

Analisi di Immagini e Video (Computer Vision)

Giuseppe Manco

Outline

- Generative Modeling
 - Motivazioni
 - task
- Approcci
 - Variational Autoencoders
 - Generative Adversarial Networks

Crediti

- Slides adattate da altri corsi:
 - Ettore Ritacco (CS Unical)

Modelli Generativi

“What I cannot create, I do not understand.”

—Richard Feynman

Modelli Generativi

- Modelli probabilistici
 - Punto di partenza: x
 - Dato ad alta dimensionalità
 - Target: $P(x)$
 - Cattura la nozione d'incertezza sul dato
 - Densità
 - Obiettivo: generare (simulare) dati realistici
 - $x \sim P$
 - Strumento: parametrizzazione di P
 - $P(x) \approx P_\theta(x)$
 - $x \sim P_\theta$

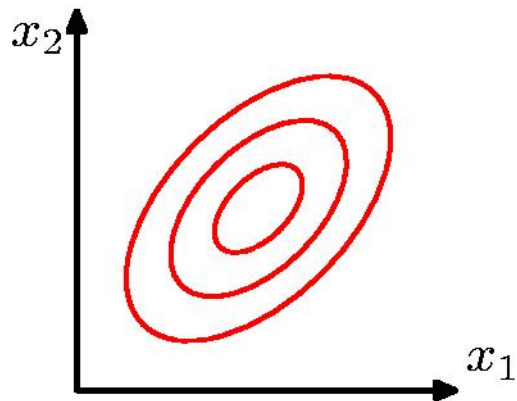
Modelli Generativi

- Tre componenti
 - Stima di parametri
 - Dati $\{x_1, \dots, x_n\}$ dove $x_i \sim P_{true}$
 - Trovare il θ ottimale
 - Il valore di θ per cui $P_\theta(x_i) \approx P_{true}(x_i)$
 - Inferenza
 - Calcolare $P(x|\theta)$
 - Generazione
 - Campionare $x \sim P(.|\theta)$

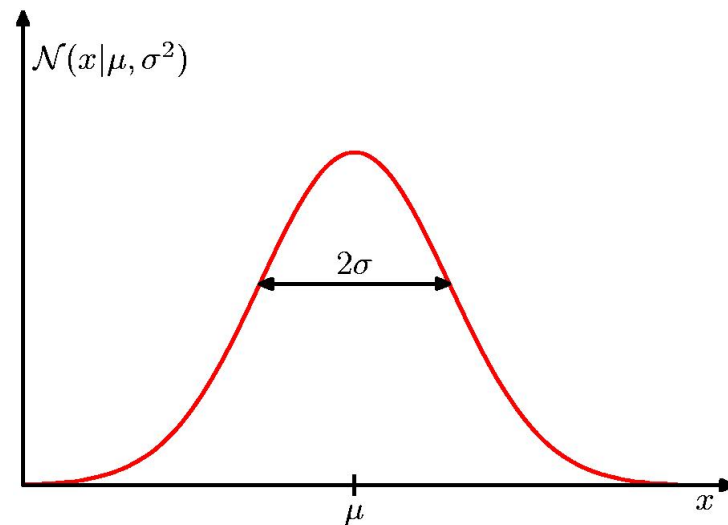
Esempio: Dati gaussiani

- $\theta = \{\mu, \Sigma\}$
- $x \sim \mathcal{N}(\cdot | \mu, \Sigma)$

$$\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$



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



Esempio: Dati gaussiani

- Stima

- Dati $\{x_1, \dots, x_n\}$, $\theta = \{\mu, \sigma\}$

- $\mu = \frac{1}{n} \sum_i x_i$

- $\sigma^2 = \frac{1}{n} \sum_i (x_i - \mu)^2$

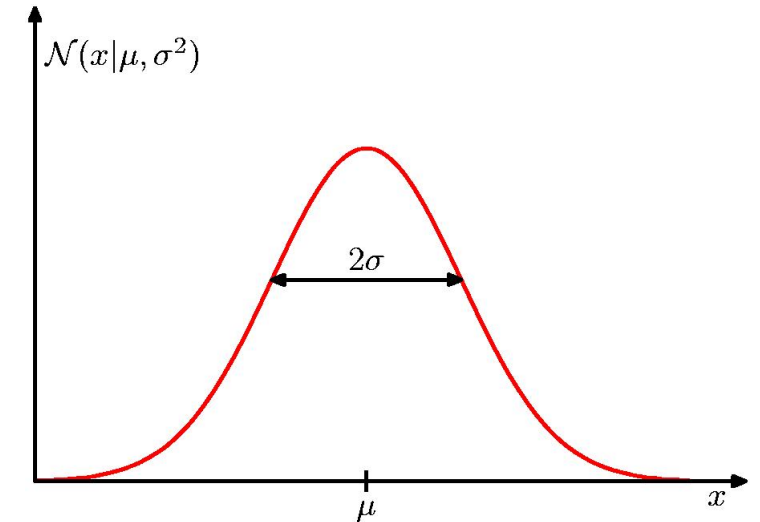
- Inferenza

- $p_\theta(x) = \mathcal{N}(x|\mu, \sigma)$

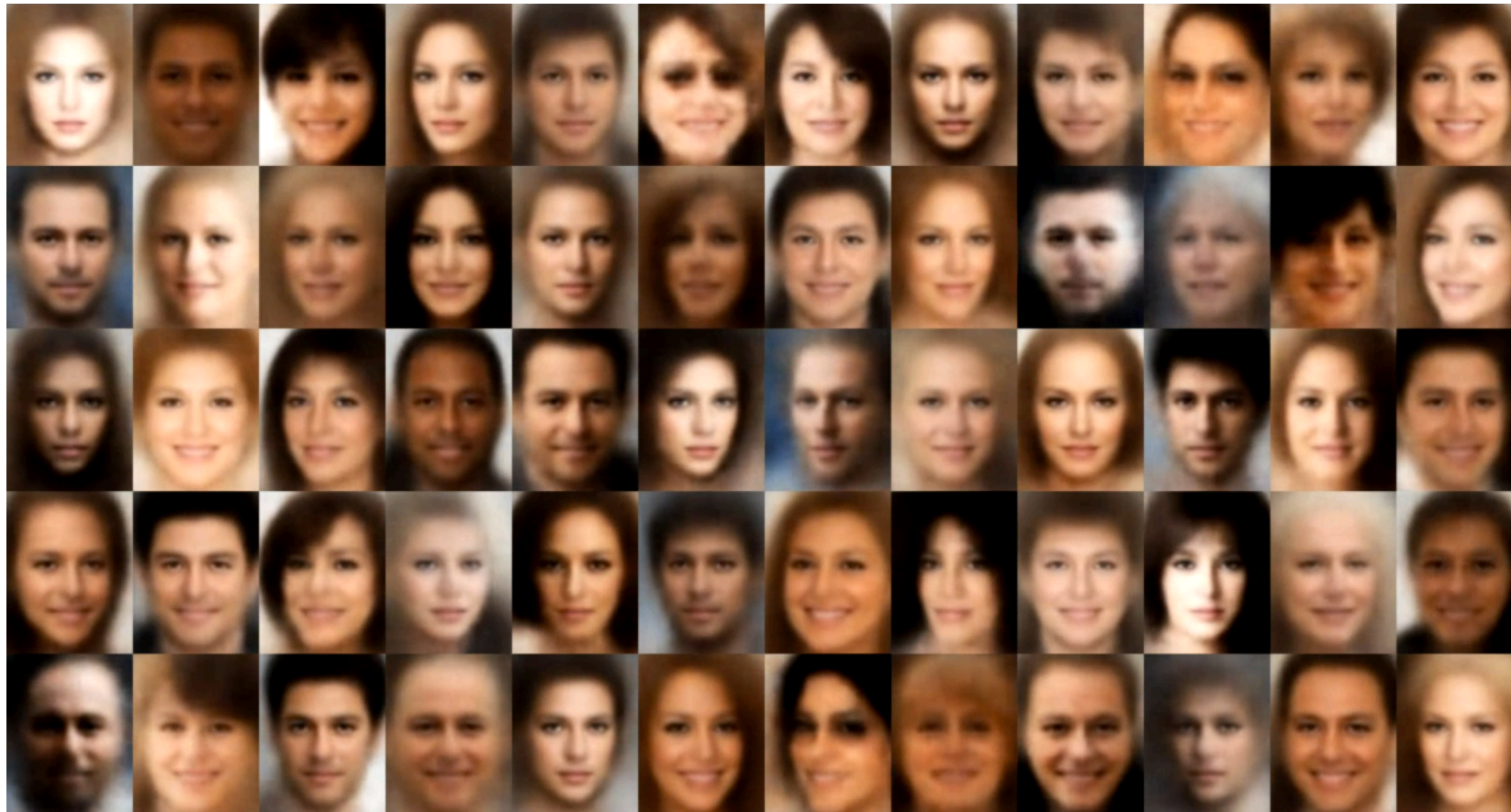
- Generazione

- $x \sim p_\theta(\cdot)$

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



Esempio: immagini

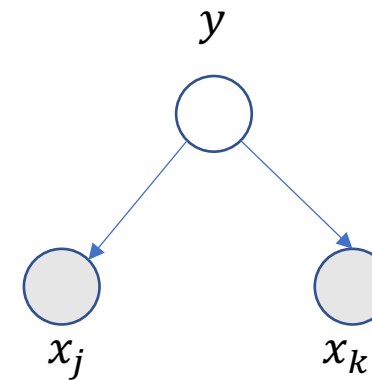
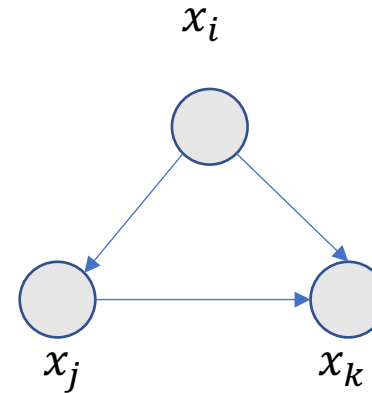


Perché i modelli generativi?

- Generazione di nuovi contenuti
 - Image completion
 - Data augmentation
- Image restoration
- Super-resolution
- Style transfer

Quali modelli generativi?

- Fully-observed models
 - Modellano le relazioni tra i dati
 - Variabili casuali
- Latent-Variable models
 - Variabili latenti associate ad ogni dato osservato
 - Esprimono causalità/correlazione nascosta con i dati osservati

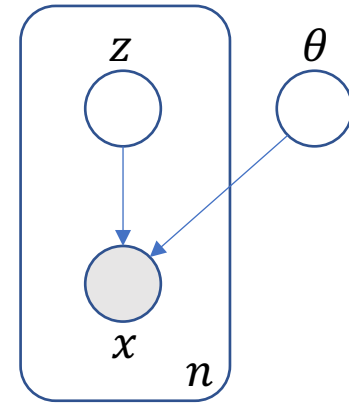


Variational Autoencoders

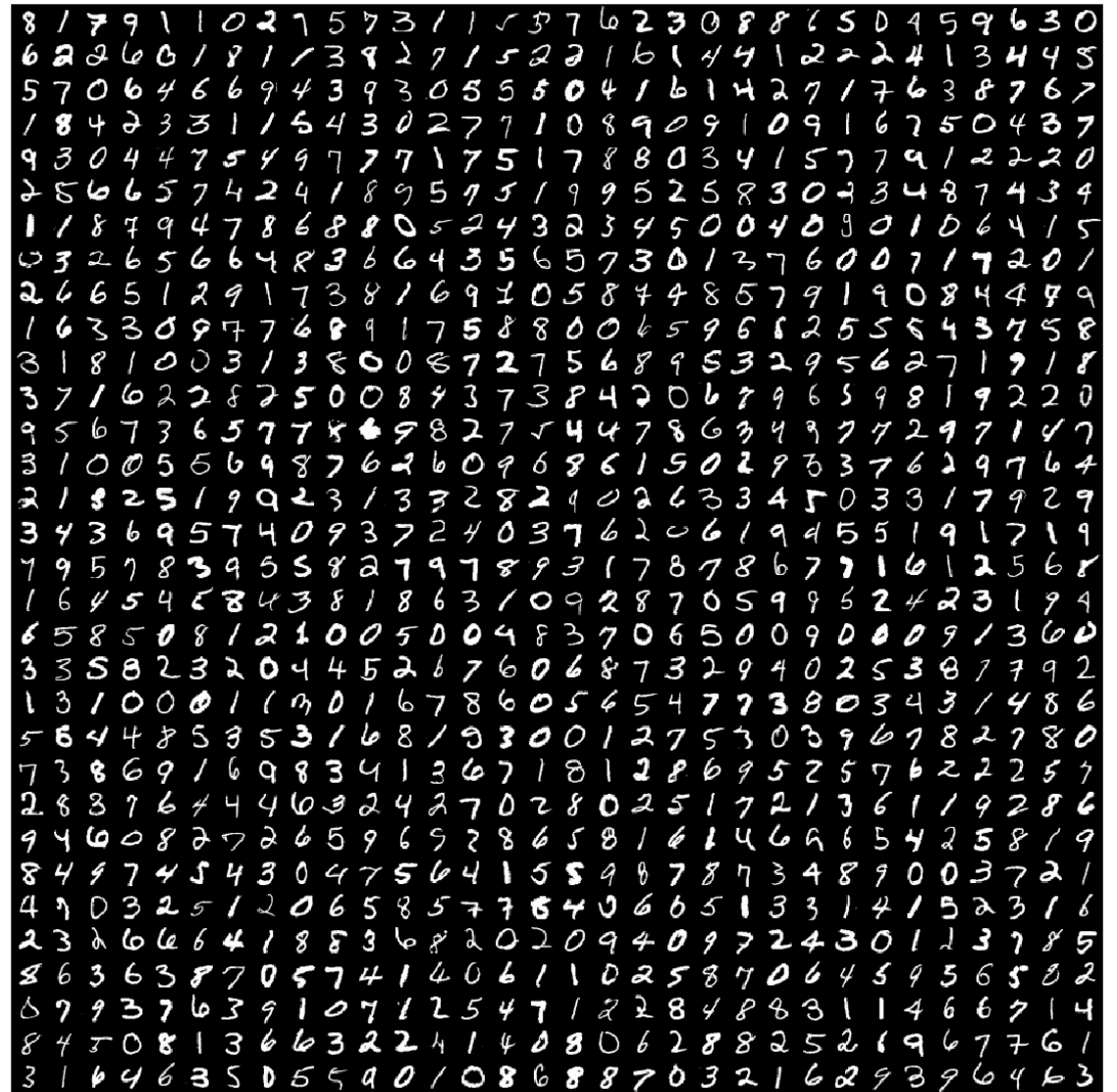
- Modello semplice
 - Assume che x sia generato condizionatamente ad una variabile latente
 - Influisce sul parametro θ

$$P(x) = \int P_{\theta}(x|z)P(z)dz$$

- Generazione facile
- Inferenza difficile...

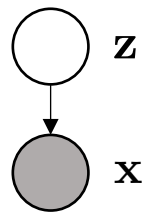


Esempio: MNIST



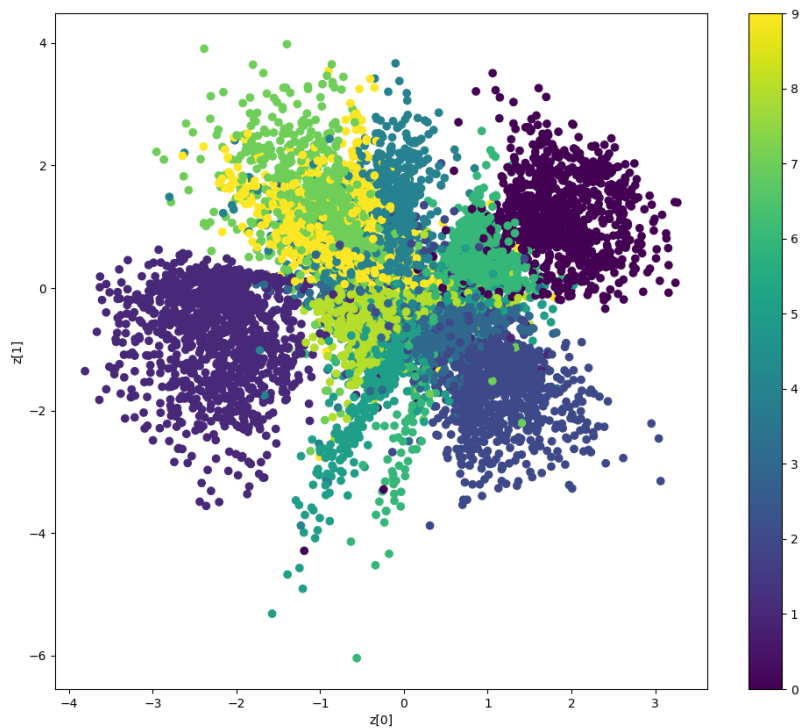
Generazione

- sorteggia $z \sim P_z$
- campiona $x \sim P_\theta(x|z)$

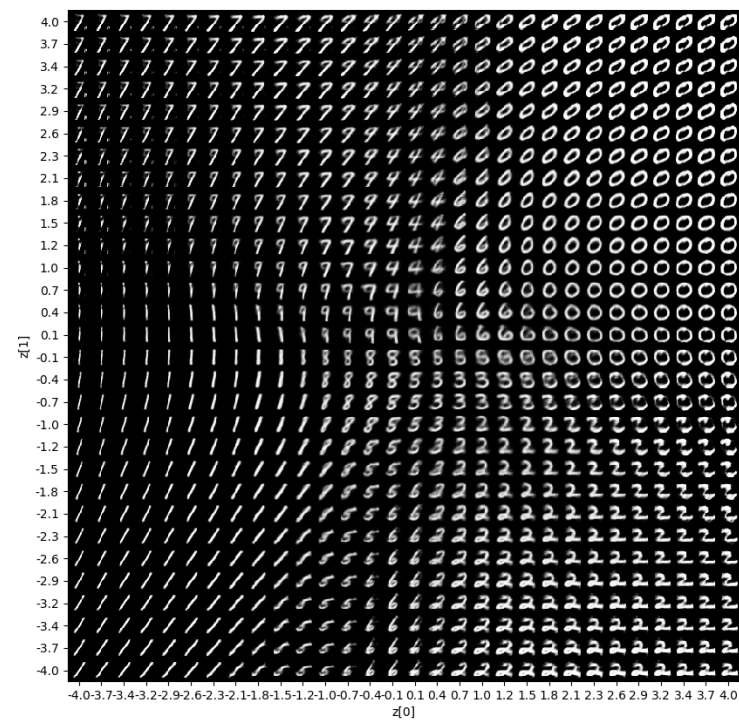


$$P(x) = \int P_{\theta}(x|z)P(z)dz$$

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$



$$\mathbf{x} \sim P_{\theta}(\mathbf{x}|\mathbf{z})$$



Quali parametri per MNIST?

- Bernoullian model

- $P_{\theta}(x|z) = \prod_{i,j} \theta_{ij}(z)^{x_{ij}} \left((1 - \theta_{ij}(z)) \right)^{1-x_{ij}}$

- $\theta_{ij}(z) \in [0,1]$ variabile bernoulliana che modella la probabilità bianco/nero

- Estensione: multinomial distribution

- $P_{\theta}(x|z) = \prod_{i,j} \theta_{ij,k}(z)$

- $\theta_{ij,k}(z)$ variabile multinomiale ($\sum_k \theta_{ijk}(z) = 1$) che modella la probabilità del colore k

- Gaussian model

- $P_{\theta}(x|z) = \prod_{ij} \mathcal{N}(x, \mu(z), \sigma(z)I)$

Inferenza

- Problema
 - Dato x , bisogna calcolare

$$P(x) = \int P_{\theta}(x|z)P(z)dz$$

- Intrattabile!

Inferenza

- **Approssimazione**

- **Metodi:**

- **Monte Carlo (MC) Sampling**

- Approssimiamo l'integrale campionando dei valori di z e facendo averaging

$$P(x) = \int P_{\theta}(x|z)P(z)dz \approx \sum_i P_{\theta}(x|z_i)P(z_i)$$

- **Variational Inference (VI)**

- Approssimiamo $P(x)$ in forma trattabile computazionalmente

Metodi Monte Carlo (MC)

- Una famiglia di metodi di simulazione per inferire variabili target
- Algoritmo generale
 - Define a set of constraints or desiderata goals related to the target variables
 - Generate data from a suitable probability distribution related to the target variables
 - Check and evaluate how many successful trials you get out of all the experiments
 - Estimate your target variables

Metodi Monte Carlo (MC)

- Esempio: Stima di π
 - In uno spazio euclideo 2D, genera valori random $(x_i, y_i)_{i \in \{1, \dots, n\}}$ con una distribuzione uniforme su entrambi gli assi
 - Calcola m , il numero di punti che si trovano all'interno della circonferenza di raggio 1
 - Devono soddisfare $\sqrt{x^2 + y^2} \leq 1$
 - restituisci $4 \cdot \frac{m}{n}$

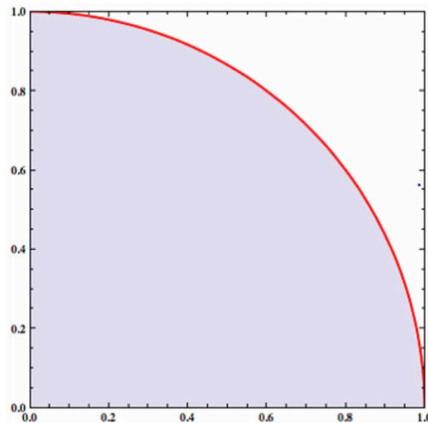
Metodi Monte Carlo (MC)

- Esempio: Stima di π
 - In uno spazio euclideo 2D, genera valori random $(x_i, y_i)_{i \in \{1, \dots, n\}}$ con una distribuzione uniforme su entrambi gli assi
 - Calcola m , il numero di punti che si trovano all'interno della circonferenza di raggio 1
 - Devono soddisfare $\sqrt{x^2 + y^2} \leq 1$
 - restituisci $4 \cdot \frac{m}{n}$

Che stiamo facendo?

Monte Carlo (MC)

- Esempio: Stima di π
 - In uno spazio euclideo 2D, genera valori random $(x_i, y_i)_{i \in \{1, \dots, n\}}$ con una distribuzione uniforme su entrambi gli assi
 - Calcola m , il numero di punti che si trovano all'interno della circonferenza di raggio 1
 - Devono soddisfare $\sqrt{x^2 + y^2} \leq 1$
 - restituisci $4 \cdot \frac{m}{n}$



Che stiamo facendo?

– stiamo approssimando quest'area

$$A = \frac{A_{sector}}{A_{square}} = A_{sector} = \frac{A_{circle}}{4} = \frac{\pi \cdot r^2}{4} = \frac{\pi}{4}$$

Metodi Monte Carlo (MC)

```
In [1]: import numpy as np

def compute_pi(n_trials):
    n_successes = 0

    for _ in range(n_trials):
        x = np.random.rand()
        y = np.random.rand()

        if np.sqrt(x**2 + y**2) <= 1.:
            n_successes += 1

    return 4 * n_successes / n_trials

print(compute_pi(int(1e6)))
print(np.pi)
```

3.14368

3.141592653589793

Metodi Monte Carlo (MC)

- Limitazioni:
 - Il campionamento deve essere “semplice”
 - Servono molti dati
 - Non c'è convergenza asintotica

Idea

- Utilizziamo una proposal distribution $q_\lambda(z|x)$
- Utilizziamo i valori di z campionati da drawn from q_λ

Inferenza Variazionale

- Approssimiamo la posterior $P(z|x)$ con una famiglia di distribuzioni "semplici" $q_\lambda(z|x)$
- Il parametro λ indicizza la famiglia di distribuzioni
 - Ad esempio, con q Gaussiana, $\lambda = \{\mu, \sigma\}$

$$P(z|x) = \frac{P(x|z) \cdot P(z)}{P(x)} \approx q_\lambda(z|x)$$

Inferenza Variazionale

- Due distribuzioni P e Q sono simili se la divergenza è bassa
- Divergenza di Kullback – Leibler:

$$\begin{aligned} KL(P||Q) &= \int_{-\infty}^{+\infty} p(\omega) \cdot \ln \frac{p(\omega)}{q(\omega)} d\omega \\ &= \int_{-\infty}^{+\infty} p(\omega) \cdot \ln p(\omega) d\omega - \int_{-\infty}^{+\infty} p(\omega) \cdot \ln q(\omega) d\omega \\ &= \mathbb{E}_{x \sim p} [\ln p(x)] - \mathbb{E}_{x \sim p} [\ln q(x)] \end{aligned}$$

Inferenza Variazionale

- Vogliamo una $q_\lambda(z|x)$ che sia una buona approssimazione di $p(z|x)$

$$\begin{aligned} KL(q_\lambda(z|x)||p(z|x)) &= \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln p(z|x)] \\ &= \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}\left[\ln \frac{p(z,x)}{p(x)}\right] \\ &= \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln p(z,x)] + \mathbb{E}_{q_\lambda(z|x)}[\ln p(x)] \\ &= \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln p(z,x)] + \ln p(x) \\ &= -\{\mathbb{E}_{q_\lambda(z|x)}[\ln p(z,x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)]\} + \ln p(x) \end{aligned}$$

ELBO (evidence lower bound)

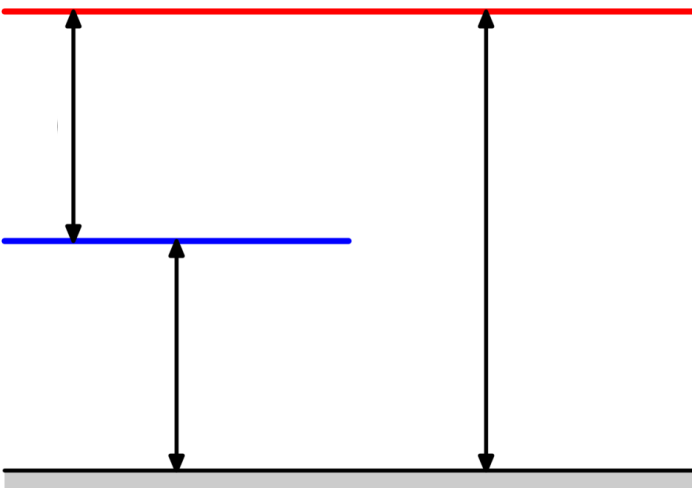
Log
likelihood

$$\begin{aligned}
\log P(x) &= \int q(z|x) \log P(x) dz \\
&= \int q(z|x) \log P(x) \frac{P(z|x)}{P(z|x)} dz \\
&= \int q(z|x) \log P(x) \frac{P(x|z)P(z)}{P(x)P(z|x)} dz \\
&= \int q(z|x) \log P(x|z)P(z) dz + \int q(z|x) \log \frac{q(z|x)}{q(z|x)P(z|x)} dz \\
&= \int q(z|x) \log P(x|z) dz + \int q(z|x) \log \frac{q(z|x)}{P(z|x)} dz \\
&\quad - \int q(z|x) \log \frac{q(z|x)}{P(z)} dz \\
&= E_{z \sim q} [\log P(x|z)] + KL[q(z|x) || P(z|x)] \\
&\quad - KL[q(z|x) || P(z)]
\end{aligned}$$

Inferenza variazionale

$$KL[q_\lambda(z|x) || p(z|x)] = -ELBO(q_\lambda) + \ln p(x)$$

- Scelto q_λ , si stimano i parametri λ che minimizzano l'ELBO



$$KL[q_\lambda(z|x) || p(z|x)] = E_{z \sim q_\lambda} [\log q_\lambda(z|x) - \log p(x|z)]$$

$$\begin{aligned} KL[q_\lambda(z|x)||p(z|x)] &= E_{z \sim q_\lambda} [\log q_\lambda(z|x) - \log p(x|z)] \\ &= KL[q_\lambda(z|x)||P(z)] - E_{z \sim q_\lambda} [p(x|z)] + \log P(x) \end{aligned}$$

$$KL[q_\lambda(z|x)||p(z|x)] = E_{z \sim q_\lambda} [\log q_\lambda(z|x) - \log p(x|z)]$$
$$= KL[q_\lambda(z|x)||P(z)] - E_{z \sim q_\lambda} [p(x|z)] + \log P(x)$$

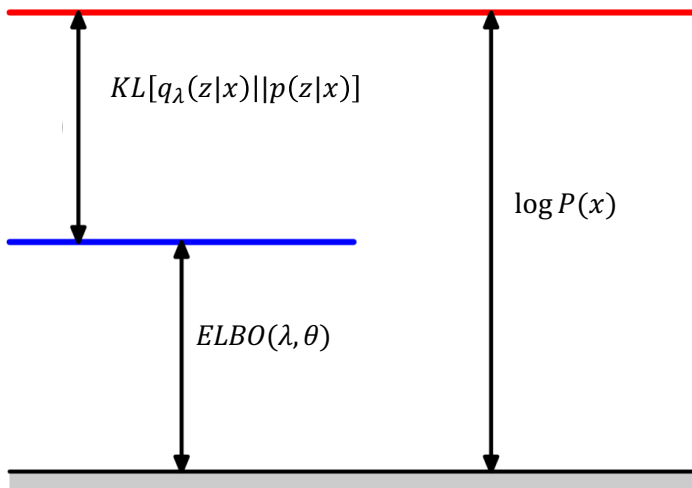
Evidence Lower Bound (ELBO)

Non dipende da z

$$\begin{aligned}
 KL[q_\lambda(z|x)||p(z|x)] &= E_{z \sim q_\lambda} [\log q_\lambda(z|x) - \log p(x|z)] \\
 &= \underbrace{KL[q_\lambda(z|x)||P(z)] - E_{z \sim q_\lambda} [p(x|z)]}_{\text{Evidence Lower Bound (ELBO)}} + \underbrace{\log P(x)}_{\text{Non dipende da } z}
 \end{aligned}$$

Evidence Lower Bound (ELBO)

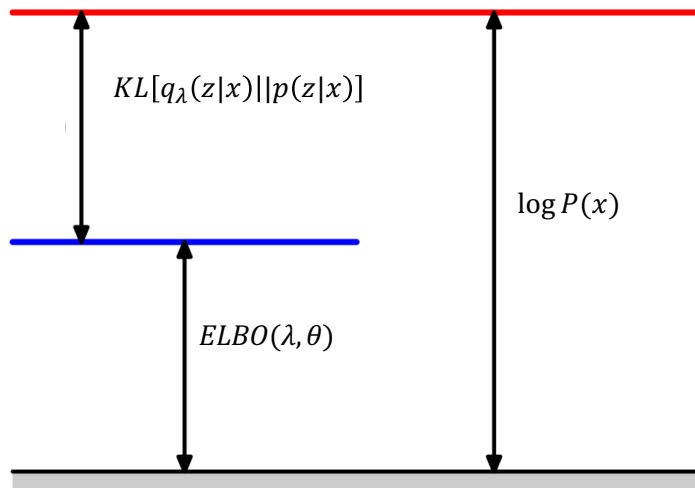
Non dipende da z



Inferenza variazionale

$$KL[q_\lambda(z|x)||p(z|x)] = -ELBO(\lambda, \theta) + \ln p(x)$$

- Scelto q_λ , si stimano i parametri λ, θ che minimizzano l'ELBO



Wrap up

- Si definisce una distribuzione q_λ
- Troviamo i parametri λ e θ che massimizzano

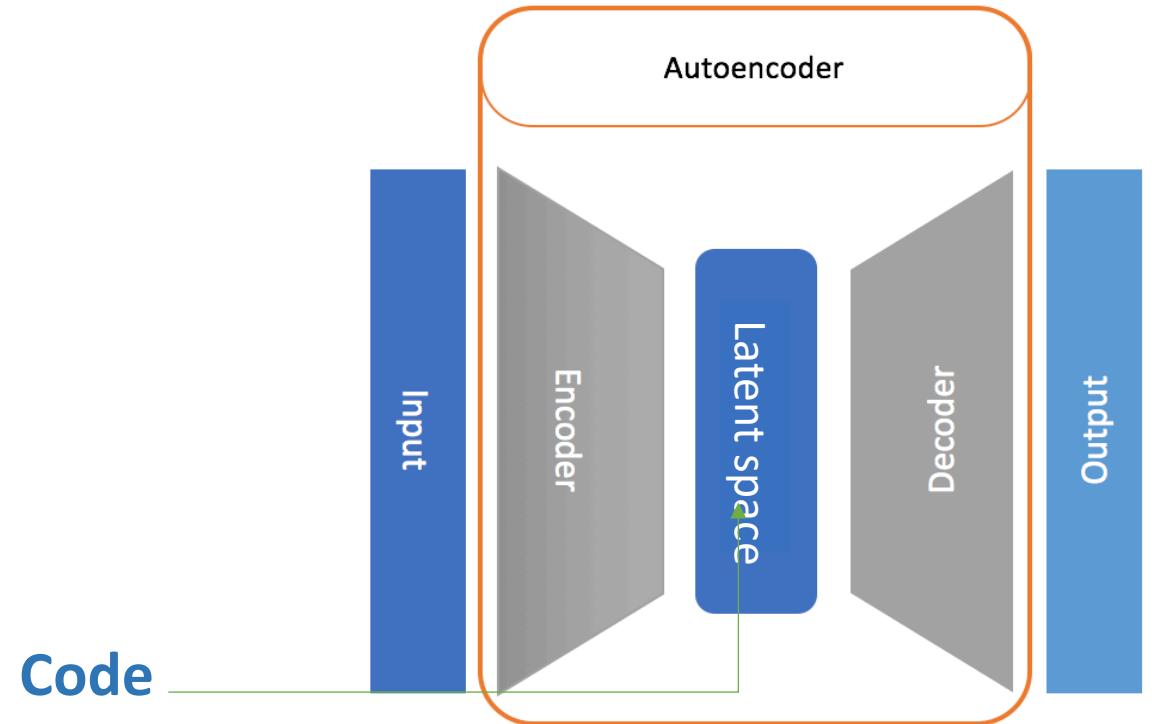
$$KL[q_\lambda(z|x) || P(z)] - E_{z \sim q_\lambda} [p(x|z)]$$

Wrap up

- Il framework dipende da due funzionali
 - $q_\lambda(z|x)$, che codifica x in una variabile latente z
 - $p(x|z)$, che decodifica z in x
- Autoencoder

Autoencoder

- un **autoencoder** è una rete neurale che produce un output (r) che è un **duplicato** dell'input (x).
- Due componenti:
 - **encoder** function $z = f(x)$
 - z è il **Codice** (dell'input)
 - **decoder** che produce una ricostruzione $r = g(z)$
 - L'**obiettivo** è $x \approx r$
 - la **loss** espressa come $L(x, g(f(x)))$



Proprietà

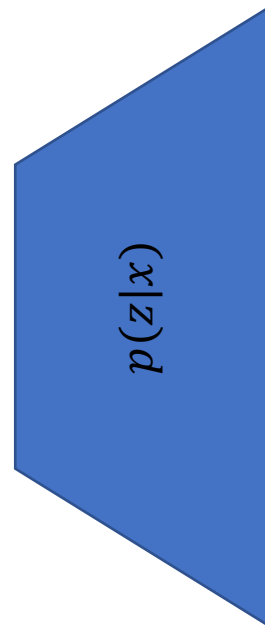
- Impara a sommarizzare le caratteristiche dell'input
- Rappresenta i dati originali in uno spazio a bassa dimensionalità

x



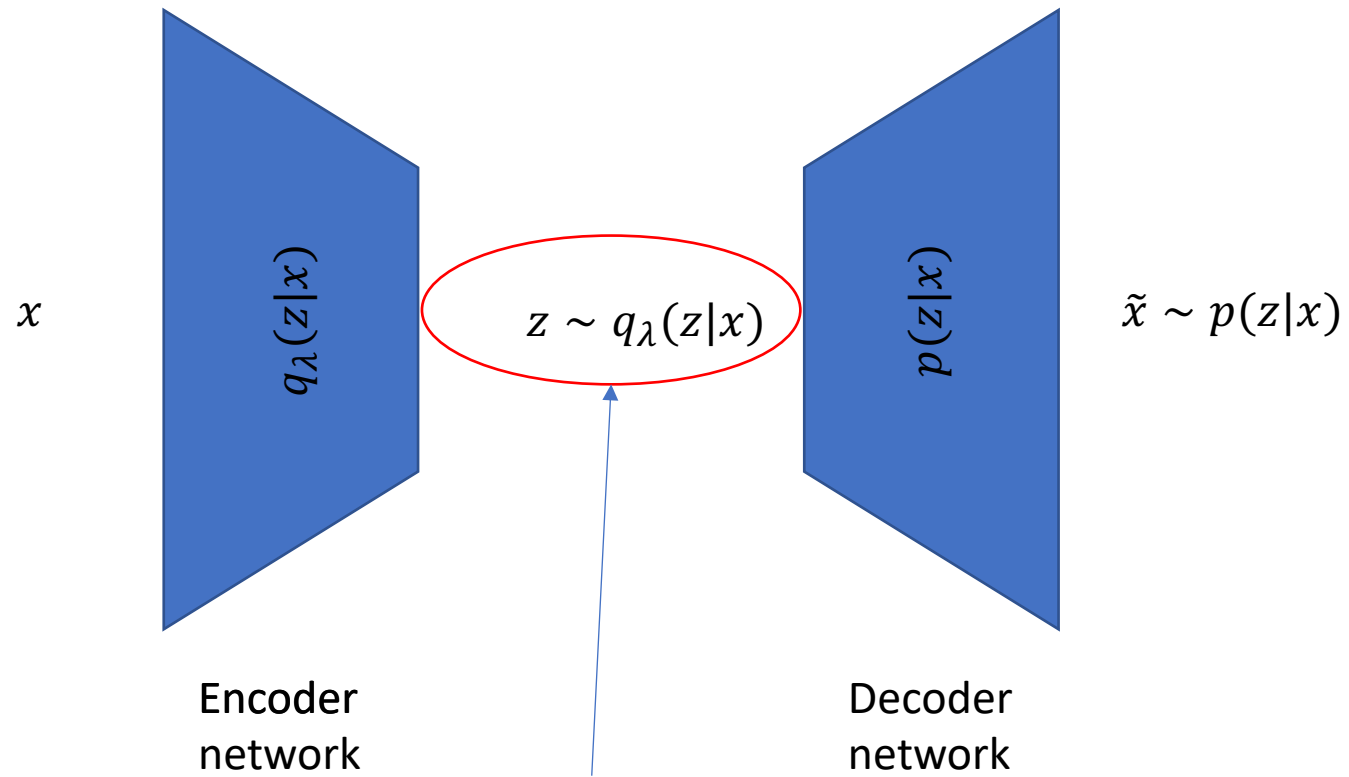
Encoder
network

$z \sim q_\lambda(z|x)$

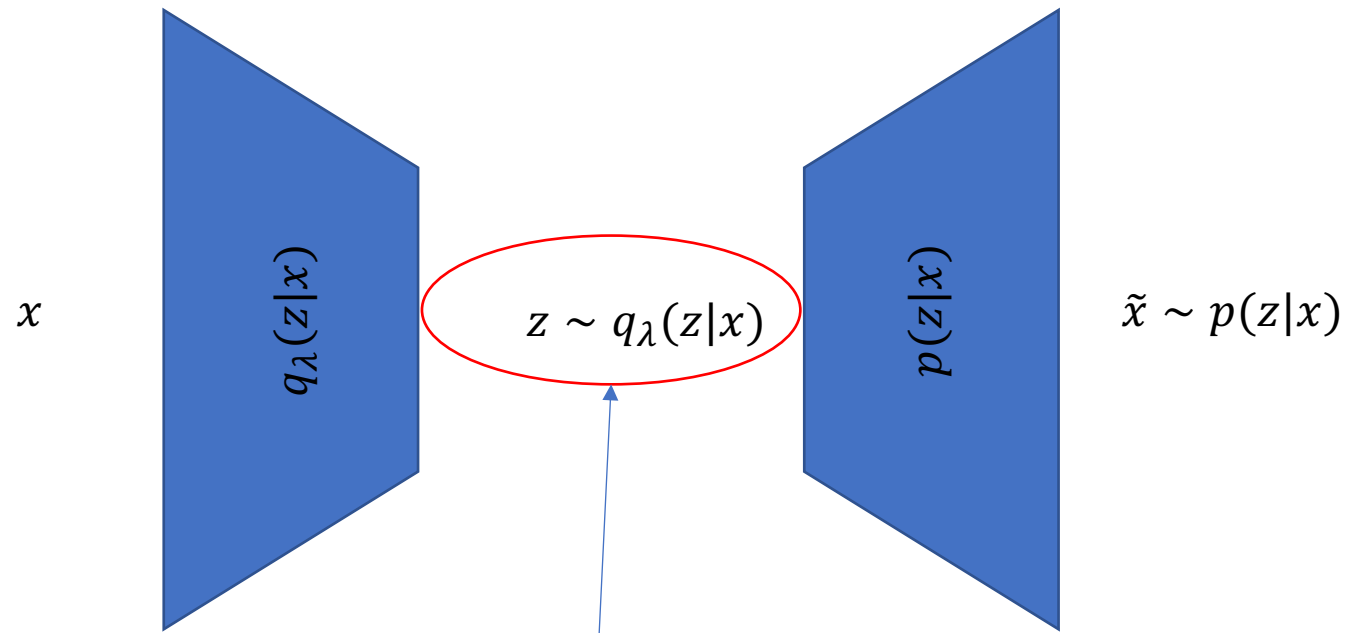


Decoder
network

$\tilde{x} \sim p(z|x)$

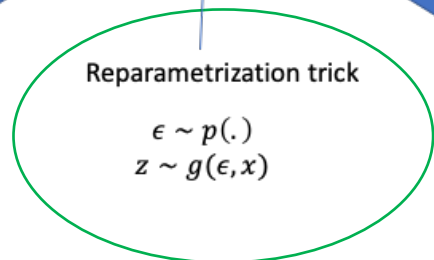
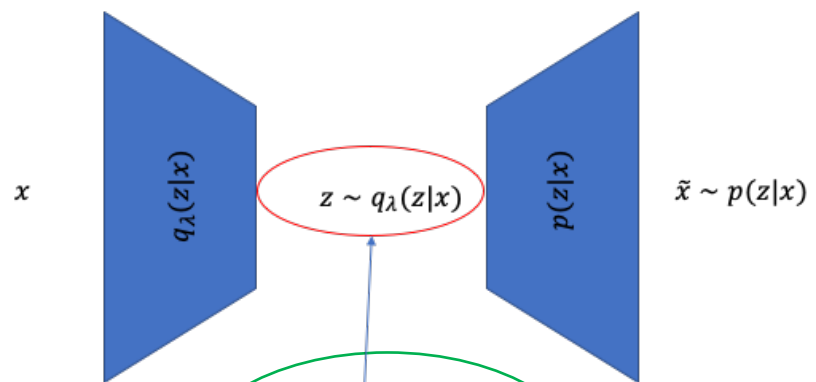


Come si ottimizza
Evitando di spezzare
La propagazione del gradiente?



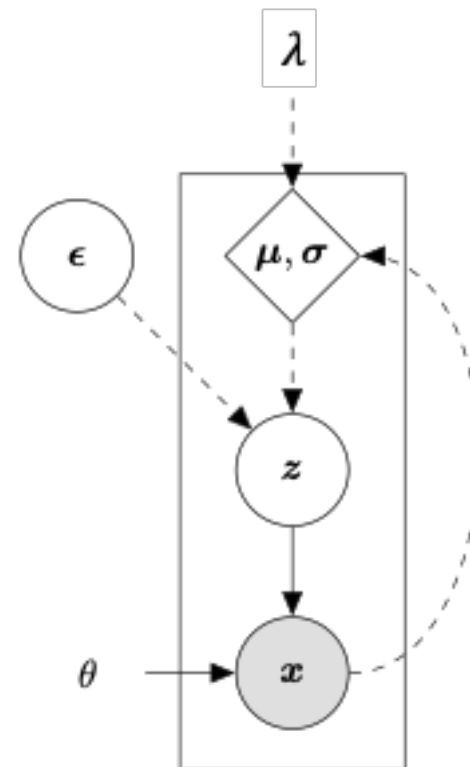
Reparameterization trick

$$\begin{aligned} \epsilon &\sim p(\cdot) \\ z &\sim g(\epsilon, x) \end{aligned}$$



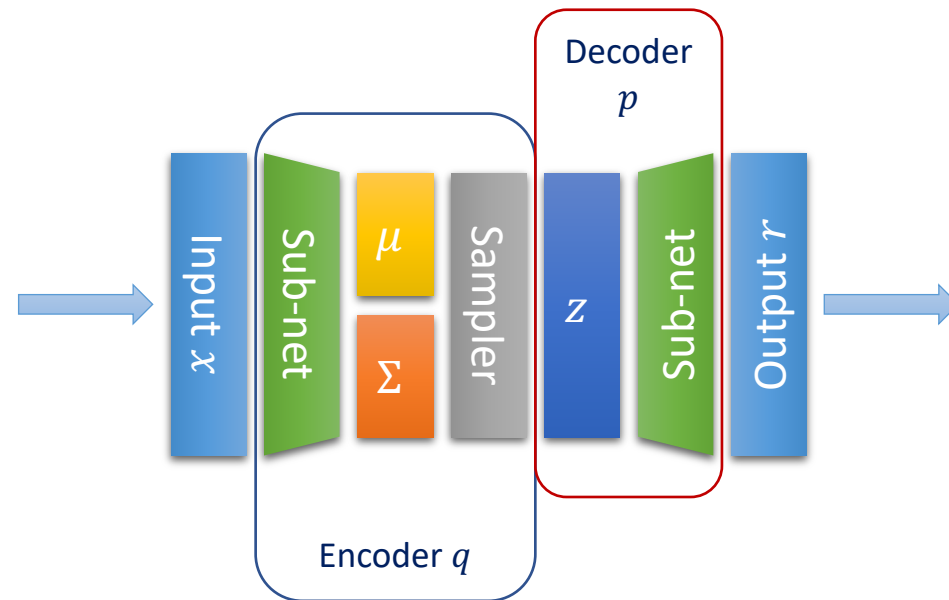
Per esempio,

$$\begin{aligned} \epsilon &\sim \mathcal{N}(0, I) \\ \mu, \sigma &= \text{mlp}(x) \\ z &= \mu + \sigma \cdot \epsilon \end{aligned}$$

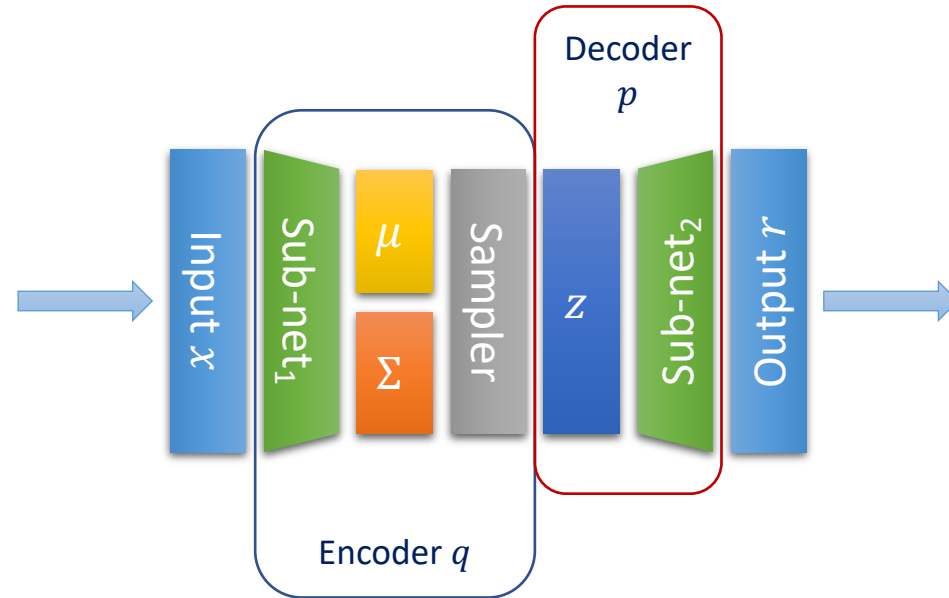


Variational Autoencoders

- Architettura generale



Variational Autoencoders



- La sottorete sub-net₁ calcola i parametri di una distribuzione gaussiana
- Il sampler sfrutta i parametri per campionare il codice
- La sub-net₂ genera l'output

Variational Autoencoders

- Cosa manca?

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = KL[q_\lambda(z|x) || P(z)] - E_{z \sim q_\lambda}[\log p(x|z)]$$

- Ammette una forma chiusa?

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = KL[q_\lambda(z|x) || P(z)] - E_{z \sim q_\lambda}[\log p(x|z)]$$

- Log-likelihood

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = KL[q_\lambda(z|x) || P(z)] - E_{z \sim q_\lambda}[\log p(x|z)]$$

- Se q_λ e P sono gaussiane ammette una forma chiusa

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = \boxed{KL[q_\lambda(z|x) || P(z)]} - E_{z \sim q_\lambda}[\log p(x|z)]$$

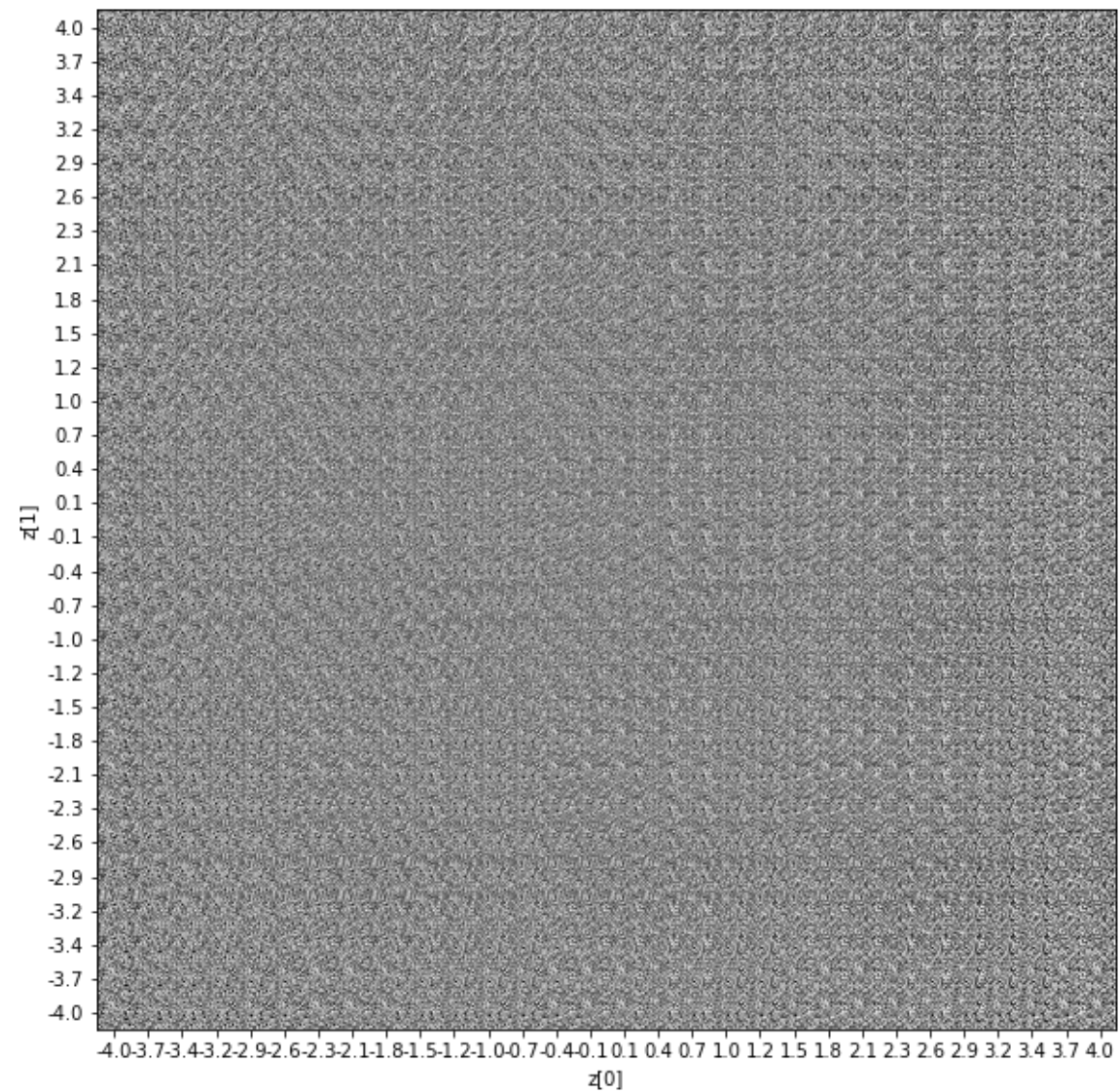
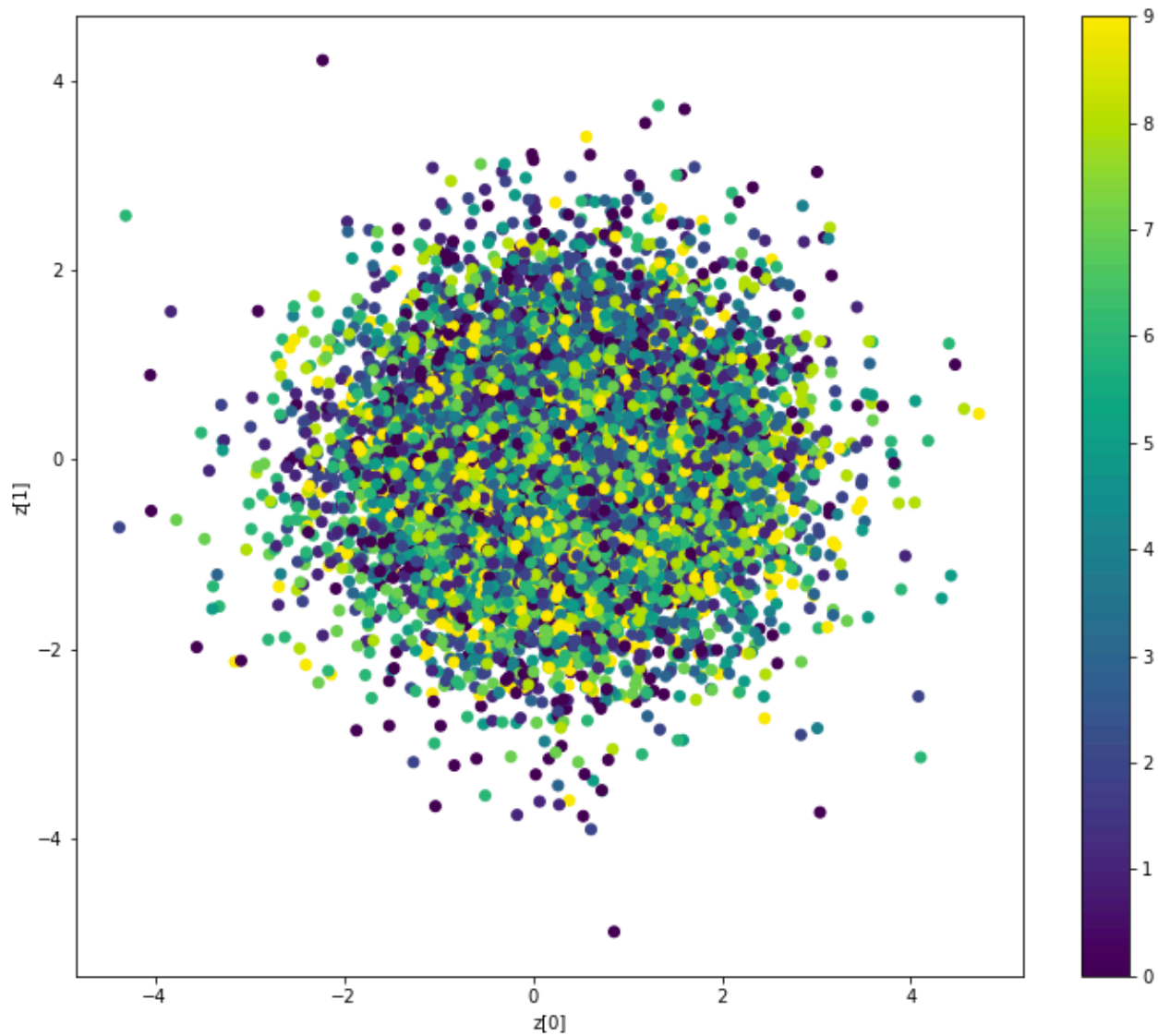
- Se q_λ e P sono gaussiane ammette una forma chiusa

$$KL[\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) || \mathcal{N}(\boldsymbol{m}, \boldsymbol{S})] = \frac{1}{2} \left(\log \frac{|\boldsymbol{S}|}{|\boldsymbol{\Sigma}|} - K + \text{tr}(\boldsymbol{S}^{-1} \boldsymbol{\Sigma}) + (\boldsymbol{m} - \boldsymbol{\mu})^T \boldsymbol{S}^{-1} (\boldsymbol{m} - \boldsymbol{\mu}) \right)$$

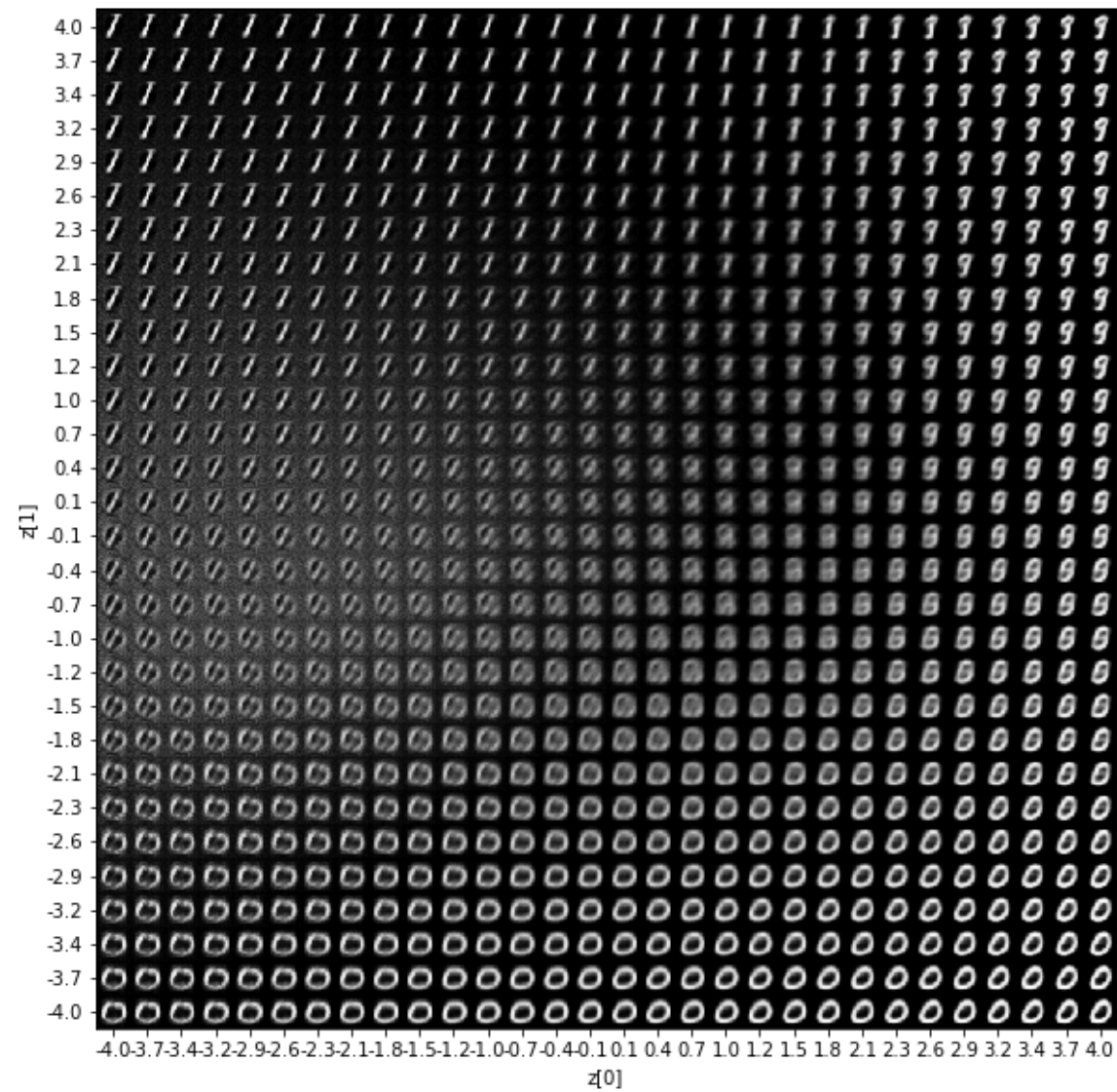
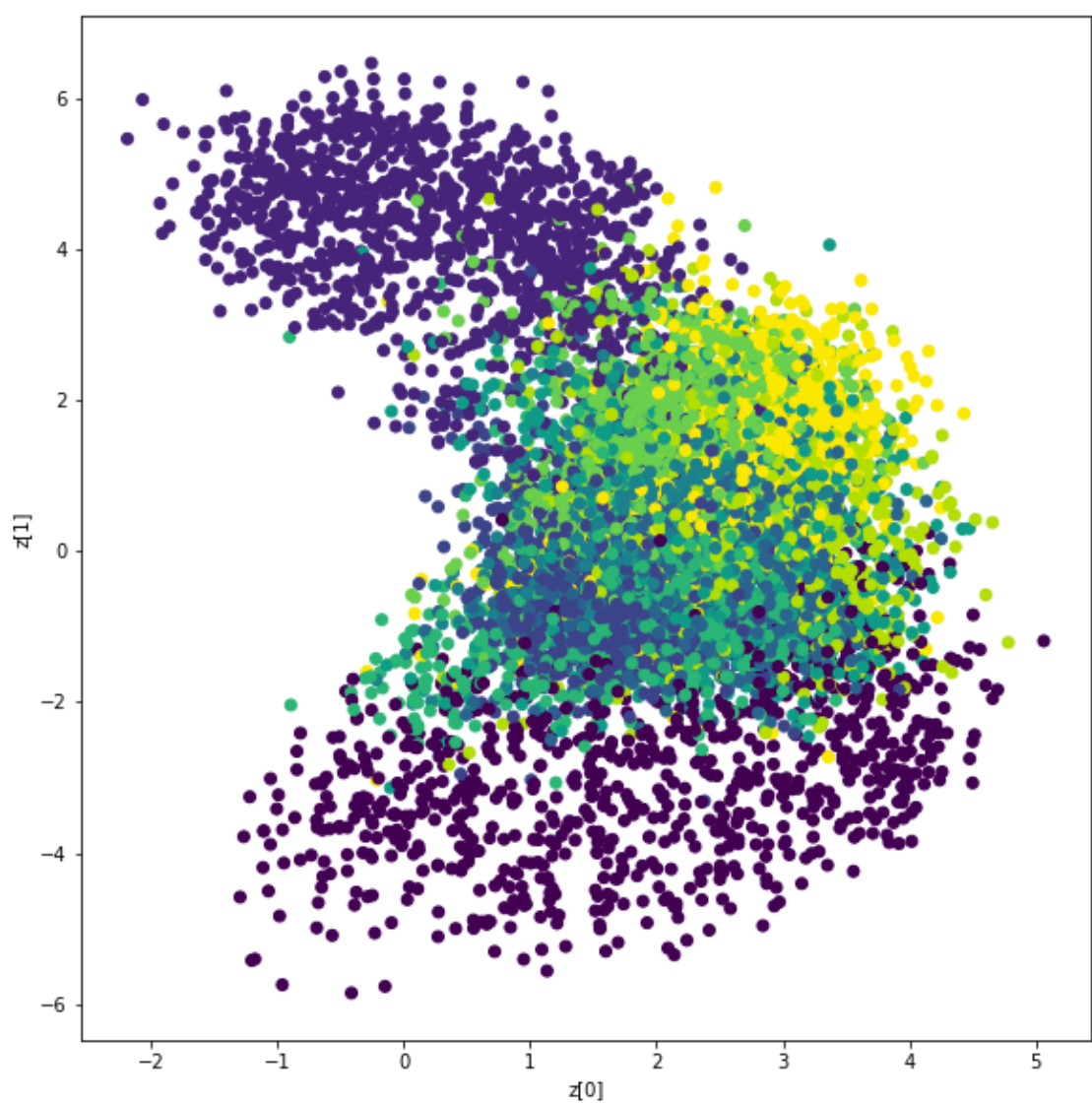
Recap: MC vs VI

- Assunzioni sulla forma delle distribuzioni
 - MCMC: **no**
 - VI: **yes**
- Bias (dobbiamo forzare delle assunzioni?)
 - MCMC: **low**
 - VI: **high**
- Variance (quanto è grande lo spazio di ricerca?)
 - MCMC: **high**
 - VI: **low**
- Computational cost
 - MCMC: **high (lots of iterations)**
 - VI: **low**
- Accuracy
 - MCMC: **very good**
 - VI: **good**

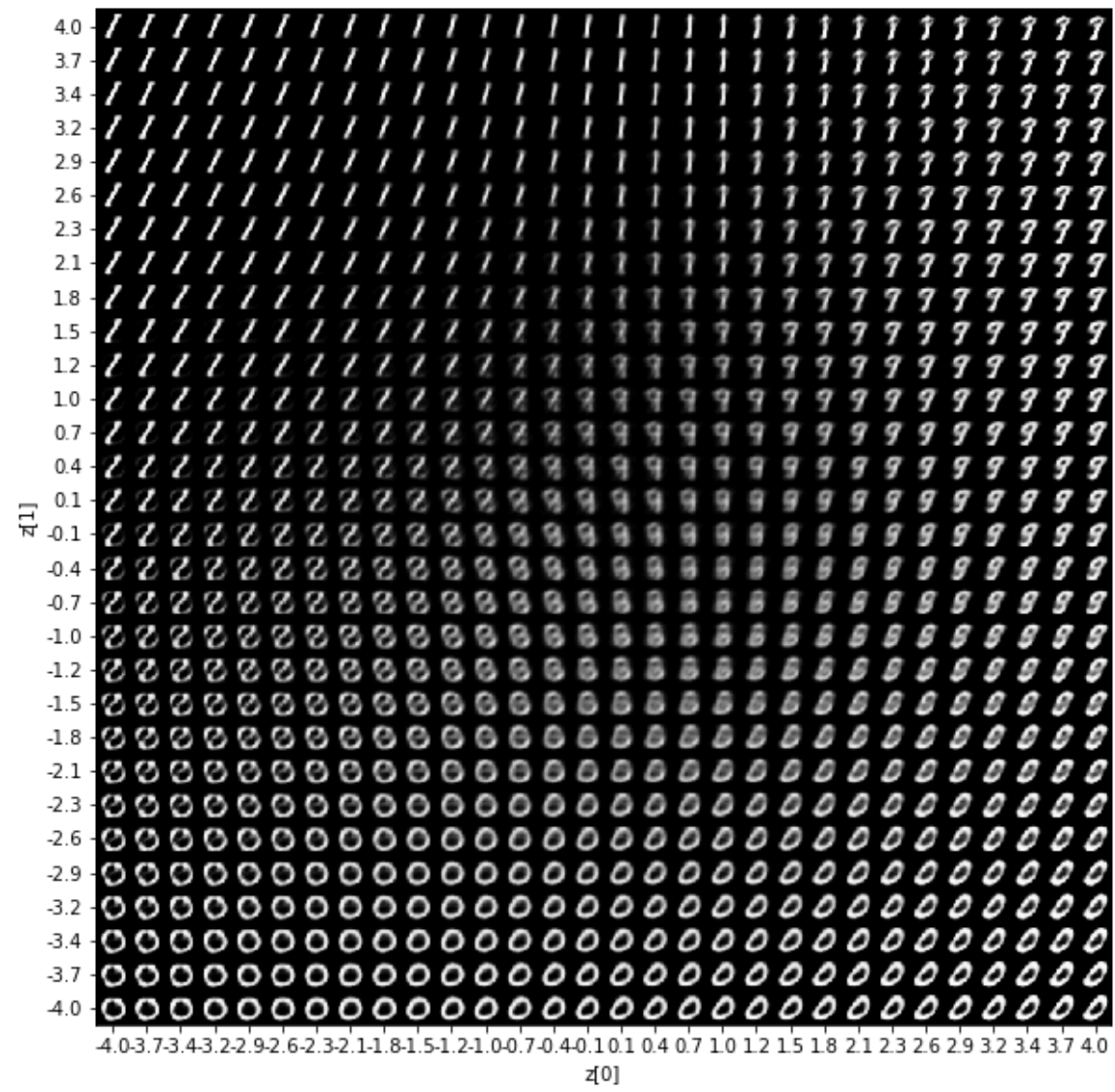
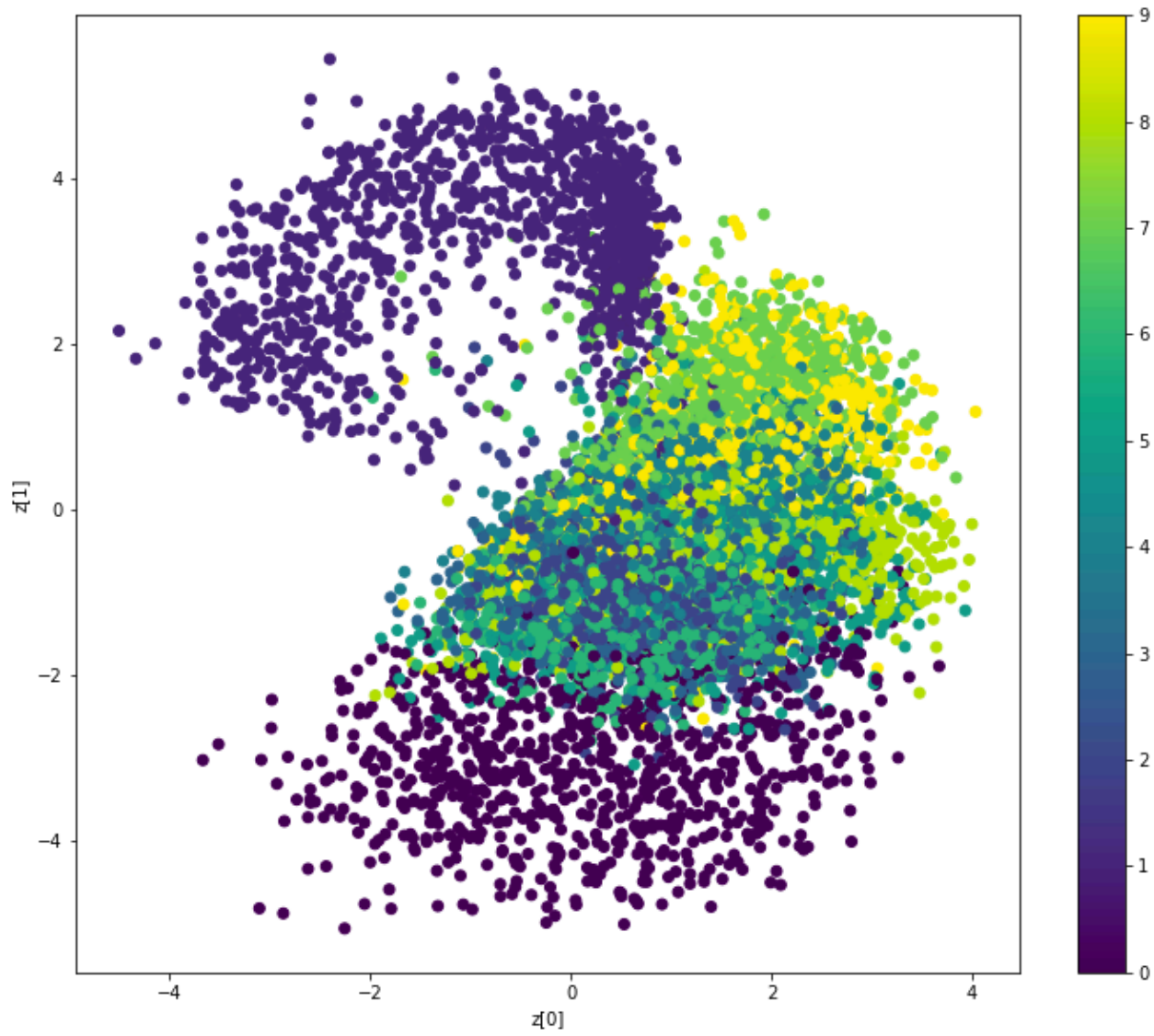
MNIST – epoca 0



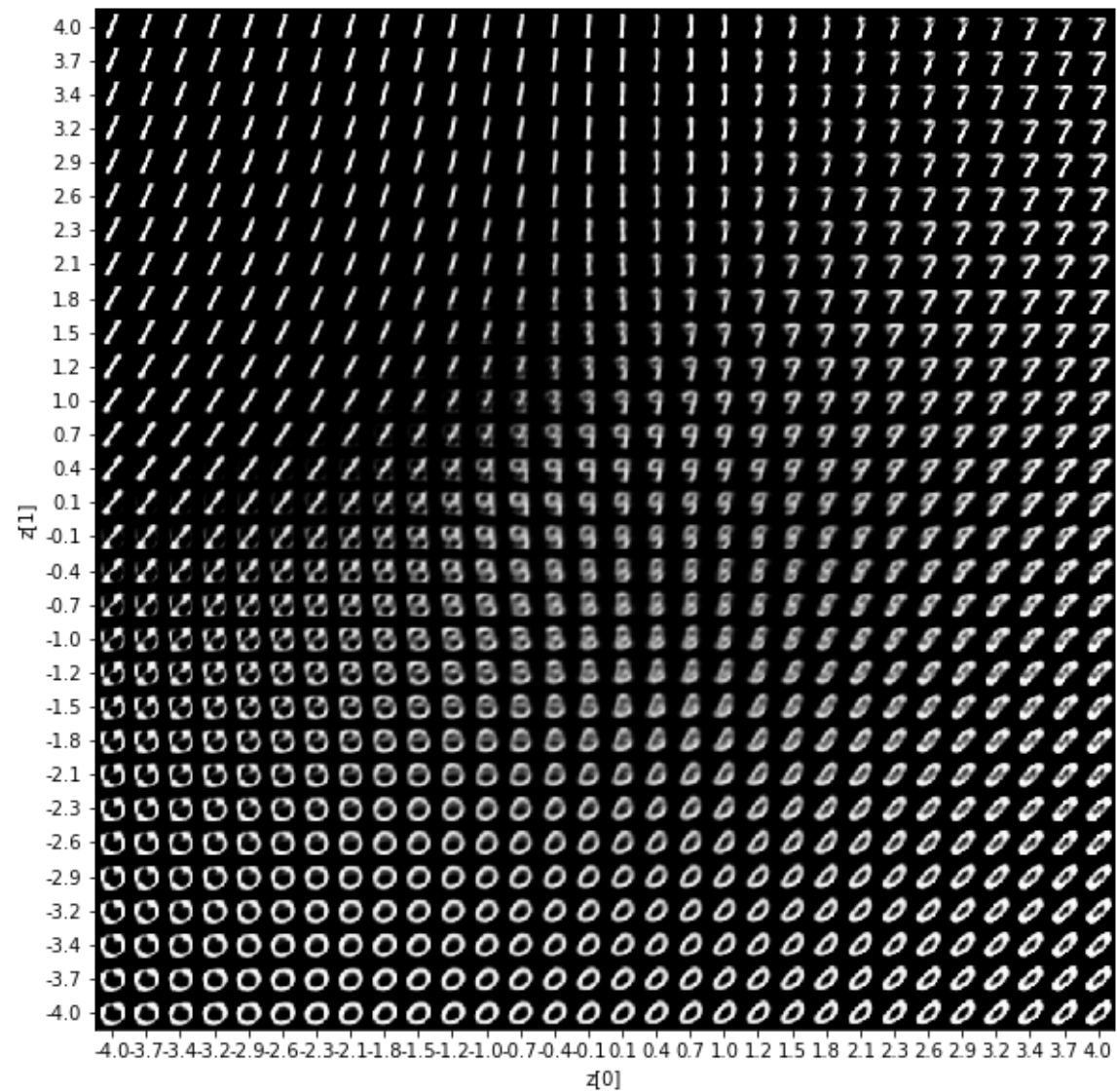
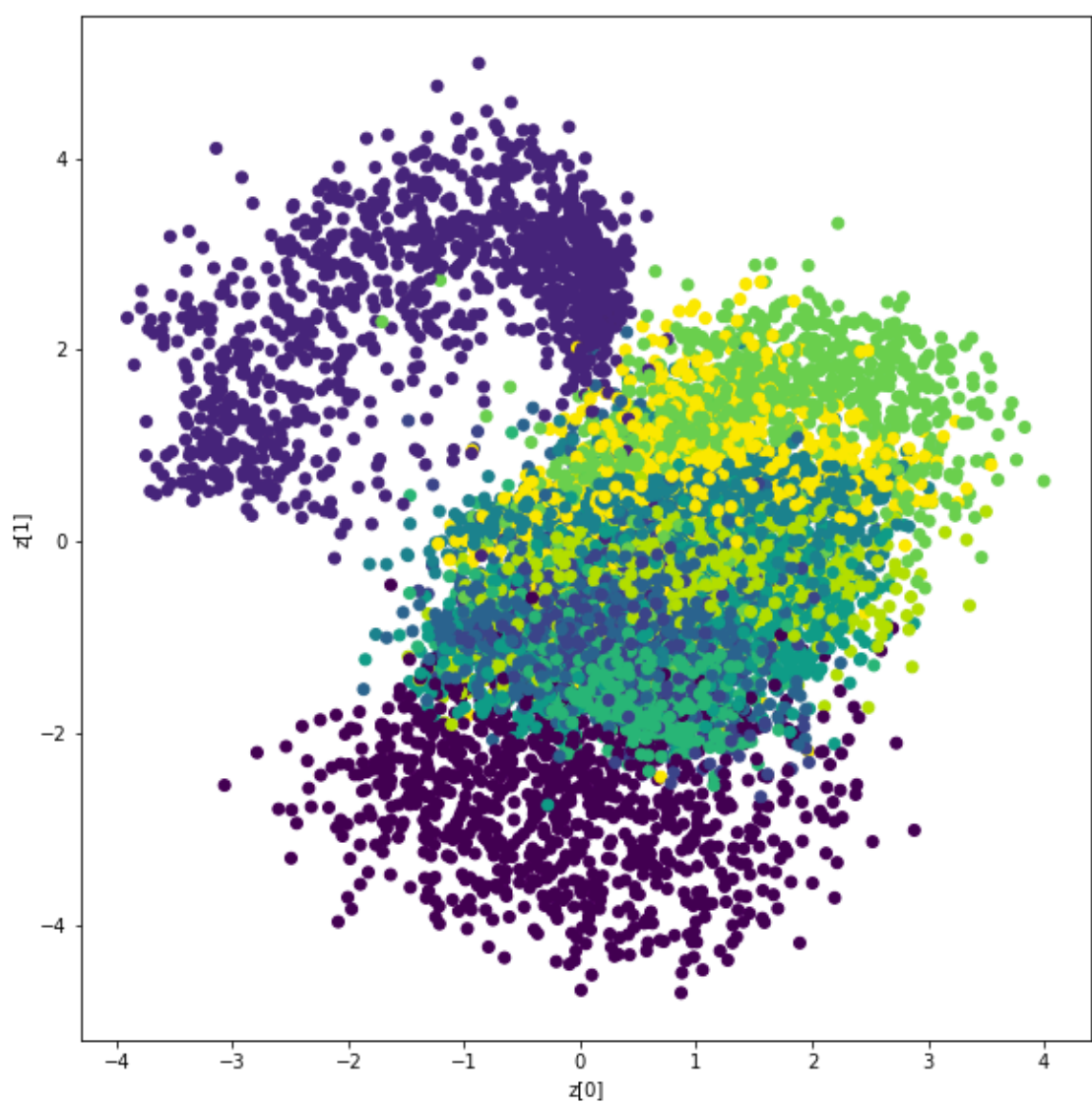
MNIST – epoca 10



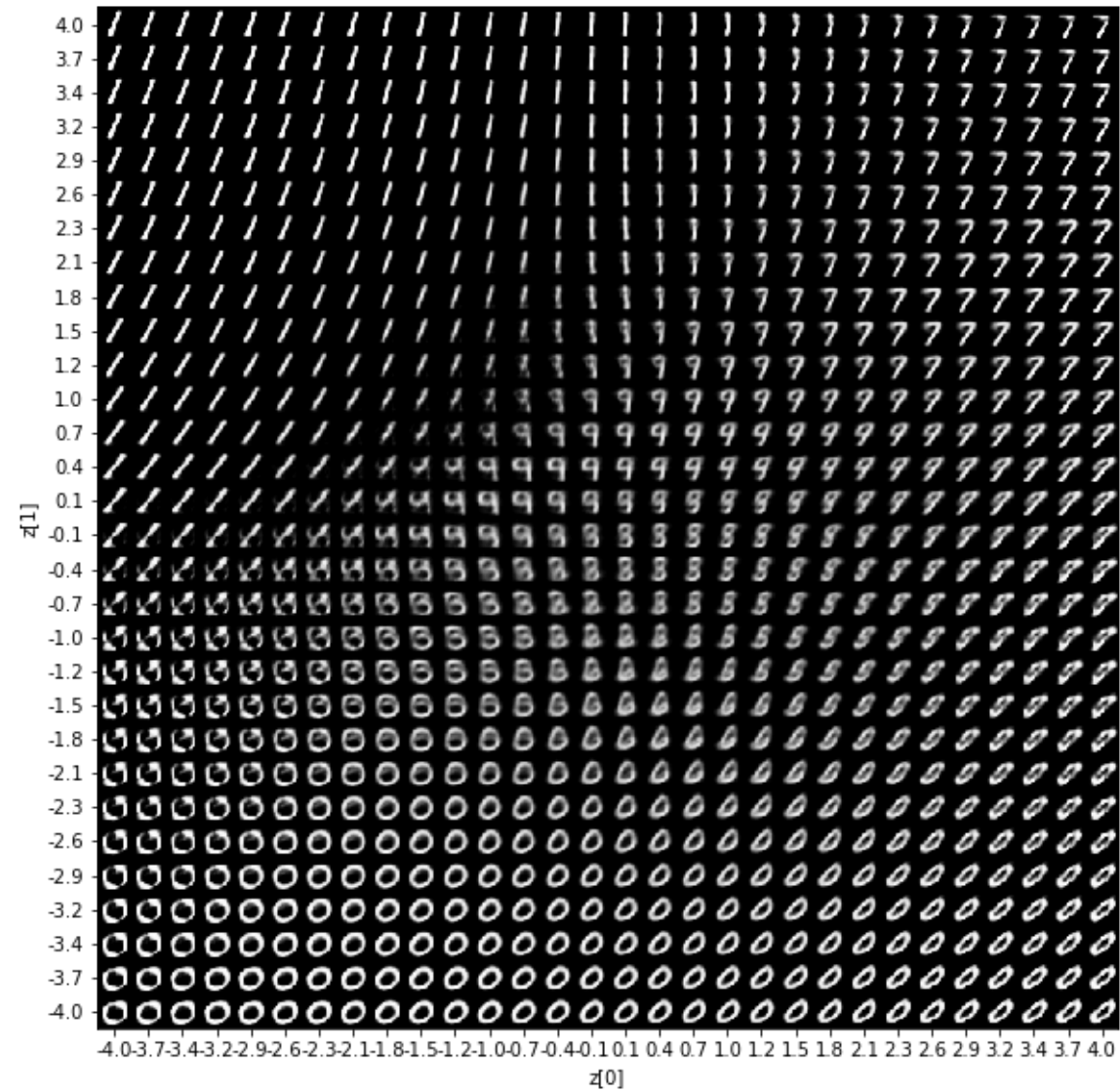
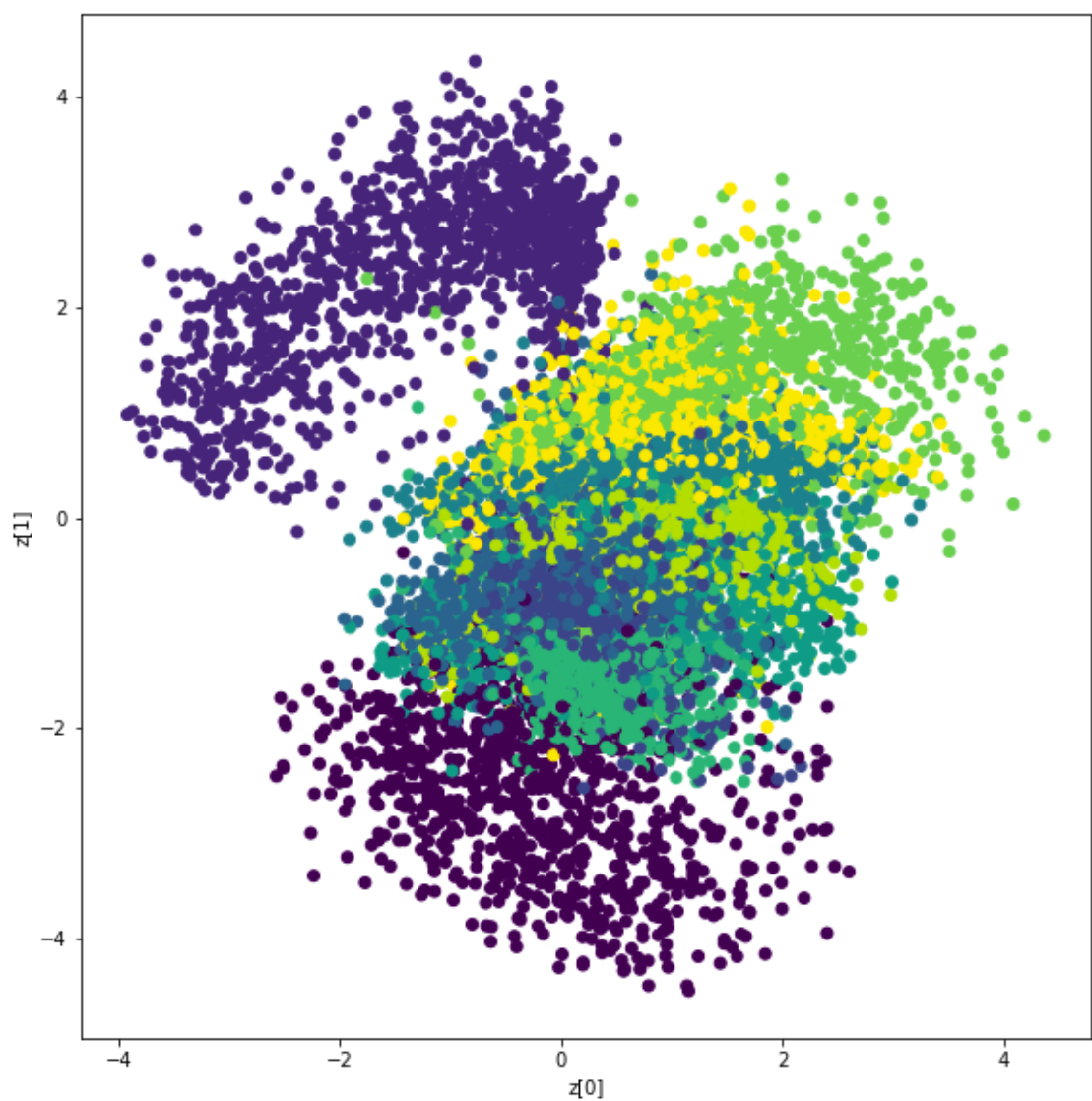
MNIST – epoca 20



MNIST – epoca 50



MNIST – epoca 100



MNIST – epoca 200

