \int P(X, Z)\, dZ = \int P(Z)P(X|Z)\, dZ.$$

2. Here:
   - $P(X|Z)$: The likelihood of observing X given latent variables Z. This is usually a complex function, often modeled with a neural network.
   - $P(Z)$: The prior over Z. In VAEs, this is typically a multivariate Gaussian distribution, e.g., $\mathcal{N}(0, I)$.
   - dZ: The integration must consider all possible values of Z.
3. **Dimension of Z:**
   - If Z is 2-dimensional ($Z = (Z_1, Z_2)$), then the integral becomes:

$$P(X) = \int_{Z_1} \int_{Z_2} P(Z_1, Z_2)P(X|Z_1, Z_2)\, dZ_1\, dZ_2.$$

4. **Why We Use Numerical Integration:**
   - For many models, especially in deep learning, the likelihood $P(X|Z)$ is not an analytically simple function, and closed-form solutions to this integral are not possible.
   - Thus, we approximate the integral numerically by discretizing the continuous space of Z_1 and Z_2 into a grid.

How Grid-Based Numerical Integration Works

Discretizing Z_1 and Z_2:

- Imagine Z_1 and Z_2 are continuous variables in the range $[a, b]$.
- To compute the integral, we divide the range of each variable into n discrete points:

$$\{z_1^{(1)}, z_1^{(2)}, \ldots, z_1^{(n)}\} \quad \text{and} \quad \{z_2^{(1)}, z_2^{(2)}, \ldots, z_2^{(n)}\}.$$

- Now, we evaluate the integrand $P(X|Z)P(Z)$ at each of the $n \times n$ combinations of (Z_1, Z_2).

Numerical Approximation of the Integral:

- The integral is approximated as:

$$P(X) \approx \sum_{i=1}^{n} \sum_{j=1}^{n} P(X|Z_1^{(i)}, Z_2^{(j)})P(Z_1^{(i)}, Z_2^{(j)})\Delta Z_1 \Delta Z_2,$$

  where ΔZ_1 and ΔZ_2 are the widths of the discretized intervals.

Computational Cost:

- For every combination of $Z_1^{(i)}$ and $Z_2^{(j)}$, we evaluate $P(X|Z)P(Z)$.
- The total number of evaluations required is $n \times n = O(n^2)$.

Toy Example

Suppose:

- Z_1 and Z_2 range from $[0, 1]$.
- Divide each into $n = 10$ points: $\{0.0, 0.1, 0.2, \ldots, 1.0\}$.

To Compute $P(X)$:

- For each of the $n^2 = 10 \times 10 = 100$ grid points (Z_1, Z_2), evaluate:
  - $P(X|Z_1, Z_2)$: Likelihood at each grid point.
  - $P(Z_1, Z_2)$: Prior at each grid point.
- Sum over all 100 combinations to approximate $P(X)$.

Scaling Issue:

- If Z were 10-dimensional $(Z_1, Z_2, \ldots, Z_{10})$:
  - $n^{10} = 10^{10} = 10,000,000,000$ evaluations would be needed.

Problem Setup

Suppose we want to compute:

$$I = \int_0^1 f(z_1)\, dz_1,$$

where $f(z_1) = z_1^2 + 2z_1 + 1$.

Step-by-Step Solution

1. Divide the Range into Intervals

For $n = 5$, the interval length is:

$$\Delta z_1 = \frac{1}{n} = \frac{1}{5} = 0.2.$$

The interval midpoints for z_1 are:

$$z_1 = \{0.1, 0.3, 0.5, 0.7, 0.9\}.$$

2. Compute $f(z_1)$ at Each Midpoint

Evaluate $f(z_1)$ at each midpoint:

$$f(0.1) = 0.1^2 + 2(0.1) + 1 = 1.21,$$

$$f(0.3) = 0.3^2 + 2(0.3) + 1 = 1.69,$$

$$f(0.5) = 0.5^2 + 2(0.5) + 1 = 2.25,$$

$$f(0.7) = 0.7^2 + 2(0.7) + 1 = 2.89,$$

$$f(0.9) = 0.9^2 + 2(0.9) + 1 = 3.61.$$

3. Apply the Midpoint Rule

The midpoint rule approximates the integral as:

$$I \approx \Delta z_1 \cdot \sum_{i=1}^{n} f(z_1^{(i)}).$$

Substitute the values:

$$I \approx 0.2 \cdot (1.21 + 1.69 + 2.25 + 2.89 + 3.61)\,.$$

4. Simplify the Calculation

$$I \approx 0.2 \cdot 11.65 = 2.33.$$

As n=5, there are 5 computation are needed inorder to solve the single integral, if n= 10, then there are 10 computation are required to compute.

4. Toy Example to Compute Marginal Probability

Problem Setup:

- Let $Z = (z_1, z_2)$ be 2-dimensional latent variables, with both $z_1, z_2 \in [0, 1]$.
- Assume the prior distribution is uniform: $P(Z) = 1$ for $Z \in [0, 1]$
- The likelihood $P(X|Z) = \exp(-(z_1 + z_2))$, representing some simple relationship between X and Z.
- The marginal $P(X)$ is:

$$P(X) = \int_0^1 \int_0^1 P(X|Z)P(Z)\, dz_1 dz_2 = \int_0^1 \int_0^1 \exp(-(z_1 + z_2))\, dz_1 dz_2.$$

Solution:

Numerical integration for this 2D integral:

- Divide $[0, 1]$ for z_1 and z_2 into n points (e.g., $n = 10$).
- Sample grid points: $z_1, z_2 = [0, 0.1, 0.2, \ldots, 1.0]$.
- Evaluate $f(z_1, z_2) = \exp(-(z_1 + z_2))$ at all n^2 grid points.

For $n = 10$:

- We compute $f(z_1, z_2)$ at $10 \times 10 = 100$ points.
- Sum up all values, weighted by the grid spacing ($\Delta z_1 \cdot \Delta z_2 = 0.1 \cdot 0.1 = 0.01$).

This process becomes increasingly expensive for higher dimensions.

Now we Z is two dimentional and if n= 10, then there are 10x10= 10^2 computation are required to compute.

In practiacally Z is 10 to 100 dimensional, so it means 10^10 to 10^100 computation are required to compute, which is very expensive. So it seem interactable

As the number of dimensions (or features/**Z**'s) increases, the computational and data requirements grow exponentiall

https://www.jeremyjordan.me/variational-autoencoders/

Variational Inference:

Variational Inference (VI): Addresses the challenge of approximating the intractable posterior distribution $P_\theta(z|x)$ by using a simpler, more manageable distribution $q_\phi(z|x)$, often chosen as a Gaussian distribution.

The goal is to make $q_\phi(z|x)$ closely approximate $P_\theta(z|x)$. This is achieved by adjusting the parameters of $q_\phi(z|x)$ (e.g., its mean and variance) to minimize the discrepancy between the two distributions.

To measure this discrepancy, the Kullback-Leibler (KL) divergence $\mathrm{KL}(q_\phi(z|x)\|P_\theta(z|x))$ is minimized, effectively making $q_\phi(z|x)$ as close as possible to $P_\theta(z|x)$.

$$
\begin{aligned}
& D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})) \\
&= \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\
&= \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x}) p_\theta(\mathbf{x})}{p_\theta(\mathbf{z},\mathbf{x})} d\mathbf{z} && ; \text{Because } p(z|x)=p(z,x)/p(x) \\
&= \int q_\phi(\mathbf{z}|\mathbf{x}) \big(\log p_\theta(\mathbf{x}) + \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z},\mathbf{x})} \big) d\mathbf{z} \\
&= \log p_\theta(\mathbf{x}) + \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z},\mathbf{x})} d\mathbf{z} && ; \text{Because } \textstyle\int q(z|x)dz=1 \\
&= \log p_\theta(\mathbf{x}) + \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{x}|\mathbf{z}) p_\theta(\mathbf{z})} d\mathbf{z} && ; \text{Because } p(z,x)=p(x|z)p(z) \\
&= \log p_\theta(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} [\log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z})} - \log p_\theta(\mathbf{x}|\mathbf{z})] \\
&= \log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})) - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z})
\end{aligned}
$$

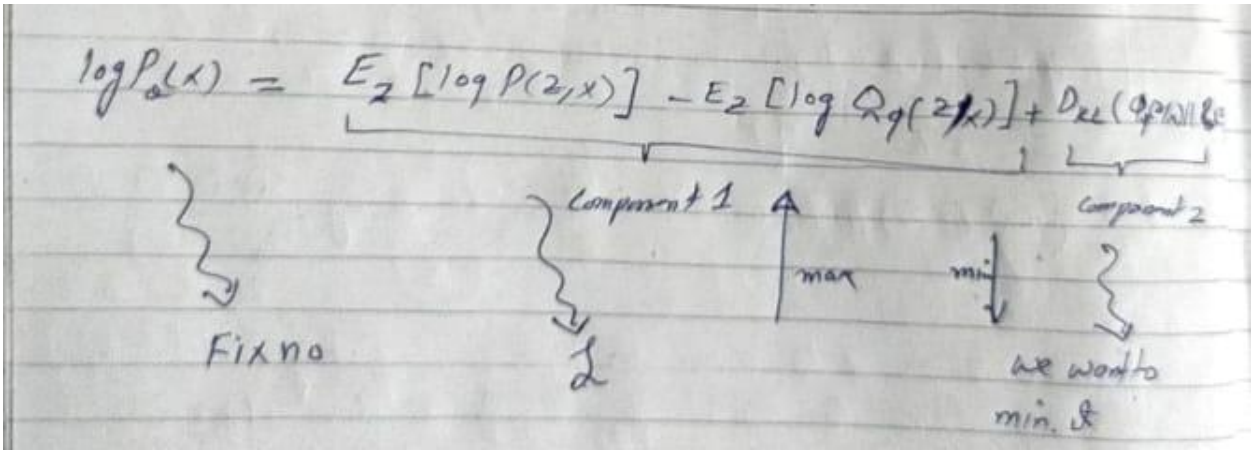
So we have:

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})) = \log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})) - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z})$$

Once rearrange the left and right hand side of the equation,

$$\log p_\theta(\mathbf{x}) - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}))$$

The LHS of the equation is exactly what we want to maximize when learning the true distributions: we want to maximize the (log-)likelihood of generating real data (that is $\log p_\theta(\mathbf{x})$) and also minimize the difference between the real and estimated posterior distributions (the term D_{KL} works like a regularizer). Note that $p_\theta(\mathbf{x})$ is fixed with respect to q_ϕ.



Instead of optimizaing the LHS of the original loss function, we optimize the RHS, evidance lower bound (ELBO), which avoid th eneed o to compute $\mathbf{P_\theta(z|x)}$

The negation of the above defines our loss function:

$$
\begin{aligned}
L_{\text{VAE}}(\theta, \phi) &= -\log p_\theta(\mathbf{x}) + D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x})) \\
&= -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) + D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) \\
\theta^*, \phi^* &= \arg\min_{\theta, \phi} L_{\text{VAE}}
\end{aligned}
$$

In Variational Bayesian methods, this loss function is known as the *variational lower bound*, or *evidence lower bound*. The "lower bound" part in the name comes from the fact that KL divergence is always non-negative and thus $-L_{\text{VAE}}$ is the lower bound of $\log p_\theta(\mathbf{x})$.

$$
-L_{\text{VAE}} = \log p_\theta(\mathbf{x}) - D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x})) \leq \log p_\theta(\mathbf{x})
$$

Therefore by minimizing the loss, we are maximizing the lower bound of the probability of generating real data samples.

$$\log p_\theta(\boldsymbol{x}) = \underbrace{E_{q_\varphi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\varphi(\boldsymbol{z}|\boldsymbol{x})}\right]}_{\textbf{ELBO}} + \underbrace{D_{KL}\left(q_\varphi(\boldsymbol{z}|\boldsymbol{x}) \| p_\theta(\boldsymbol{z}|\boldsymbol{x})\right)}_{\geq 0}$$

ELBO = Evidence Lower Bound

$$\text{Total Compensation} = \text{Base Salary} + \underbrace{\text{Bonus}}_{\geq 0}$$

We can for sure deduce the following:

$$\text{Total Compensation} \geq \text{Base Salary}$$

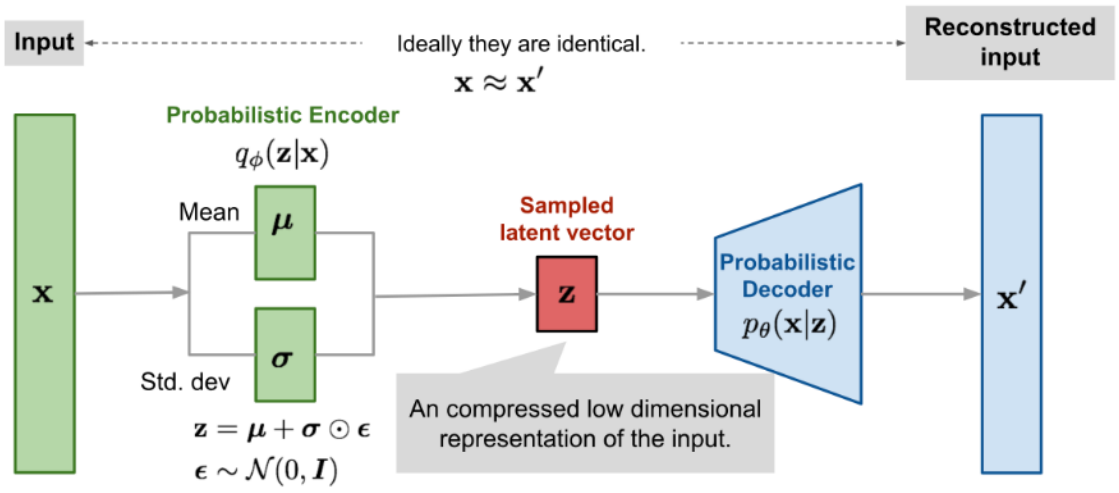
$$\log p_\theta(\boldsymbol{x}) \geq E_{q_\varphi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\varphi(\boldsymbol{z}|\boldsymbol{x})}\right]$$

Umar Jamil

Today lecture:

1. Intuition of loss function

2. # Derivation KL Divergence for multivariate GD



$$L_{\text{VAE}}(\theta, \phi) = -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) + D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}))$$

To illustrate the effects of using only reconstruction loss, only KL divergence loss, or both in a Variational Autoencoder (VAE), we can consider how each scenario influences the learning process and the quality of the generated outputs.

1. **Only Reconstruction Loss [Act like auto encoder]**

When training a VAE using only reconstruction loss, the model focuses solely on minimizing the difference between the original input and its reconstruction. This can lead to:

- **Overfitting**: The model may memorize the training data without learning a meaningful latent representation.
- **Poor Generalization**: The latent space may not be structured, making it difficult to generate new samples that resemble the training data.
- **Loss Function**: The loss function would look like this:

Total Loss=Reconstruction Loss

2. **Only KL Divergence Loss**

Using only KL divergence loss encourages the model to shape the latent space to resemble a standard normal distribution but ignores how well it reconstructs the input data. This leads to:

- **Underfitting**: The model may not learn to represent the input data effectively.
- **Poor Reconstruction Quality**: Outputs may be random or nonsensical since there is no penalty for poor reconstructions.
- **Loss Function**: The loss function would look like this:

Total Loss=KL Divergence

3. **Both Reconstruction and KL Divergence Loss**

When both losses are combined, the VAE achieves a balance between accurately reconstructing inputs and maintaining a structured latent space. This leads to:

- **Better Generalization**: The model learns useful representations that can generate new data similar to the training set.
- **Controlled Latent Space**: The latent variables are regularized to follow a Gaussian distribution, allowing for smooth interpolation between points in latent space.
- **Loss Function**: The total loss function is formulated as:

Total Loss=Reconstruction Loss+KL Divergence

2. Derivation KL of multivariate GD

Univariate Gaussian: Lets we have a single/ univariate random variable X, height of students in a class, then its gaussian distribution can be define using the following formula

General Formulation of Univariate Gaussian Distribution

The probability density function (pdf) of a univariate Gaussian distribution is given by:

$$p(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

X ~ N (0, 1),

$$p(x) = \frac{1}{\sqrt{2\pi(1)}} e^{-\frac{1}{2}\left(\frac{x-0}{1}\right)^2}$$

This simplifies to:

$$p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$$

- Gaussian or normal distribution, 1D

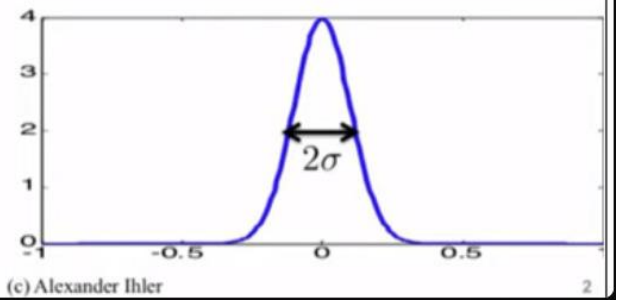
$$\mathcal{N}(x\ ;\ \mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2}(x-\mu)^2/\sigma^2\right]$$

  - Parameters: mean μ, variance σ^2 (standard deviation σ)

Maximum Likelihood estimates

$$\hat{\mu} = \frac{1}{N}\sum_i x^{(i)}$$

$$\hat{\sigma}^2 = \frac{1}{N}\sum_i (x^{(i)} - \hat{\mu})^2$$



(c) Alexander Ihler 2

Multivariate Gaussian:

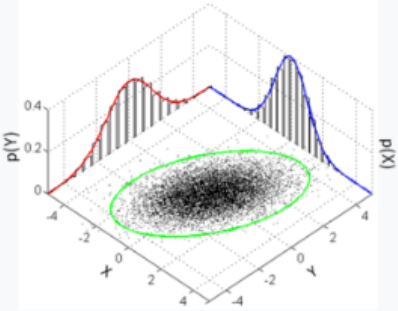
- Flatten image X is example multivariate random variable that have of multivariate probability function, where every pixel is a random variable that have a sample space of [0 to 255]

$$p(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

Multivariate Gaussian: Let $\mathbf{X} \sim \mathcal{N}\left(\begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1 & 0.5\\0.5 & 2\end{bmatrix}\right)$. Joint PDF is:

$$p(\mathbf{X}) = \frac{1}{2\pi\sqrt{1.75}} \exp\left(-\frac{1}{2}\begin{bmatrix}X_1\\X_2-1\end{bmatrix}^\top \begin{bmatrix}1 & -0.5\\-0.5 & 2\end{bmatrix}\begin{bmatrix}X_1\\X_2-1\end{bmatrix}\right)$$

Multivariate normal

Probability density function

Many sample points from a multivariate normal distribution with $\boldsymbol{\mu} = \begin{bmatrix}0\\0\end{bmatrix}$ and $\boldsymbol{\Sigma} = \begin{bmatrix}1 & 3/5\\3/5 & 2\end{bmatrix}$, shown along with the 3-sigma ellipse, the two marginal distributions, and the two 1-d histograms.

Notation	$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
Parameters	$\boldsymbol{\mu} \in \mathbf{R}^k$ — location $\boldsymbol{\Sigma} \in \mathbf{R}^{k \times k}$ — covariance (positive semi-definite matrix)
Support	$\boldsymbol{x} \in \boldsymbol{\mu} + \operatorname{span}(\boldsymbol{\Sigma}) \subseteq \mathbf{R}^k$
PDF	$(2\pi)^{-k/2}\det(\boldsymbol{\Sigma})^{-1/2}\ \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$, exists only when $\boldsymbol{\Sigma}$ is positive-definite

Variance vs Covariance matrix:

4. Example for $\mathbf{X} = [2, 4, 6]^T$:

(a) Compute Mean:

$$\bar{x} = \frac{1}{3}(2 + 4 + 6) = 4$$

(b) Subtract Mean:

$$\mathbf{X} - \bar{x} = \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix} - 4 = \begin{bmatrix} -2 \\ 0 \\ 2 \end{bmatrix}$$

(c) Compute Variance:

$$\text{Var}(X) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 = \frac{1}{3-1} \left[(-2)^2 + 0^2 + 2^2 \right]$$

$$\text{Var}(X) = \frac{1}{2}(4 + 0 + 4) = \frac{8}{2} = 4$$

(d) Covariance Matrix (Single Variable Case):

The covariance matrix is just a 1×1 scalar:

$$\Sigma = [4]$$

5. For Multivariable Data Example:

If you had data $\mathbf{X} = \begin{bmatrix} 2 & 1 \\ 4 & 3 \\ 6 & 5 \end{bmatrix}$, representing two variables:

(a) Compute Column-wise Means:

$$\bar{\mathbf{X}} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

(b) Subtract Mean:

$$\mathbf{X} - \bar{\mathbf{X}} = \begin{bmatrix} 2 & 1 \\ 4 & 3 \\ 6 & 5 \end{bmatrix} - \begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} -2 & -2 \\ 0 & 0 \\ 2 & 2 \end{bmatrix}$$

(c) Compute Covariance Matrix:

$$\Sigma = \frac{1}{n-1}(\mathbf{X} - \bar{\mathbf{X}})^T(\mathbf{X} - \bar{\mathbf{X}})$$

For this example:

$$\Sigma = \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix}$$

This covariance matrix indicates high correlation between the two variables.

Derivation of KL divergence between the multivariate Gaussian distribution:

We want to calculate the KL divergence between two distributions:

$$p(x) \sim \mathcal{N}(\mu_1, \Sigma_1), \quad q(x) \sim \mathcal{N}(\mu_2, \Sigma_2).$$

The formula for KL divergence is:

$$D_{KL}(p(x)\|q(x)) = \int p(x) \log \frac{p(x)}{q(x)}\, dx.$$

Step 1: Write down the Gaussian PDFs

The probability density function (PDF) of a multivariate Gaussian distribution is:

$$p(x) = \frac{1}{(2\pi)^{k/2}|\Sigma_1|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right),$$

$$q(x) = \frac{1}{(2\pi)^{k/2}|\Sigma_2|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right),$$

where:

- k is the dimensionality of x,
- $|\Sigma_1|$ and $|\Sigma_2|$ are the determinants of the covariance matrices,
- $(x-\mu)^T\Sigma^{-1}(x-\mu)$ is the Mahalanobis distance.

The Mahalanobis Distance is a measure of distance between a point and a distribution. Unlike the Euclidean distance, it accounts for the correlations and variances of the data dimensions, making it especially useful for multivariate data.

Step 2: Compute the ratio $\frac{p(x)}{q(x)}$

The ratio of $p(x)$ and $q(x)$ is:

$$\frac{p(x)}{q(x)} = \frac{|\Sigma_2|^{1/2}}{|\Sigma_1|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1) + \frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right).$$

Taking the logarithm of the ratio:

$$\log\frac{p(x)}{q(x)} = \frac{1}{2}\log\frac{|\Sigma_2|}{|\Sigma_1|} - \frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1) + \frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2).$$

Step 3: Substitute into the KL divergence formula

Now substitute the expression for $\log\frac{p(x)}{q(x)}$ into the KL divergence formula:

$$D_{KL}(p(x)\|q(x)) = \int p(x)\left[\frac{1}{2}\log\frac{|\Sigma_2|}{|\Sigma_1|} - \frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1) + \frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right]dx.$$

Separate the terms of the integral into three parts:

$$D_{KL}(p(x)\|q(x)) = \frac{1}{2}\log\frac{|\Sigma_2|}{|\Sigma_1|} + I_1 - I_2,$$

where:

$$I_1 = \int p(x)\left[\frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right]dx,$$

$$I_2 = \int p(x)\left[\frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right]dx.$$

Step 4: Evaluate each term

(a) First Term: $\log \frac{|\Sigma_2|}{|\Sigma_1|}$

This term is constant and independent of x, so it directly contributes:

$$\frac{1}{2} \log \frac{|\Sigma_2|}{|\Sigma_1|}.$$

(b) Second Term: $\int p(x)(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)dx$

Using the expectation property of the Gaussian distribution:

$$\mathbb{E}_{p(x)}[(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)] = \text{tr}(\Sigma_1^{-1}\Sigma_1) = k,$$

where k is the dimensionality of the space.

Thus, the integral:

$$I_2 = \frac{1}{2}k.$$

(c) Third Term: $\int p(x)(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)dx$

Expand $(x-\mu_2)$ as $(x-\mu_1)+(\mu_1-\mu_2)$:

$$(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2) = (x-\mu_1)^T\Sigma_2^{-1}(x-\mu_1) + 2(x-\mu_1)^T\Sigma_2^{-1}(\mu_1-\mu_2) + (\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2).$$

Take the expectation under $p(x)$:

1. $\mathbb{E}_{p(x)}[(x-\mu_1)^T\Sigma_2^{-1}(x-\mu_1)] = \text{tr}(\Sigma_2^{-1}\Sigma_1)$,
2. $\mathbb{E}_{p(x)}[2(x-\mu_1)^T\Sigma_2^{-1}(\mu_1-\mu_2)] = 0$ (since $\mathbb{E}_{p(x)}[x-\mu_1]=0$),
3. $\mathbb{E}_{p(x)}[(\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)] = (\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)$.

Thus:

$$I_1 = \frac{1}{2}\left[\text{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)\right].$$

Step 5: Combine All Terms

Finally, substitute everything back:

$$D_{KL}(p(x)\|q(x)) = \frac{1}{2}\left[\log\frac{|\Sigma_2|}{|\Sigma_1|} - k + \text{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)\right].$$

This is the final result.

For KL Divergence for VAE:

To compute $D_{KL}[q(z|x)\|p(z)]$ for the VAE case using the given formulation:

$$D_{KL}(p(x)\|q(x)) = \frac{1}{2}\left[\log\frac{|\Sigma_2|}{|\Sigma_1|} - k + \operatorname{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)\right],$$

where:

- $\mu_1 = \mu$ and $\Sigma_1 = \Sigma$ come from $q(z|x)$,
- $\mu_2 = 0$ and $\Sigma_2 = I$ come from $p(z)$.

Step-by-Step Calculation

1. Log Determinant Term: $\log\frac{|\Sigma_2|}{|\Sigma_1|}$

- $\Sigma_2 = I$, so $|\Sigma_2| = 1$,
- $|\Sigma_1| = |\Sigma|$,

$$\log\frac{|\Sigma_2|}{|\Sigma_1|} = -\log|\Sigma|.$$

2. Trace Term: $\operatorname{tr}(\Sigma_2^{-1}\Sigma_1)$

- $\Sigma_2 = I$, so $\Sigma_2^{-1} = I$,
- Therefore, $\operatorname{tr}(\Sigma_2^{-1}\Sigma_1) = \operatorname{tr}(\Sigma_1) = \operatorname{tr}(\Sigma)$.

3. Quadratic Term: $(\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)$

- $\mu_1 = \mu$, $\mu_2 = 0$, and $\Sigma_2^{-1} = I$,
- So, $(\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2) = \mu^T I\mu = \mu^T\mu$.

4. Combine Terms

Plugging all these into the KL divergence formula:

$$D_{KL}[q(z|x)\|p(z)] = \frac{1}{2}\left[-\log|\Sigma| - k + \operatorname{tr}(\Sigma) + \mu^T\mu\right].$$

Final Result

$$D_{KL}[q(z|x)\|p(z)] = \frac{1}{2}\left[-\log|\Sigma| - k + \operatorname{tr}(\Sigma) + \|\mu\|^2\right].$$

This is the KL divergence between the approximate posterior $q(z|x)$ and the prior $p(z)$ in a VAE.

Now the final loss function of VAE:

Loss Function with the Computed KL Divergence

The VAE loss is:

$$\mathcal{L}_{\text{VAE}} = -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + D_{KL}[q_\phi(z|x)\|p(z)].$$

For D_{KL}, using the formula for two Gaussian distributions $q(z|x) = \mathcal{N}(\mu, \Sigma)$ and $p(z) = \mathcal{N}(0, I)$, we computed:

$$D_{KL}[q_\phi(z|x)\|p(z)] = \frac{1}{2}\left(-\log|\Sigma| - k + \text{tr}(\Sigma) + \|\mu\|^2\right).$$

Incorporating into the Loss

Substituting the KL divergence into the VAE loss:

$$\mathcal{L}_{\text{VAE}} = -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + \frac{1}{2}\left(-\log|\Sigma| - k + \text{tr}(\Sigma) + \|\mu\|^2\right).$$

1. **Reconstruction Loss** ($-\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$):
   - For Gaussian decoders, this term can often be approximated by the Mean Squared Error (MSE):
     $$\text{Reconstruction Loss} \approx \frac{1}{2}\|x - \hat{x}\|^2,$$
     where $\hat{x}$ is the reconstructed input.

2. **KL Divergence**:
   - From the computation:
     $$D_{KL}[q_\phi(z|x)\|p(z)] = \frac{1}{2}\left(-\log|\Sigma| - k + \text{tr}(\Sigma) + \|\mu\|^2\right).$$

Final Loss Function

The final loss function becomes:

$$\mathcal{L}_{\text{VAE}} = \frac{1}{2}\|x - \hat{x}\|^2 + \frac{1}{2}\left(-\log|\Sigma| - k + \text{tr}(\Sigma) + \|\mu\|^2\right).$$

Simplifications for Diagonal Covariance Matrix

If Σ is diagonal with entries $\sigma_1^2, \sigma_2^2, \ldots, \sigma_k^2$:

1. $\log |\Sigma| = \sum_{i=1}^{k} \log \sigma_i^2$,
2. $\operatorname{tr}(\Sigma) = \sum_{i=1}^{k} \sigma_i^2$,
3. The KL divergence simplifies to:

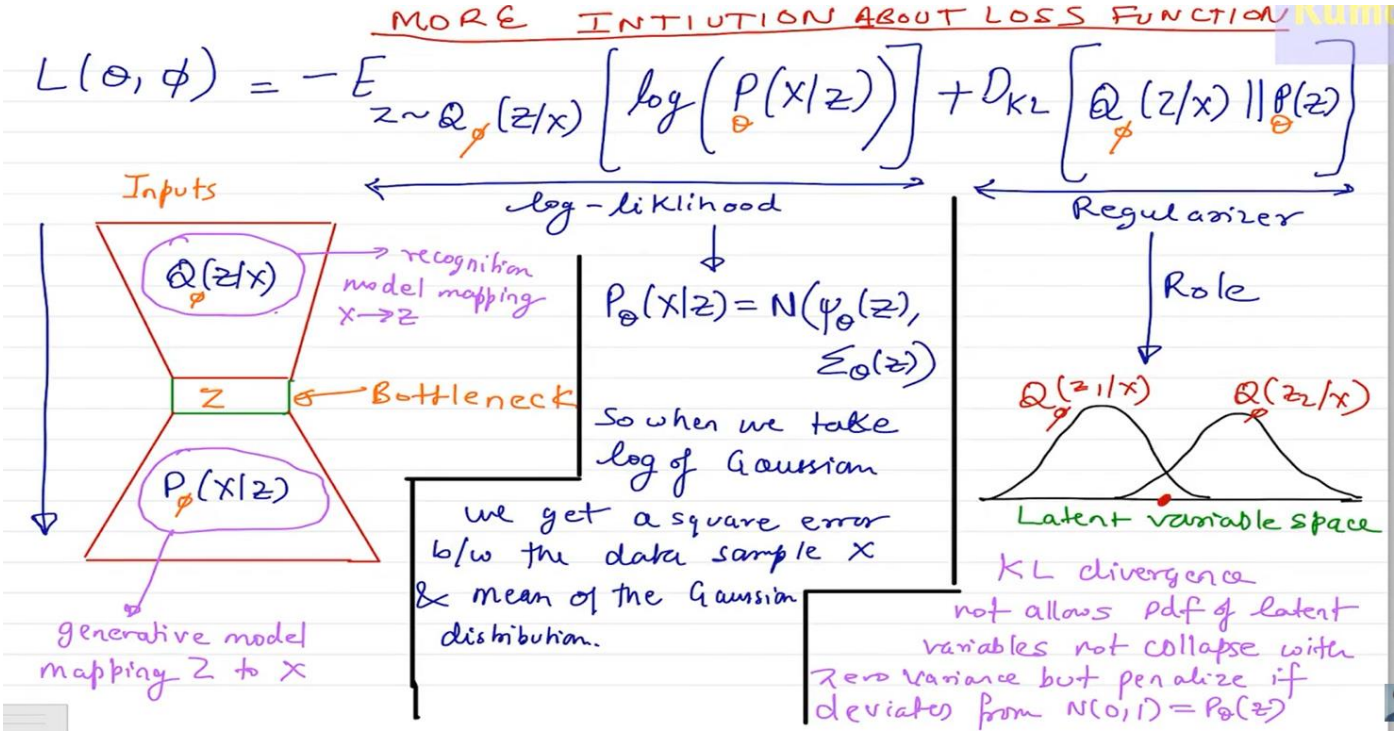
$$D_{KL} = \frac{1}{2} \sum_{i=1}^{k} \left(-\log \sigma_i^2 - 1 + \sigma_i^2 + \mu_i^2 \right).$$

Thus, the loss becomes:

$$\mathcal{L}_{\text{VAE}} = \frac{1}{2} \|x - \hat{x}\|^2 + \frac{1}{2} \sum_{i=1}^{k} \left(-\log \sigma_i^2 - 1 + \sigma_i^2 + \mu_i^2 \right).$$

Interpretation

1. **First term**: Ensures $\hat{x}$ (reconstructed input) is close to x (original input).
2. **Second term**: Regularizes the encoder by forcing $q_\phi(z|x)$ to align with the prior $p(z)$.



https://www.youtube.com/watch?v=w8F7_rQZxXk&list=PLdxQ7SoCLQANizknbIiHzL_hYjEaI-wUe

1. **Expression for $P_\theta(X|Z)$:**

The likelihood $P_\theta(X|Z)$ is modeled as a Gaussian distribution:

$$P_\theta(X|Z) = \frac{1}{\sqrt{(2\pi)^k |\Sigma_\theta(Z)|}} \exp\left(-\frac{1}{2}(X - \mu_\theta(Z))^T \Sigma_\theta(Z)^{-1}(X - \mu_\theta(Z))\right)$$

- $\mu_\theta(Z)$: Mean of the Gaussian, predicted by the decoder network.
- $\Sigma_\theta(Z)$: Covariance matrix of the Gaussian, often simplified to a diagonal matrix for computational efficiency.
- X: Input data.
- Z: Latent variable sampled from the encoder's posterior distribution.

2. **Log of the Likelihood:**

Taking the logarithm of $P_\theta(X|Z)$ simplifies the expression:

$$\log P_\theta(X|Z) \propto -\frac{1}{2}(X - \mu_\theta(Z))^T \Sigma_\theta(Z)^{-1}(X - \mu_\theta(Z))$$

This step eliminates constant terms (like $\sqrt{(2\pi)^k}$) since they do not affect optimization.

3. **Connection to Reconstruction Error:**

The term $(X - \mu_\theta(Z))^T \Sigma_\theta(Z)^{-1}(X - \mu_\theta(Z))$ represents the **Mahalanobis distance** between X and the predicted mean $\mu_\theta(Z)$, weighted by the inverse covariance matrix.

- If $\Sigma_\theta(Z)$ is the identity matrix or a scalar, this simplifies to a **squared reconstruction error**:

$$(X - \mu_\theta(Z))^T (X - \mu_\theta(Z))$$

This measures how well the decoder reconstructs the input X from the latent representation Z.