# Generative Models

PHYS591000 2022.05.04

# Announcement

- Due to remote gathering, each student hands in his/her own lab for this week.

# Outline

- Today we are going to talk about an important task of unsupervised learning: generative modeling.

Ref:
Lecture 18 by Prof. Hung-Yi Lee (youtube)
Lecture 13 of CS231(2017) at Stanford (youtube)

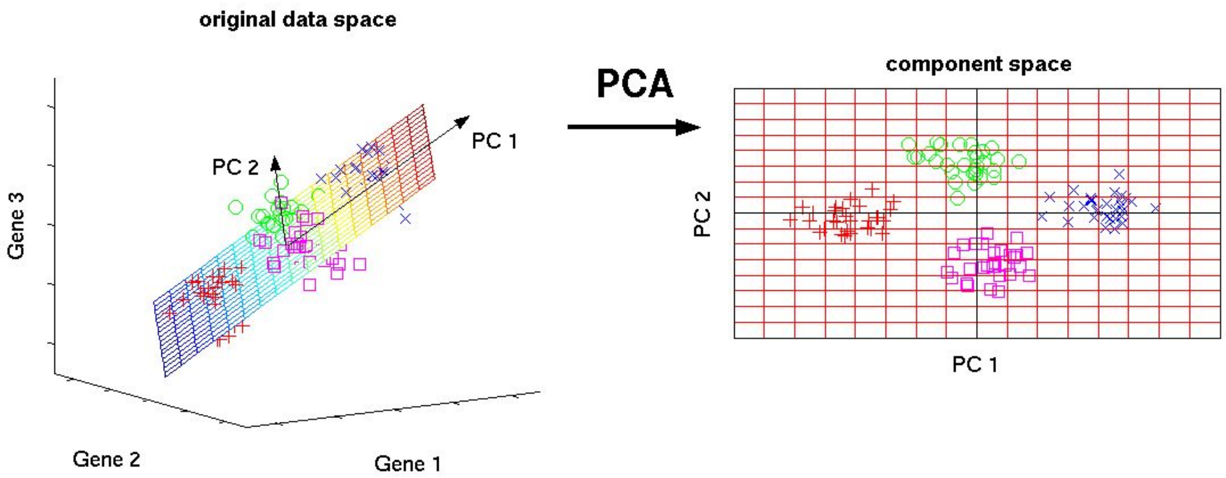
# Review: Unsupervised Learning

- The training data are not **labeled** (no information of the **ground truth** given to the model).

- The goal of learning is to find some hidden structure of the data.

# Review: Unsupervised Learning

- **Clustering**: Grouping objects according to their similarities.
  **Dimensionality reduction**: Find the most representative features (principal component analysis, PCA).



# Unsupervised Learning

- **Feature learning**: Reproduce data (images) that look like the original ones, e.g. Autoencoder

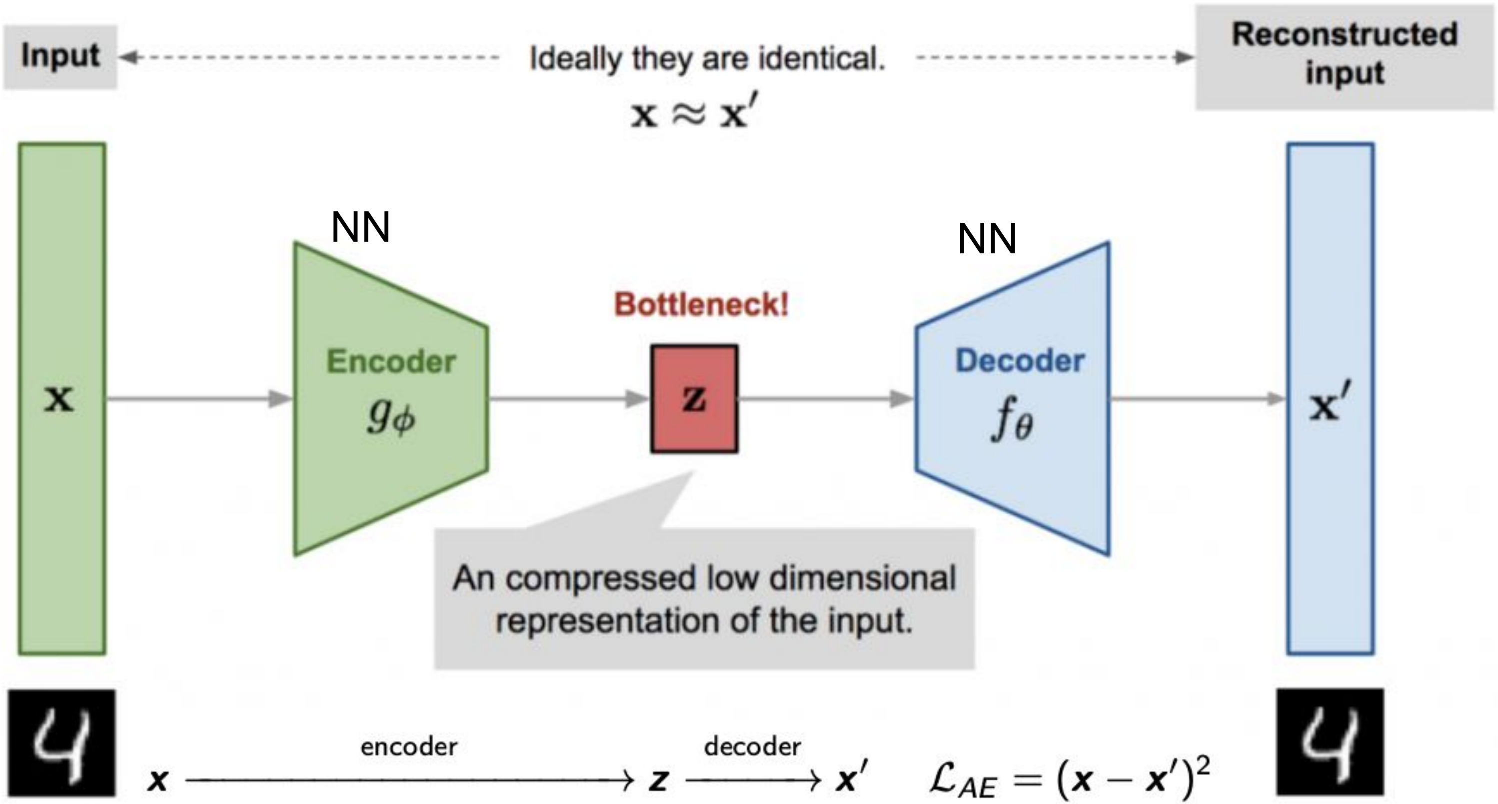
- **Density estimation**: Find/approximate the underlying probability distribution of data. Use it to generate new data (Generative models).
  E.g. Variational Autoencoder (VAE), Generative Adversarial Network (GAN)

# Autoencoder

- An autoencoder compresses data into a lower-dimensional representation (‘keep only important features’) with an **encoder**, then reconstructs data from the lower-dimensional space (**latent space**) with a **decoder**.

- Goal: Make the output as close to the input as possible.
  Loss: ‘Distance’ between the input and output images.
  Autoencoding = encoding itself.

# Autoencoder

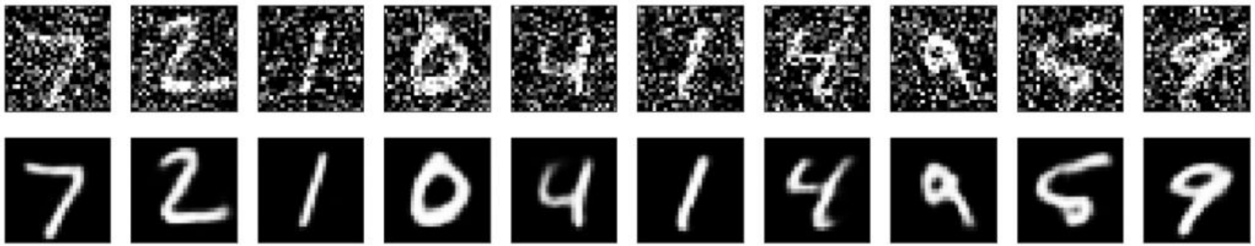



$\boldsymbol{x} \xrightarrow{\text{encoder}} \boldsymbol{z} \xrightarrow{\text{decoder}} \boldsymbol{x}' \quad \mathcal{L}_{AE} = (\boldsymbol{x} - \boldsymbol{x}')^2$

# Autoencoder

- Autoencoders capture the most important features of the data automatically.
- Application: Data denoising (filtering out noises).

Input



Output

# Variational Autoencoders (VAE)

- Autoencoders do not know about the *distribution* of data (x) in the latent space (z). → Cannot generate a (new) sample from an arbitrary point in z.

- Variational autoencoder (VAE): Probabilistic variation of autoencoder. Allow us to *sample* from the model to generate data!

# Variational Autoencoders (VAE)

- The **decoder** of VAE is trained to obtain two outputs, the mean and the variance of the probability distribution P(z) (usually assume to be Gaussian).

*Lecture from Prof. Hung-Yi Lee (NTU)*

$z \sim N(0, I)$

z is a vector from normal distribution

$x|z \sim N\big(\mu(z), \sigma(z)\big)$

Each dimension of z represents an attribute


$$P(x) = \int_z P(z)P(x|z)dz$$

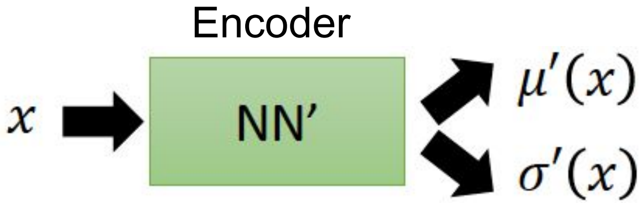
# Variational Autoencoders (VAE)

- However it is very difficult to optimize on P(x): almost impossible to compute P(x|z) for every z

$$P(x) = \int_z P(z)P(x|z)dz$$

- Idea: Make use of Bayes' theorem: P(z|x) = P(x|z)P(z)/P(x)
  - Use an **encoder** NN to predict a distribution q(z|x) which approximates P(z|x)

$$z|x \sim N(\mu'(x), \sigma'(x))$$



# Variational Autoencoders (VAE)

- Reason: This allows us to derive a lower bound on the data *likelihood*, which we can optimize.

$$P(x) = \int_z P(z)P(x|z)dz$$

$$L = \sum_x \log P(x)$$

Maximizing the likelihood of the observed x

# Variational Autoencoders (VAE)

*Lecture from Prof. Hung-Yi Lee (NTU)*

$$\log P(x) = \int_z q(z|x) \log P(x) dz$$

q(z|x) can be any distribution

$$= \int_z q(z|x) \log\left(\frac{P(z,x)}{P(z|x)}\right) dz = \int_z q(z|x) \log\left(\frac{P(z,x)}{q(z|x)} \frac{q(z|x)}{P(z|x)}\right) dz$$

$$= \int_z q(z|x) \log\left(\frac{P(z,x)}{q(z|x)}\right) dz + \underbrace{\int_z q(z|x) \log\left(\frac{q(z|x)}{P(z|x)}\right) dz}_{KL(q(z|x)||P(z|x))}$$

$\geq 0$

$$\geq \int_z q(z|x) \log\left(\frac{P(x|z)P(z)}{q(z|x)}\right) dz$$

*lower bound* $L_b$

**Kullback-Leibler (KL) divergence**

- KL is a measure of how different one probability distribution is from another

# Variational Autoencoders (VAE)

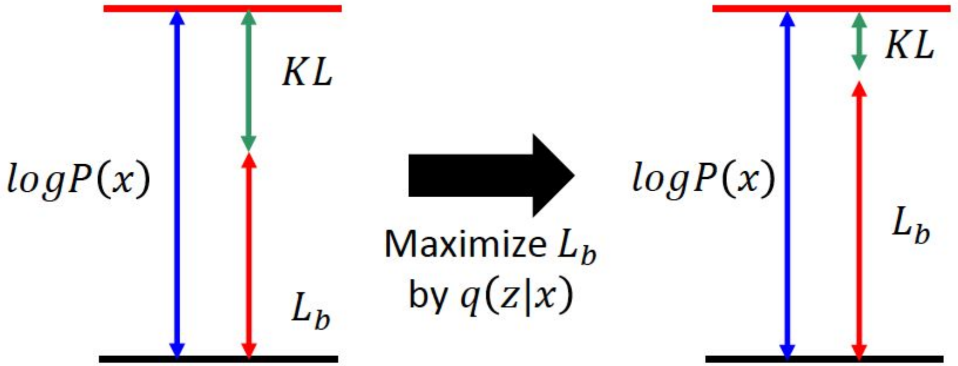
$$logP(x) = L_b + KL\big(q(z|x)||P(z|x)\big)$$

$$L_b = \int_z q(z|x)log\left(\frac{P(x|z)P(z)}{q(z|x)}\right)dz$$

Original Goal: Max. likelihood by tuning P(x|z)

$$L = \sum_x logP(x)$$

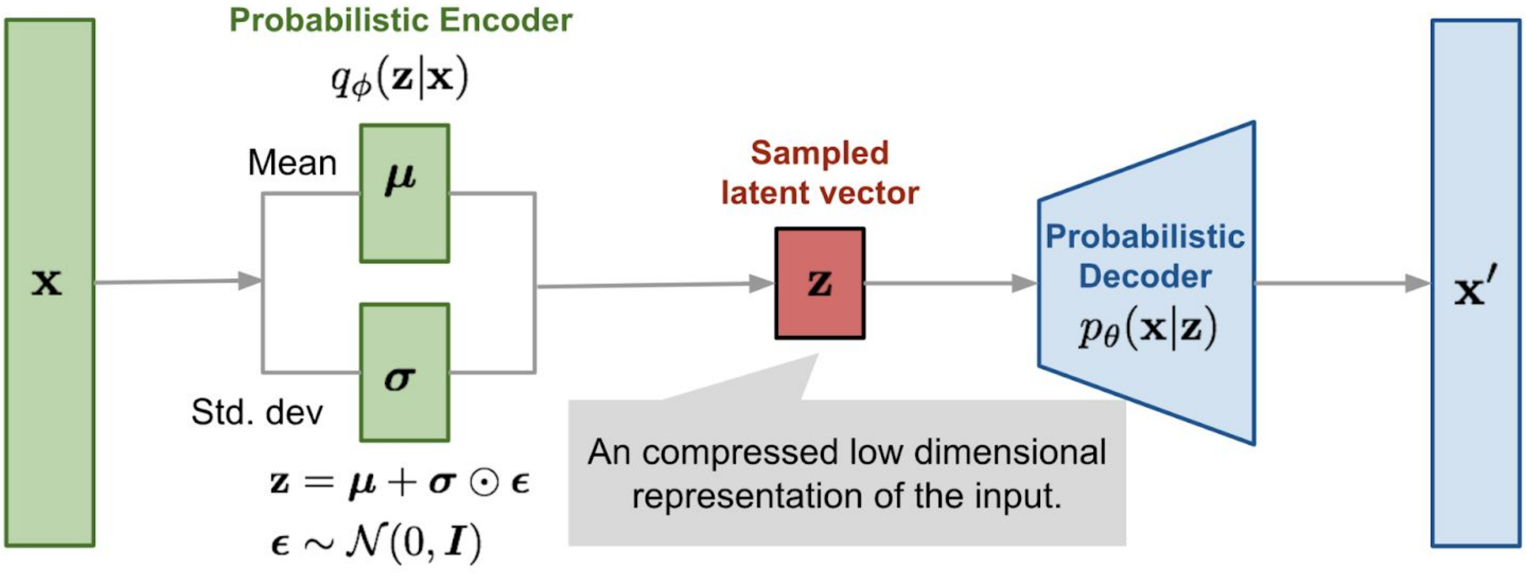
$$P(x) = \int_z P(z)P(x|z)dz$$




Add a 2nd goal: Max. L_b by tuning q(z|x)

– Not only get an estimation of P(x|z), but also make q(z|x) a good approx. of P(z|x).

# Variational Autoencoders (VAE)




- Total loss (to be minimized) = Reconstruction loss (as in AE) + KL-loss

# In-class demo for this week

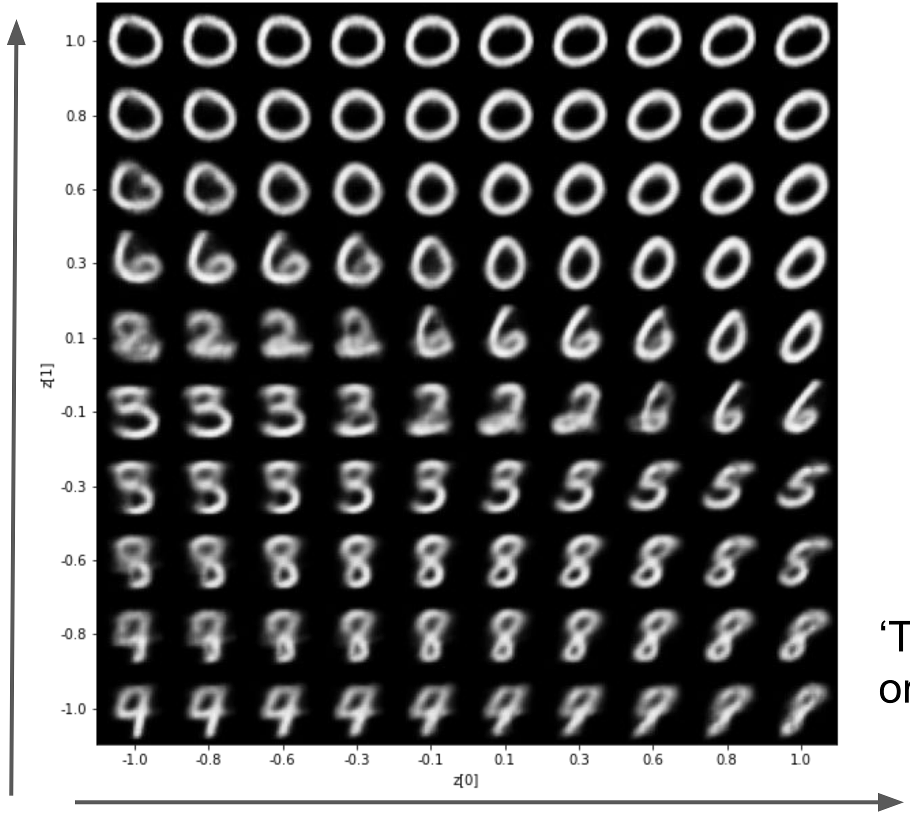
- Due to remote teaching we'll simply show the results of in-class exercise. Feel free to play with it afterwards.

- We'll train a VAE to extract distributions of 2 features (2D latent space z) of MNIST, and plot digits generated by the VAE in different points on the 2D z-plane.

# In-class demo for this week

Sometimes we can make sense of the latent space

‘The size and position of the circles’




‘The amount and orientation of tilt’

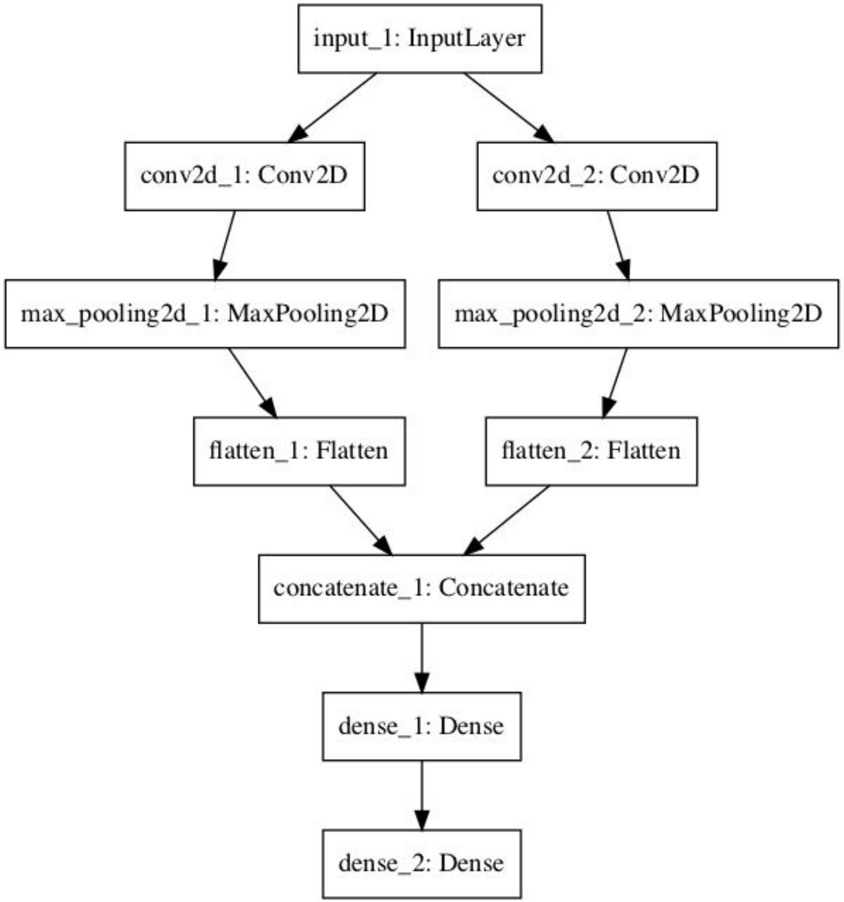
# Lab for this week

- For Lab this week we’ll build an autoencoder (AE) to compress and reproduce a picture from the FlyCircuit (brains of fruit flies) dataset.

# Backup

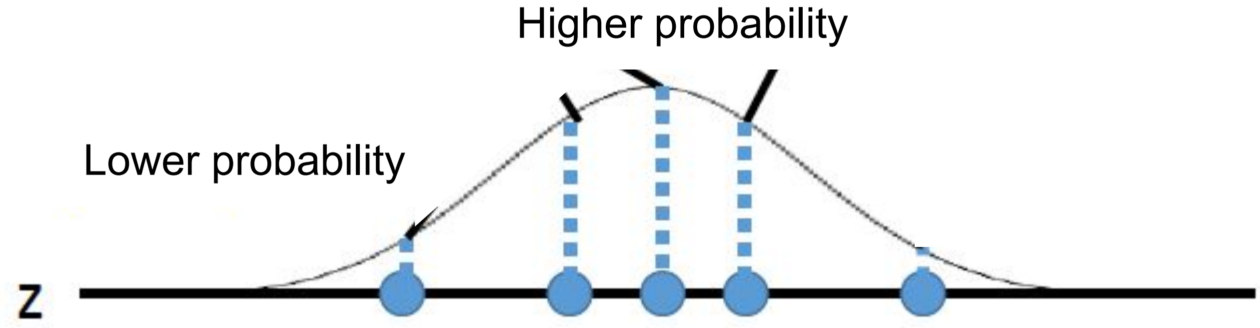
# Functional API

- Keras functional API allows one to create a model with shared layers like the topology shown on the right. This cannot be carried out in Sequential API.



# More statistics jargon

- Sampling from a probability distribution: Get a random sample (point) according to the underlying probability distribution

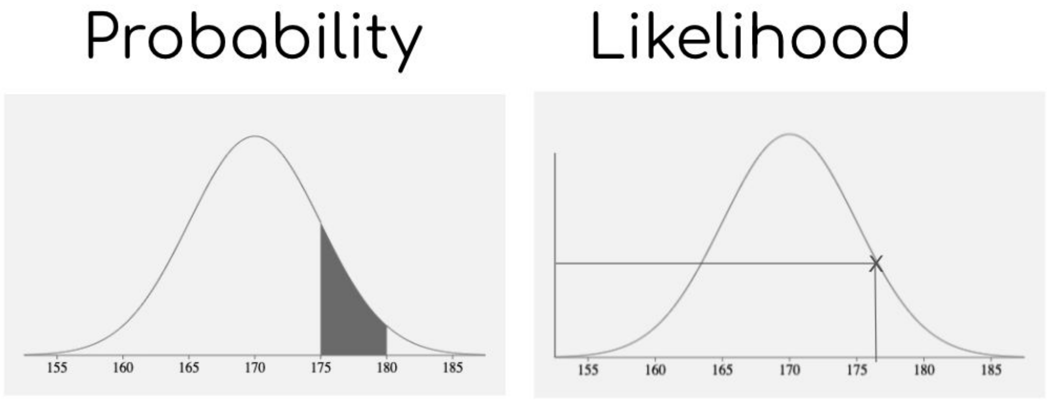


# More statistics jargon

**“Given”**

- Probability: Prob(observation | distribution)

  Likelihood: How likely a distribution to be the real underlying distribution *given* the observation (data)



# More statistics jargon

- Example:
    - Given it’s a fair coin (50-50 prob. distribution), what is the probability to get 3 heads and 7 tails if you flip the coin 10 times?
    - If you flip a coin 10 times and get 3 heads and 7 tails, how likely it is a fair coin/What is the likelihood of it being a fair coin?