2024-1

Multivariate Data Analysis

Youjin Shin
(yj.shinn@catholic.ac.kr)

Likelihood

◆ Likelihood

- It tells us how "likely" our observed data is given certain parameters of the model
- The higher the “likelihood” value, the more likely the event will occur
  - ✓ We cannot say that the probability will be high.
- A measure of how well a statistical model or set of parameters explains observed data
- Unlike probability, it does not sum or integrate to 1

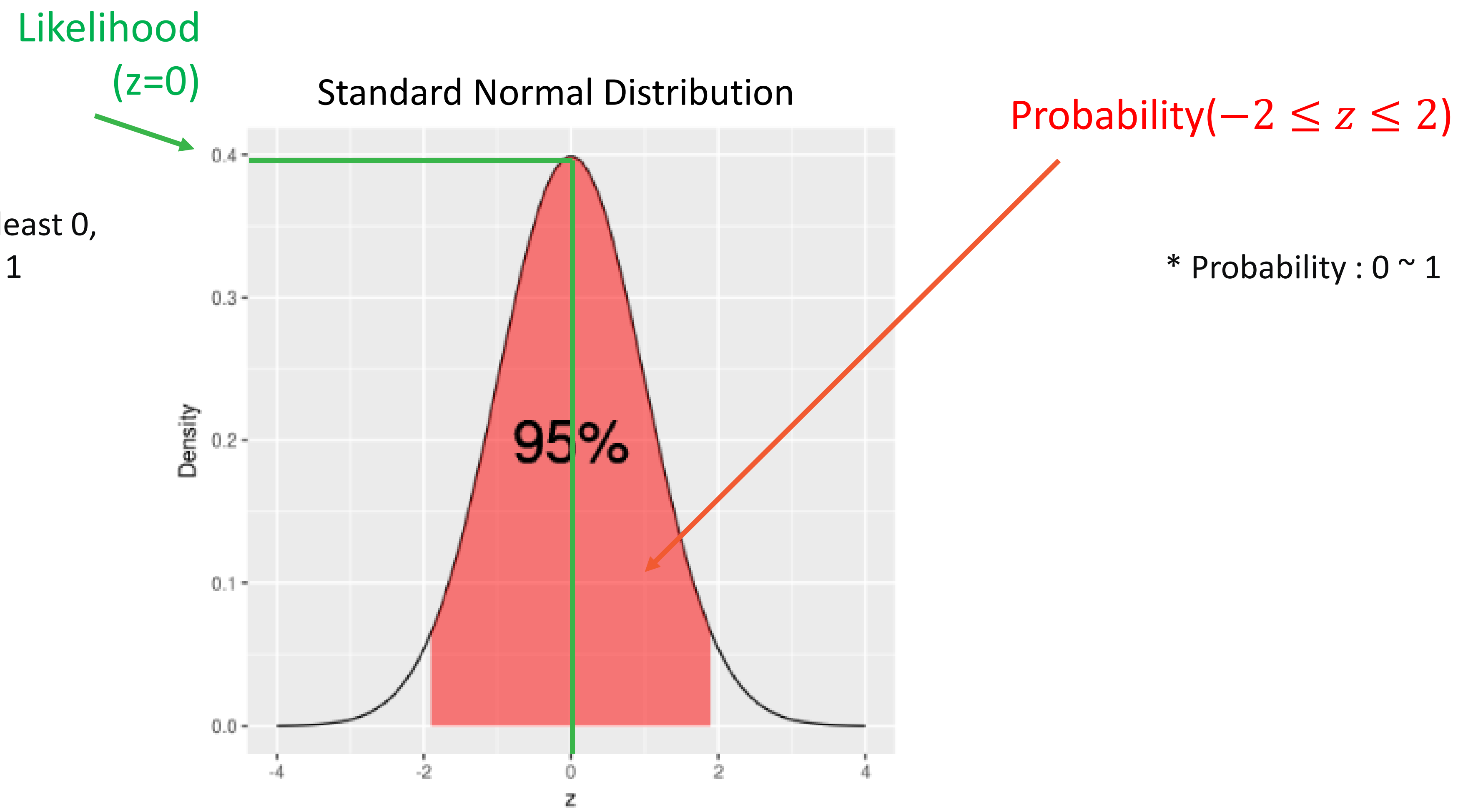
◆ Probability = Likelihood?

- **For discrete events** (like rolling dice), the terms "probability" and "likelihood" can often be used interchangeably. Probability directly quantifies the chance of each specific outcome.

- **For continuous events** (like measurements modeled by normal distributions), “likelihood“ is not the probability of specific points. “probability” pertains to intervals, not specific points, because the probability of any exact value is 0. Instead, we use a probability density function (pdf) to describe the density of probability across an interval. "Likelihood," in this context, refers to the value of the pdf at a specific point, which is crucial for statistical inference but does not represent probability in the traditional sense.

Probability vs. Likelihood

* Likelihood must be at least 0, and can be greater than 1



* Probability : 0 ~ 1

Maximum Likelihood Estimator, (MLE)

◆ Maximum Likelihood Estimator

- A method used for estimating the parameters of a probability distribution
- The goal of MLE is to find the best parameter values for your model that make the observed data most likely to occur

◆ How MLE Works

① **Model Assumption:** assuming a specific form for the probability distribution (e.g., Normal distribution, Binomial distribution) of the data, which is characterized by certain parameters (e.g., mean and variance for Normal distribution).

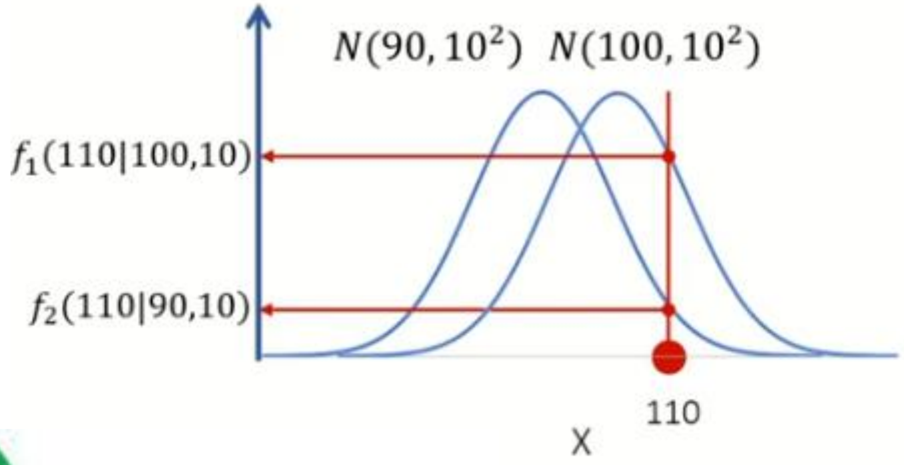
② **Likelihood Function:** constructing a likelihood function, $L(\theta|x)$, which is the probability of observing your data given the parameters θ. This function is essentially the product of the probability densities (or mass functions, for discrete variables) of all observed data points, considering the assumed distribution.

③ **Maximization:** finding the parameter values that maximize the likelihood function. This process often involves taking the logarithm of the likelihood function to convert the product into a sum (making it easier to work with), resulting in what's called the log-likelihood. Then, you differentiate this log-likelihood with respect to the parameters and solve for the values that set the derivatives to zero, adjusting for any constraints (like parameters that must be positive).

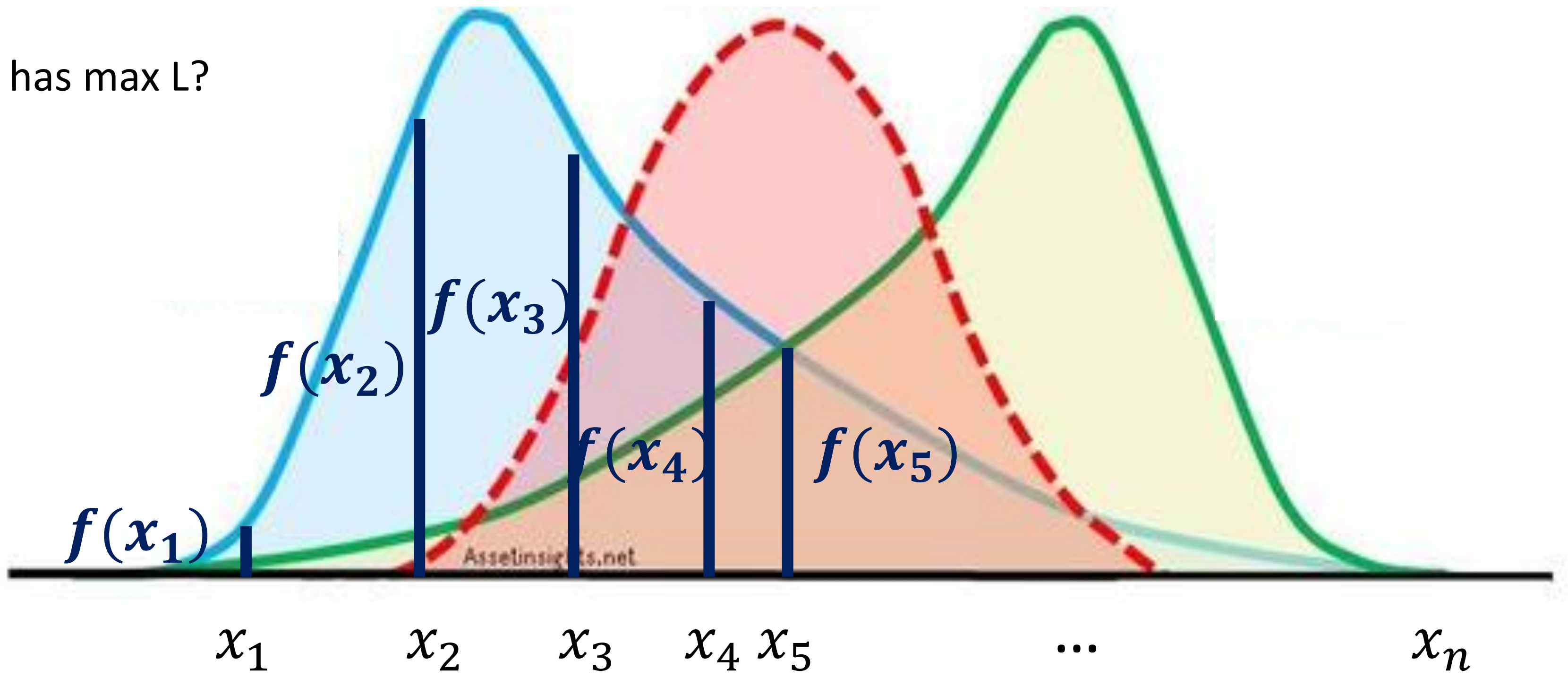
Maximum Likelihood Estimator, (MLE)

◆ **Maximum Likelihood Estimator**

$$L(\theta|\boldsymbol{x}) = f(x_1) \times f(x_2) \times \cdots \times f(x_n)$$
$$= \prod_{i=1}^{n} f(x_i;\theta)$$



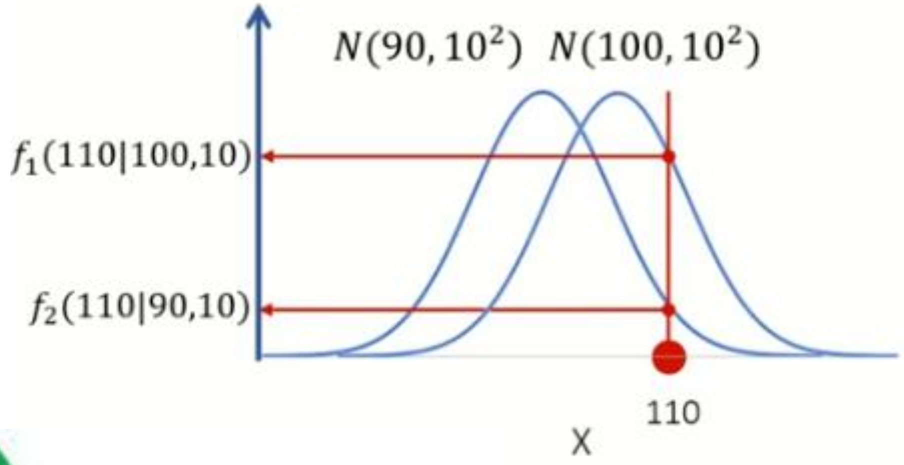
Which distribution has max L?



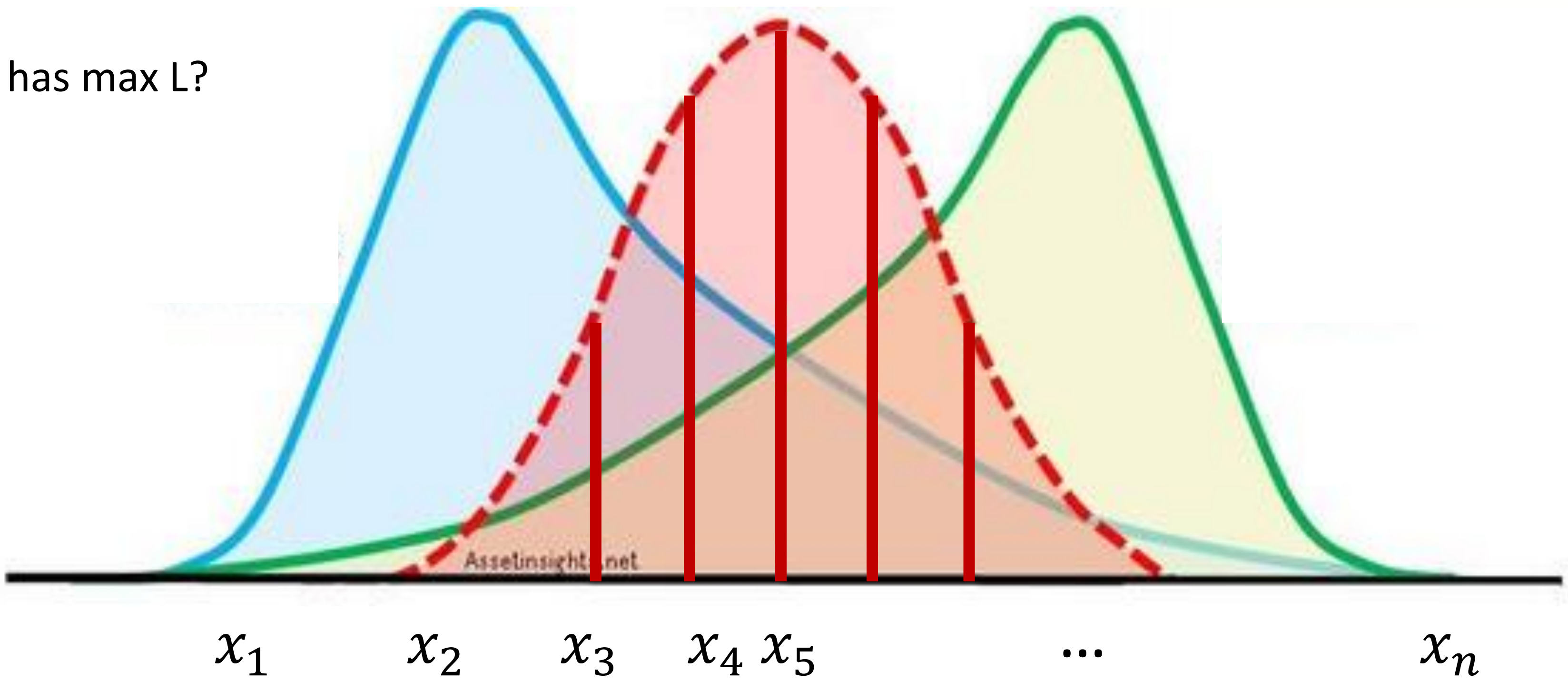
Maximum Likelihood Estimator, (MLE)

◆ **Maximum Likelihood Estimator**

$$L(\theta|\boldsymbol{x}) = f(x_1) \times f(x_2) \times \cdots \times f(x_n)$$
$$= \prod_{i=1}^{n} f(x_i;\theta)$$



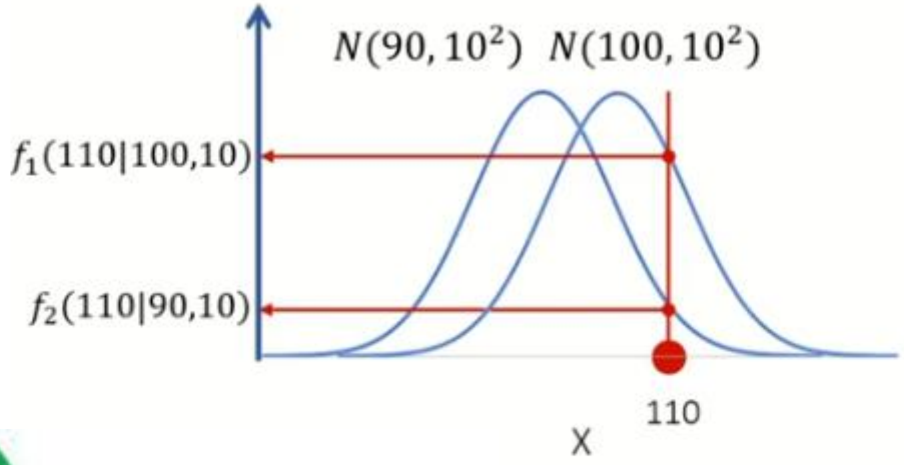
Which distribution has max L?



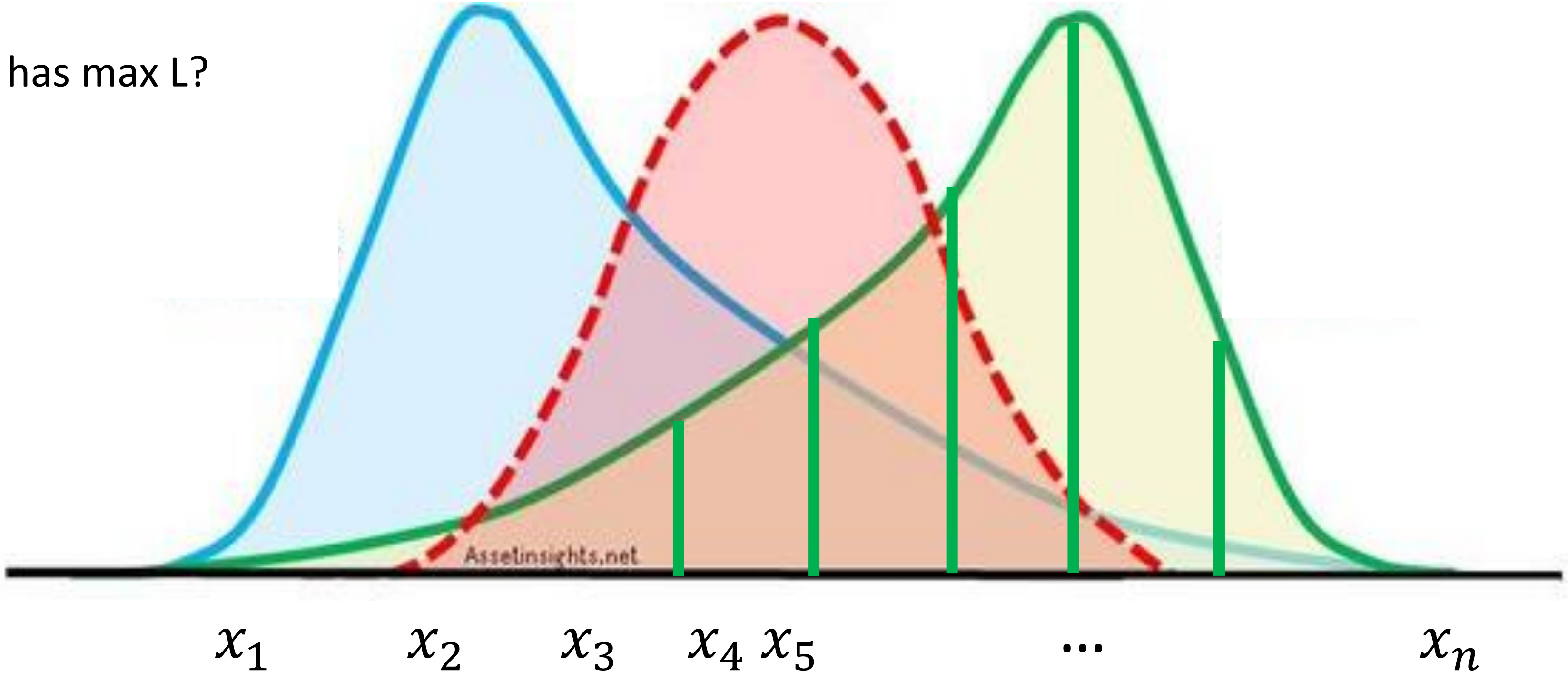
Maximum Likelihood Estimator, (MLE)

◆ **Maximum Likelihood Estimator**

$$L(\theta|\boldsymbol{x}) = f(x_1) \times f(x_2) \times \cdots \times f(x_n)$$
$$= \prod_{i=1}^{n} f(x_i;\theta)$$

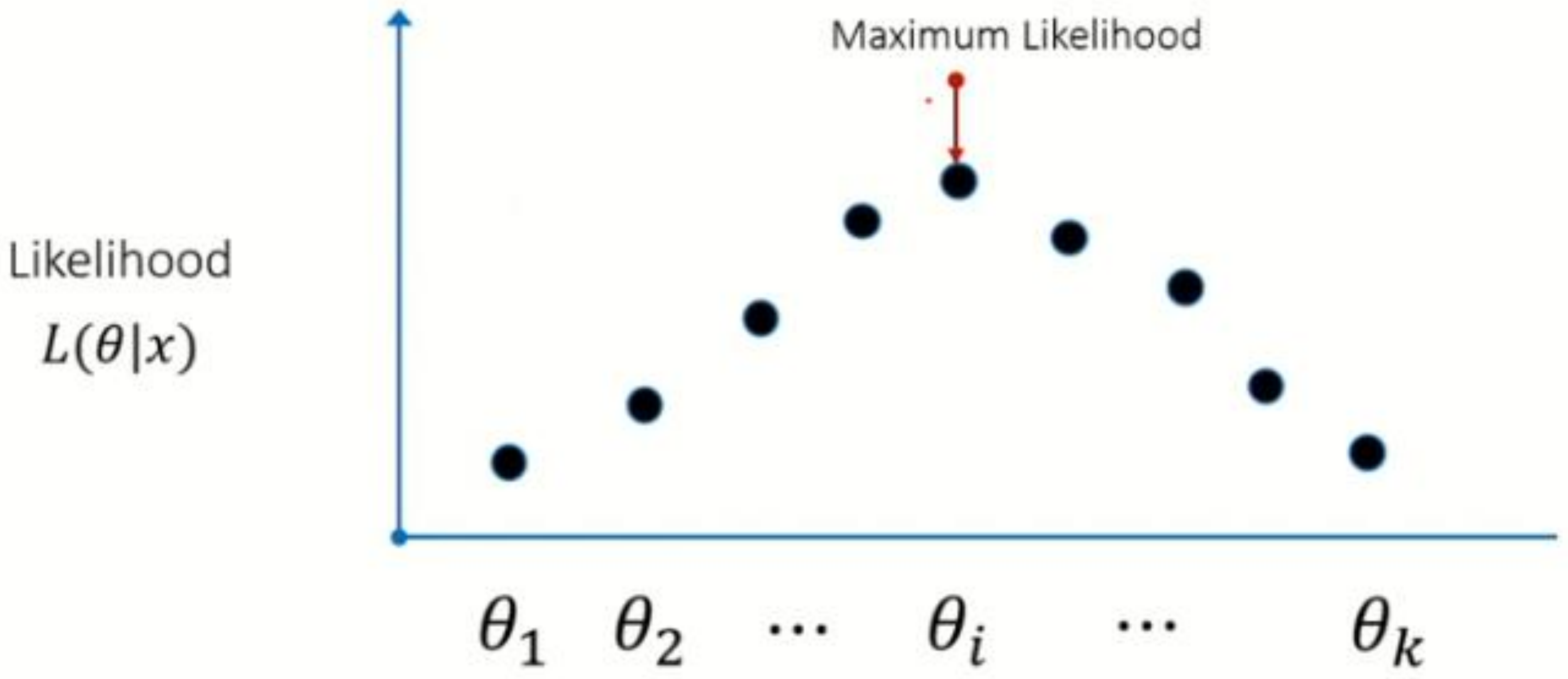


Which distribution has max L?



Maximum Likelihood Estimator, (MLE)

- **Maximum Likelihood Estimator**
  - **Finding theta (=parameters of the distribution) that makes the Likelihood be max!! => MLE**



LAB: MLE

◆ The formula for calculating the likelihood of an observation (measurement) under a normal distribution (Gaussian distribution).

➢ A normal distribution is defined by its mean (μ) and standard deviation (σ), indicating how spread out values are around the mean.

◆ First step : $exponent = -\frac{(measurement-\mu)^2}{2\sigma^2})$

➢ It calculates the exponent by taking the difference between the observation and the mean, squaring it, dividing the result by $2\sigma^2$, and then making it negative.

➢ This step computes the exponential part of the Gaussian function, indicating how far the observation is from the mean (i.e., how unusual it is). The larger this value, the closer the observation is to the mean, and the higher its likelihood.

◆ Second step :

➢ It updates the likelihood using the calculated exponent.

➢ Likeliehood* = $(1/(\sigma\sqrt{2\pi})) \cdot e^{exponent}$ (from PDF)

➢ This formula quantifies how "likely" a given observation is, given a specific mean and standard deviation of a normal distribution. The denominator, $\sigma\sqrt{2\pi}$, serves as a normalization constant to ensure the sum of all probabilities equals 1. $e^{exponent}$ applies an exponential weight to the likelihood based on the previously calculated exponent.

LAB: MLE

```python
import math

##### Sample Dataset (10) #####
measurements = [2.1, 2.2, 2.0, 2.3, 2.1, 2.2, 2.1, 2.4, 2.3, 2.2]

##### Function: :Likelihood calculation #####
def calculate_likelihood(data, mu, sigma):
    likelihood = 1.0
    for measurement in data:


    return likelihood
```

$$exponent = -\frac{(measurement-\mu)^2}{2\sigma^2})$$

Likelihood= $(1/(\sigma\sqrt{2\pi})) \cdot e^{exponent}$

LAB: MLE

```python
import math

##### Sample Dataset (10) #####
measurements = [2.1, 2.2, 2.0, 2.3, 2.1, 2.2, 2.1, 2.4, 2.3, 2.2]

##### Function: :Likelihood calculation #####
def calculate_likelihood(data, mu, sigma):
    likelihood = 1.0
    for measurement in data:
        exponent = -((measurement - mu) ** 2) / (2 * sigma ** 2)
        likelihood *= (1 / (sigma * math.sqrt(2 * math.pi))) * math.exp(exponent)
    return likelihood
```

$$exponent = -\frac{(measurement - \mu)^2}{2\sigma^2})$$

Likelihood *= $(1/(\sigma\sqrt{2\pi})) \cdot e^{exponent}$

LAB: MLE

```python
### 1. Calculate the likelihood ###
print("### 1. Calculate the likelihood ###")

# Assumed values for mean and standard deviation
mean = 2.2
std_dev = 0.1

# Calculate the Likelihood
likelihood = calculate_likelihood(measurements, mean, std_dev)
print("Likelihood:", likelihood)
```

```
### 1. Calculate the likelihood ###
Likelihood: 1535.2762294855227
```

LAB: MLE

```python
### 2. Calculate the likelihoods by changing mean values ###
print("### 2. Calculate the likelihoods by changing mean values ###")

# Assumed values for standard deviation
mean_list = [2.0, 2.1, 2.2, 2.3, 2.4]
std_dev = 0.1

# Calculate the likelihoods by changing mean values
for mean in mean_list:
    likelihood = calculate_likelihood(measurements, mean, std_dev)
    print(f"Mean: {mean}, Likelihood: {likelihood}")
```

```
### 2. Calculate the likelihoods by changing mean values ###
Mean: 2.0, Likelihood: 2.3382225877603254e-05
Mean: 2.1, Likelihood: 28.119565013714418
Mean: 2.2, Likelihood: 1535.2762294855227
Mean: 2.3, Likelihood: 3.8055692956214573
Mean: 2.4, Likelihood: 4.282604055890112e-07
```

LAB: MLE

```python
### 3. Calculate the likelihoods by changing both mean and std values ###
print("### 3. Calculate the likelihoods by changing both mean and std values ###")

mean_list = [2.0, 2.1, 2.2, 2.3, 2.4]
std_dev_list = [0.1, 0.2, 0.3, 0.4, 0.5]

# Calculate the likelihoods by changing both mean and std values
```

LAB: MLE

```python
### 3. Calculate the likelihoods by changing both mean and std values ###
print("### 3. Calculate the likelihoods by changing both mean and std values ###")

mean_list = [2.0, 2.1, 2.2, 2.3, 2.4]
std_dev_list = [0.1, 0.2, 0.3, 0.4, 0.5]

# Calculate the likelihoods by changing both mean and std values
for mean in mean_list:
    for std_dev in std_dev_list:
        likelihood = calculate_likelihood(measurements, mean, std_dev)
        print(f"Mean: {mean}, Standard Deviation: {std_dev}, Likelihood: {likelihood}")
```

LAB: MLE

```
### 3. Calculate the likelihoods by changing both mean and std values ###
Mean: 2.0, Standard Deviation: 0.1, Likelihood: 2.3382225877603254e-05
Mean: 2.0, Standard Deviation: 0.2, Likelihood: 2.181458723602369
Mean: 2.0, Standard Deviation: 0.3, Likelihood: 1.1366899530712442
Mean: 2.0, Standard Deviation: 0.4, Likelihood: 0.210614040322775
Mean: 2.0, Standard Deviation: 0.5, Likelihood: 0.039245694837152616
Mean: 2.1, Standard Deviation: 0.1, Likelihood: 28.119565013714418
Mean: 2.1, Standard Deviation: 0.2, Likelihood: 72.23999156132467
Mean: 2.1, Standard Deviation: 0.3, Likelihood: 5.385316291542496
Mean: 2.1, Standard Deviation: 0.4, Likelihood: 0.5052368178928953
Mean: 2.1, Standard Deviation: 0.5, Likelihood: 0.06870635870641578
Mean: 2.2, Standard Deviation: 0.1, Likelihood: 1535.2762294855227
Mean: 2.2, Standard Deviation: 0.2, Likelihood: 196.36865634918325
Mean: 2.2, Standard Deviation: 0.3, Likelihood: 8.39906583033428
Mean: 2.2, Standard Deviation: 0.4, Likelihood: 0.6487369156209131
Mean: 2.2, Standard Deviation: 0.5, Likelihood: 0.08062765884824176
Mean: 2.3, Standard Deviation: 0.1, Likelihood: 3.8055692956214573
Mean: 2.3, Standard Deviation: 0.2, Likelihood: 43.81576973932554
Mean: 2.3, Standard Deviation: 0.3, Likelihood: 4.312224181175323
Mean: 2.3, Standard Deviation: 0.4, Likelihood: 0.445869926862178
Mean: 2.3, Standard Deviation: 0.5, Likelihood: 0.06342396282259623
Mean: 2.4, Standard Deviation: 0.1, Likelihood: 4.282604055890112e-07
Mean: 2.4, Standard Deviation: 0.2, Likelihood: 0.8025138161774141
Mean: 2.4, Standard Deviation: 0.3, Likelihood: 0.72882330563465
Mean: 2.4, Standard Deviation: 0.4, Likelihood: 0.16402637952920995
Mean: 2.4, Standard Deviation: 0.5, Likelihood: 0.033442975099142944
```

LAB: MLE

```
### 4. Maximum likelihood estimation ###
print("### 4. Maximum likelihood estimation ###")

max_likelihood = -1
best_mean = None
best_std_dev = None

for mean in np.arange(2, 3, 0.001):
    for std_dev in np.arange(0.1, 0.5, 0.001):
        likelihood = calculate_likelihood(measurements, mean, std_dev)
        if likelihood > max_likelihood:
            max_likelihood = likelihood
            best_mean = mean
            best_std_dev = std_dev

print("Maximum Likelihood Estimation:")
print(f"Best Mean: {best_mean}")
print(f"Best Standard Deviation: {best_std_dev}")
print(f"Max Likelihood: {max_likelihood}")
```

LAB: MLE

```python
### 4. Maximum likelihood estimation ###
print("### 4. Maximum likelihood estimation ###")
```

Let's make an MLE!!

➔ Calculate the best mean and standard deviation that best describes the sample data.
= calculate the mean and standard deviation when maxing the likelihood value.

STEP1> Calculate the Likelihood values by changing the mean between 2 and 3 (unit: 0.001)
and the standard deviation between 0.1 and 0.5 (unit: 0.001).

STEP2> Save the largest Likelihood value in max_likelihood, the mean at that time in best_mean, and std_dev in best_std_dev.

```python
print("Maximum Likelihood Estimation:")
print(f"Best Mean: {best_mean}")
print(f"Best Standard Deviation: {best_std_dev}")
print(f"Max Likelihood: {max_likelihood}")
```

* Likelihood must be at least 0, and can be greater than 1

LAB: MLE

```python
### 4. Maximum likelihood estimation ###
print("### 4. Maximum likelihood estimation ###")

max_likelihood = -1
best_mean = None
best_std_dev = None

for mean in np.arange(2, 3, 0.001):
    for std_dev in np.arange(0.1, 0.5, 0.001):
        likelihood = calculate_likelihood(measurements, mean, std_dev)
        if likelihood > max_likelihood:
            max_likelihood = likelihood
            best_mean = mean
            best_std_dev = std_dev

print("Maximum Likelihood Estimation:")
print(f"Best Mean: {best_mean}")
print(f"Best Standard Deviation: {best_std_dev}")
print(f"Max Likelihood: {max_likelihood}")
```

* Likelihood must be at least 0, and can be greater than 1

LAB: MLE

```
### 4. Maximum likelihood estimation ###
Maximum Likelihood Estimation:
Best Mean: 2.1899999999999979
Best Standard Deviation: 0.11400000000000002
Max Likelihood: 1925.8398074486447
```

Bye Bye