

# Machine Learning

## Dirichlet Processes

---

Carl Henrik Ek - [carlhenrik.ek@bristol.ac.uk](mailto:carlhenrik.ek@bristol.ac.uk)

October 29, 2018

<http://www.carlhenrik.com>

## Introduction

---

# Myrtle.ai

Sam Davis (MEng Computer Science 2018)

Lecture: 11th of December

---

<sup>1</sup><https://www.myrtle.ai/>

# Optimisation

- Classic optimisation

$$\hat{x} = \operatorname{argmin}_x f(x)$$

- Much more common

$$\hat{x} = \operatorname{argmin}_x \text{black-box}(x)$$

# Optimisation

---

- in most cases we have an objective function that we do not know

# Optimisation

---

- in most cases we have an objective function that we do not know
- we can test something and see how it works, i.e. evaluate the function

# Optimisation

---

- in most cases we have an objective function that we do not know
- we can test something and see how it works, i.e. evaluate the function
- the number of parameters, possible tests are huge

# Optimisation

---

- in most cases we have an objective function that we do not know
- we can test something and see how it works, i.e. evaluate the function
- the number of parameters, possible tests are huge
- the cost of each test is expensive

# Optimisation

---

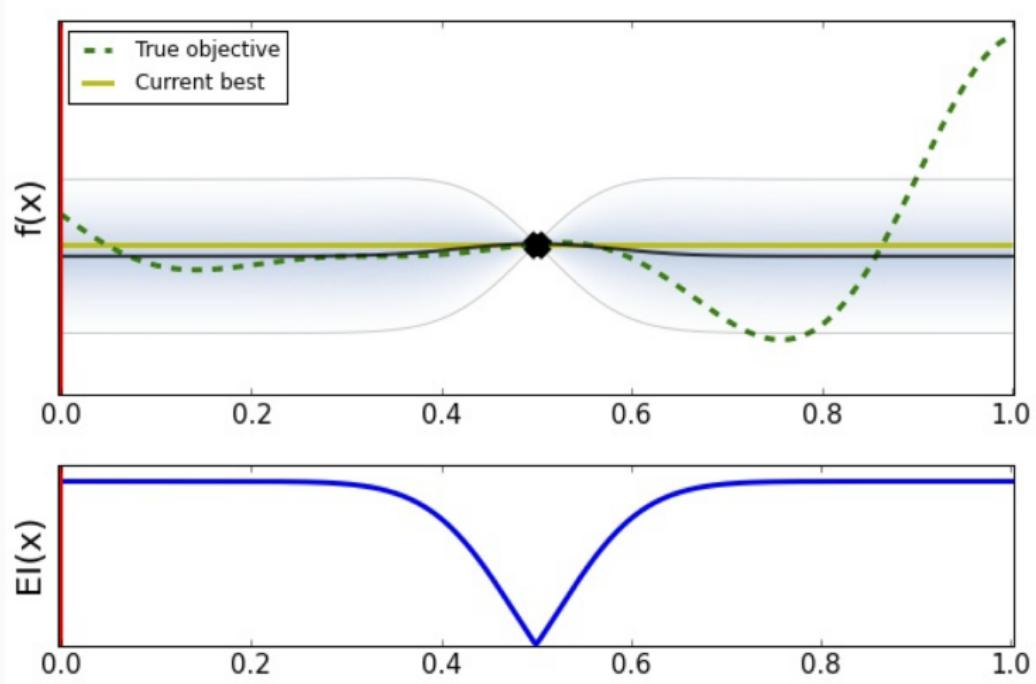
- in most cases we have an objective function that we do not know
- we can test something and see how it works, i.e. evaluate the function
- the number of parameters, possible tests are huge
- the cost of each test is expensive
- the test is noisy

# Optimisation

---

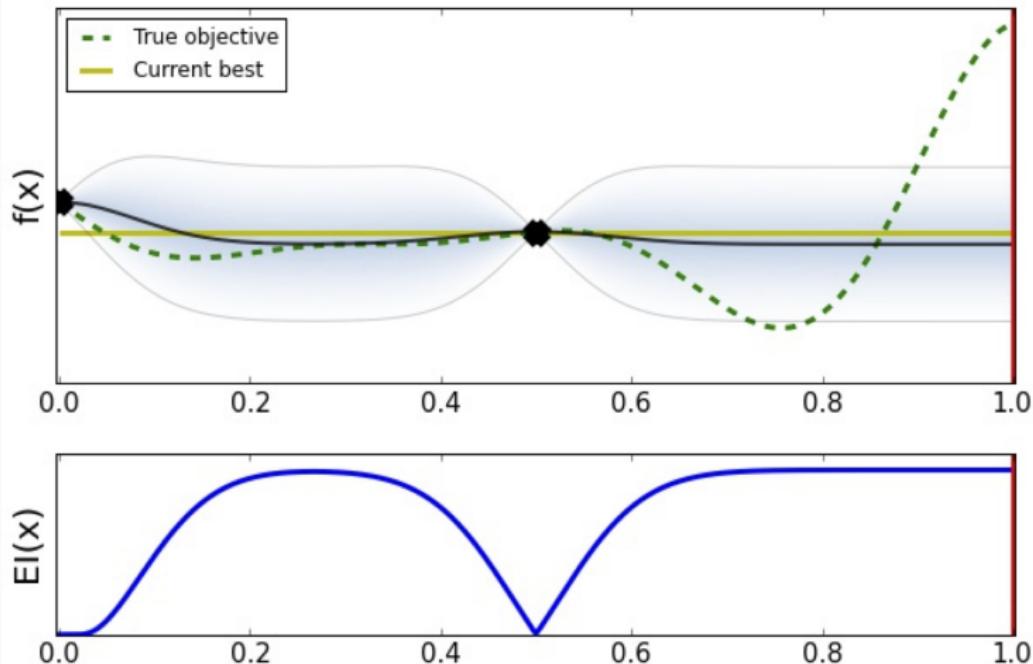
- in most cases we have an objective function that we do not know
- we can test something and see how it works, i.e. evaluate the function
- the number of parameters, possible tests are huge
- the cost of each test is expensive
- the test is noisy
- *can we use machine learning to do this for us?*

# Bayesian Optimisation <sup>2</sup>



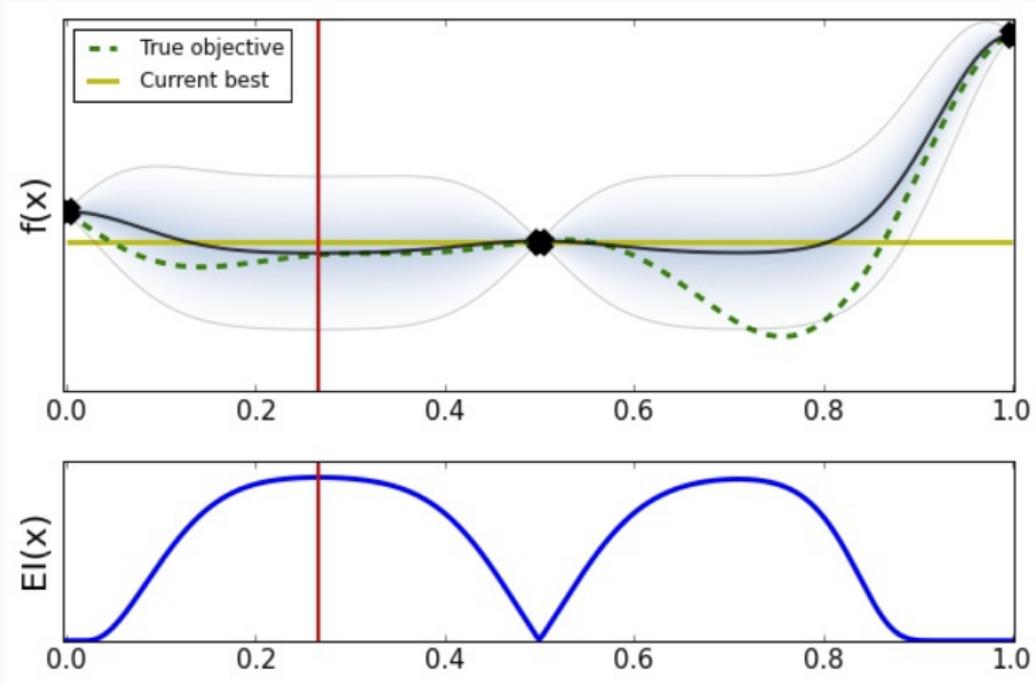
<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation <sup>2</sup>



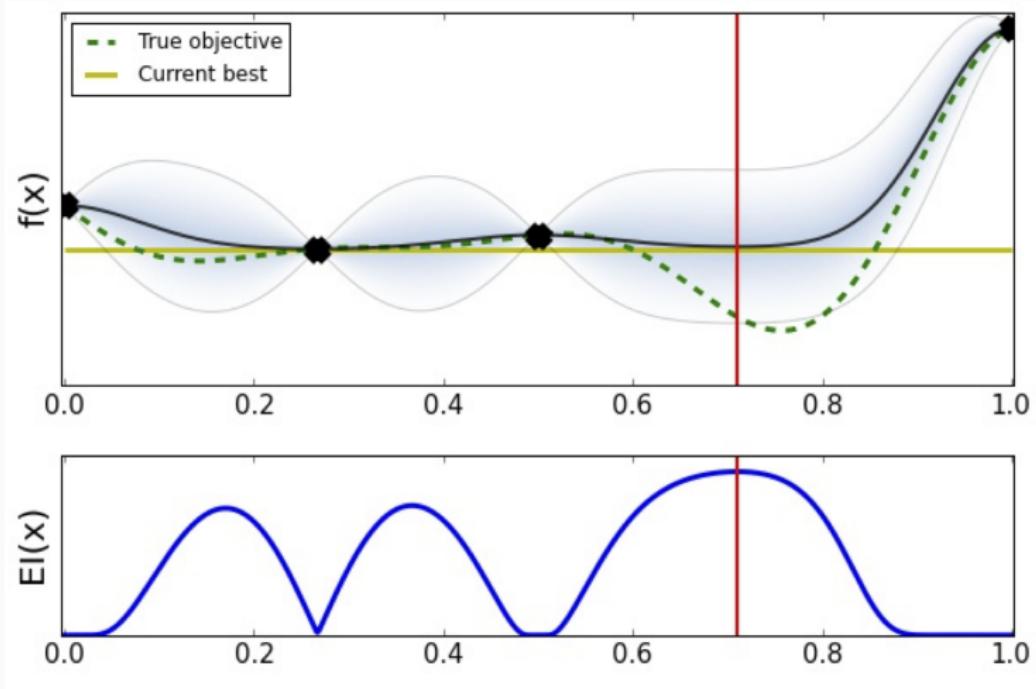
<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation <sup>2</sup>



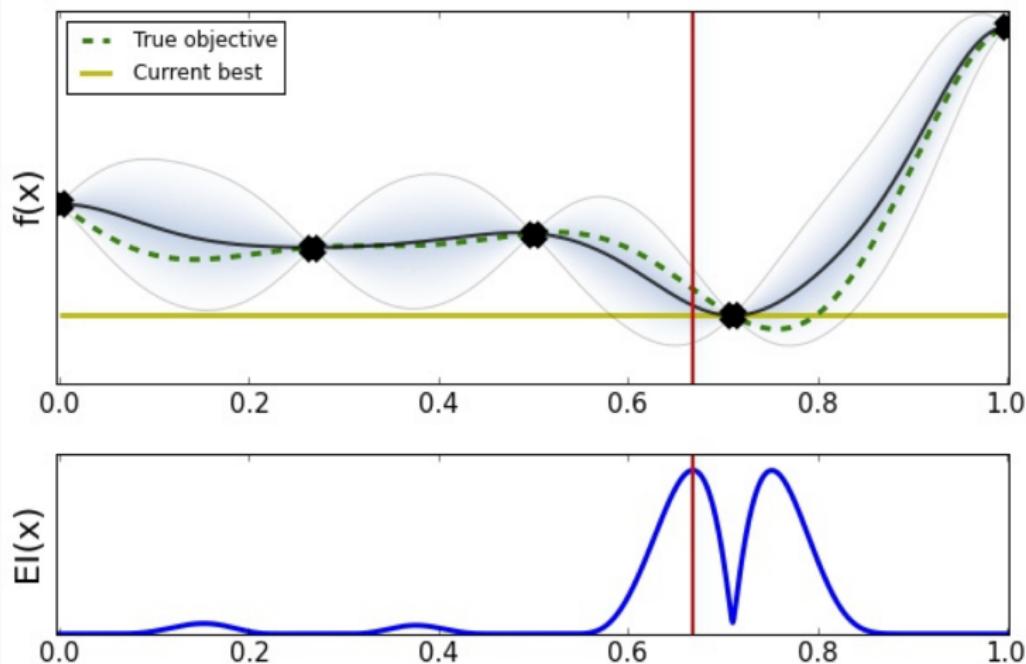
<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation <sup>2</sup>



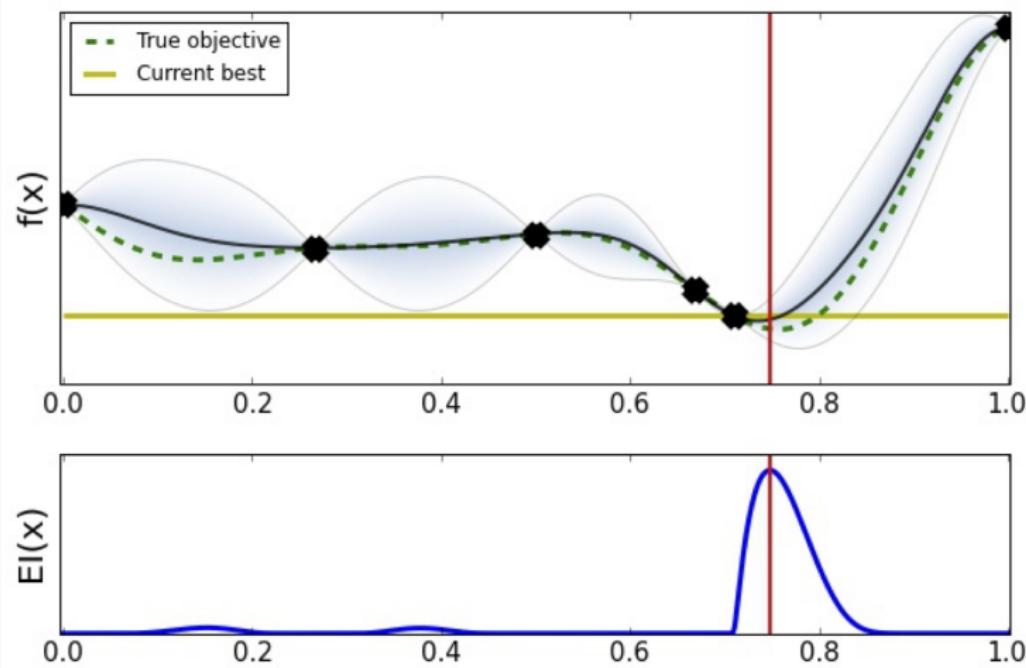
<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation <sup>2</sup>



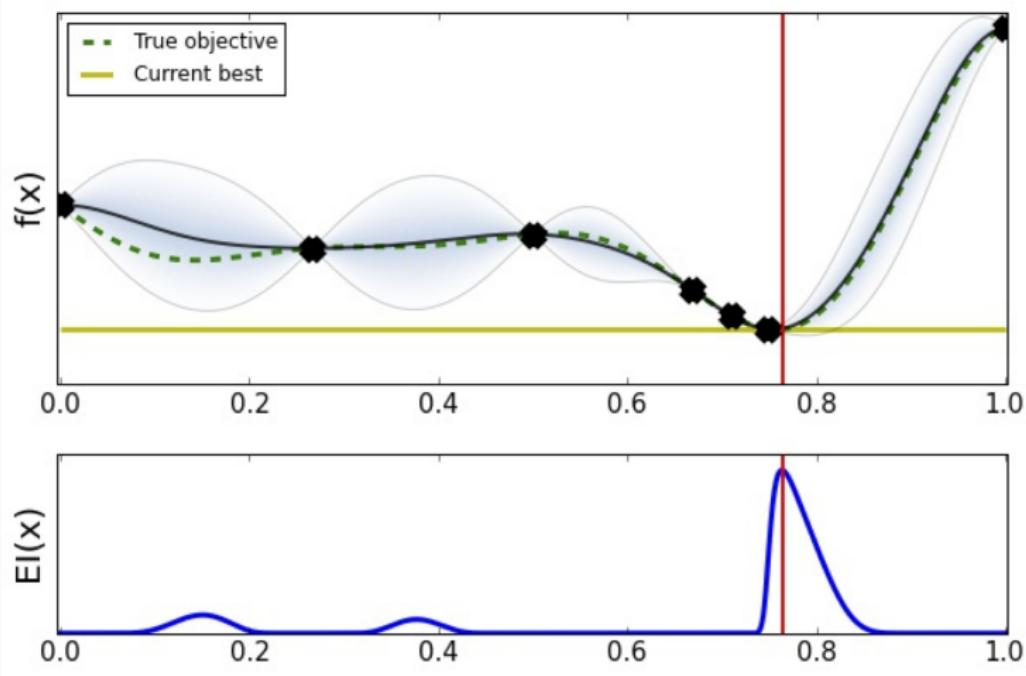
<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation <sup>2</sup>



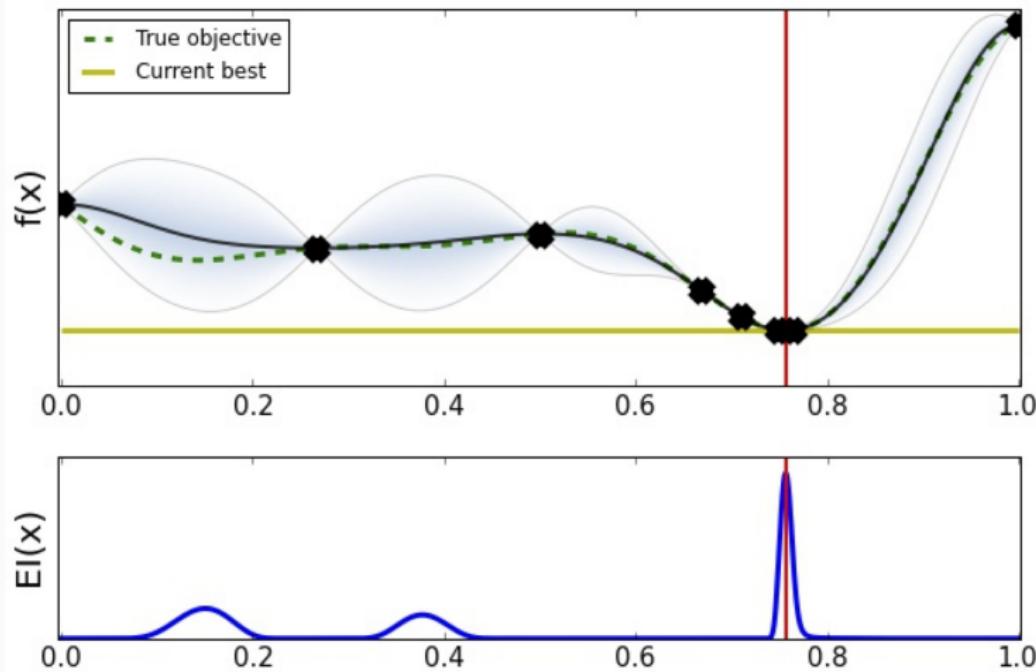
<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation <sup>2</sup>



<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation <sup>2</sup>



<sup>2</sup>Slides courtesy of Javier Gonzales

# Bayesian Optimisation

- We cannot solve the direct problem

$$x_M = \operatorname{argmin}_{x \in \mathcal{X}} f(x)$$

# Bayesian Optimisation

- We cannot solve the direct problem

$$x_M = \operatorname{argmin}_{x \in \mathcal{X}} f(x)$$

- Transform to a series of simpler problems

$$x_{n+1} = \operatorname{argmin}_{x \in \mathcal{X}} \alpha(x; \mathcal{D}_n, \mathcal{M}_n)$$

# Bayesian Optimisation

- We cannot solve the direct problem

$$x_M = \operatorname{argmin}_{x \in \mathcal{X}} f(x)$$

- Transform to a series of simpler problems

$$x_{n+1} = \operatorname{argmin}_{x \in \mathcal{X}} \alpha(x; \mathcal{D}_n, \mathcal{M}_n)$$

- this will work well if

# Bayesian Optimisation

- We cannot solve the direct problem

$$x_M = \operatorname{argmin}_{x \in \mathcal{X}} f(x)$$

- Transform to a series of simpler problems

$$x_{n+1} = \operatorname{argmin}_{x \in \mathcal{X}} \alpha(x; \mathcal{D}_n, \mathcal{M}_n)$$

- this will work well if
  - $\alpha(x)$  is cheap to compute

# Learning

---

## Posterior

---

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta}$$

We integrate our belief  $p(\theta)$  with the evidence in the data  $p(y|\theta)$  and recover an updated belief in  $p(\theta|y)$

## MAP (point estimate)

---

$$\hat{\theta} = \operatorname{argmax}_{\theta} p(\mathbf{y}|\theta)p(\theta) = \operatorname{argmax}_{\theta} \log(p(\mathbf{y}|\theta)) + \log(p(\theta))$$

We find the most likely value of  $\theta$  "regularised" with our prior belief  $p(\theta)$

## ML (point estimate)

---

$$\hat{\theta} = \operatorname{argmax}_{\theta} p(\mathbf{y}|\theta)$$

We find the most likely value of  $\theta$ . If we think of the likelihood as an "error function" we find the parameters that minimises the error between the model and the observed data

## Type-II ML (point estimate)

---

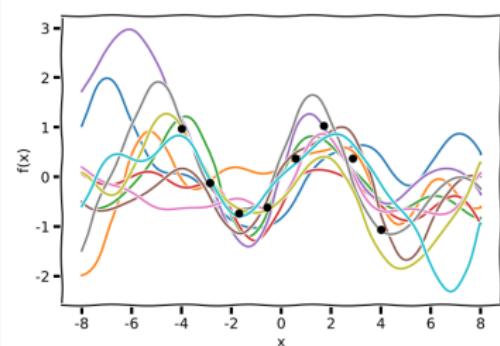
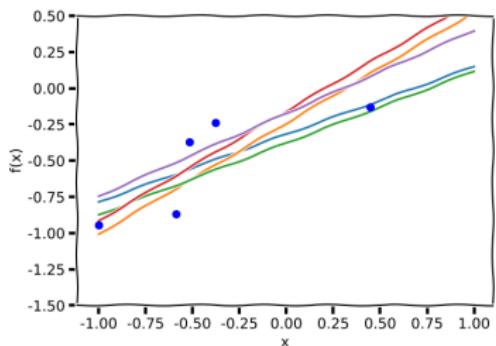
$$\hat{\theta} = \operatorname{argmax}_{\theta} \int p(\mathbf{y}|f, \theta) p(f) df$$

We first integrate our belief in one variable  $f$  with the evidence in the data, then we find the most likely value of the other variable.

## Non-parametrics

---

# Non-parametrics



## Parametric model

- Number of parameters fixed with respect to sample size
- fixed parameter space

## Non-parametric model

- Number of parameters grows with sample size
- $\infty$ -dimensional parameter space

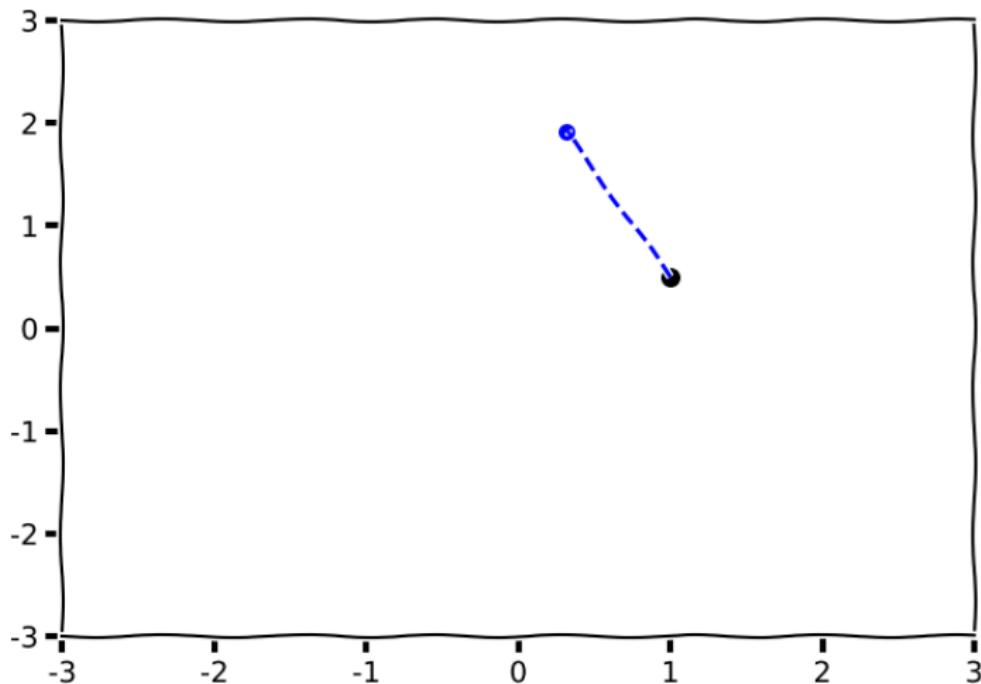
# Nearest Neighbour

- Training data:  $\{\mathbf{x}_i, y_i\}_{i=1}^N$
- Test data:  $\{\mathbf{x}_i\}_{i=1}^M$
- Inference

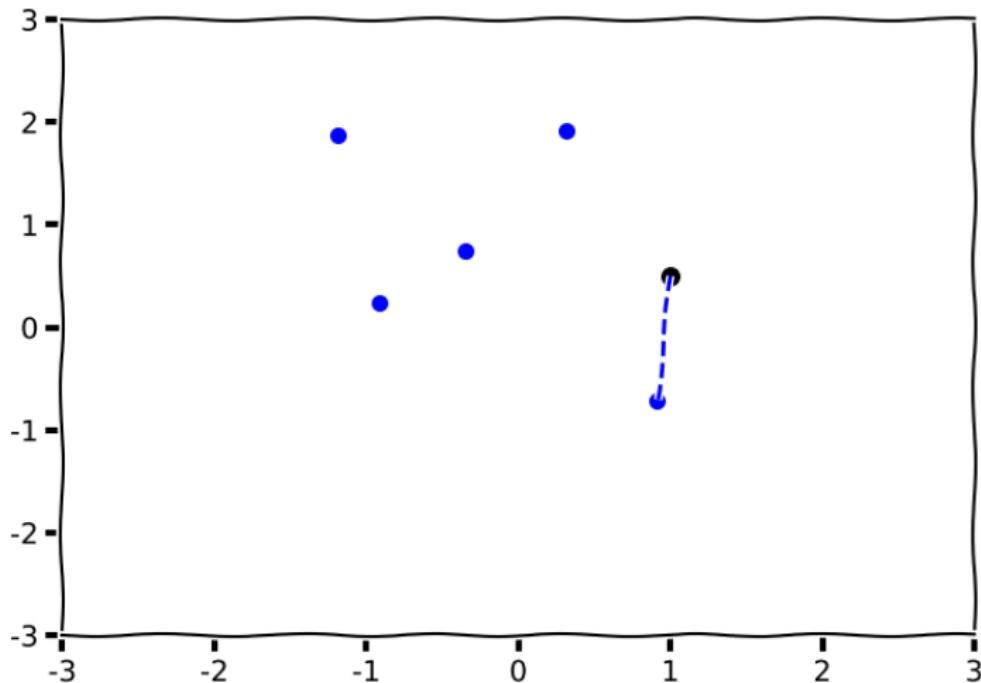
$$\hat{i} = \operatorname{argmin}_i D(\mathbf{x}_*, \mathbf{x}_i)$$

- Complexity grows with number of training data
- Does not generalise at all

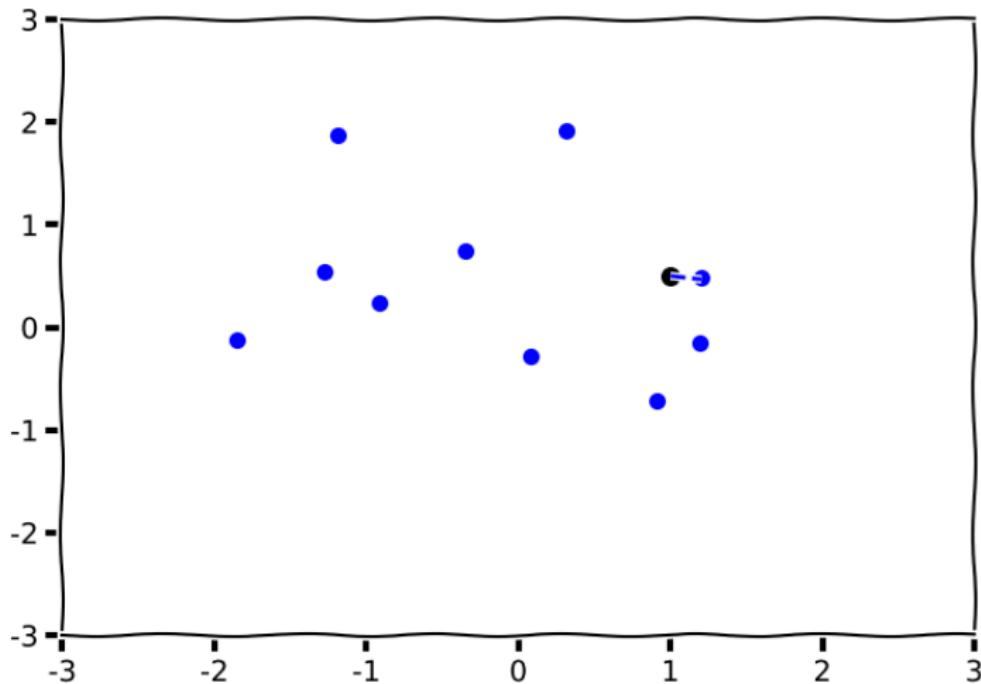
## Nearest Neighbour



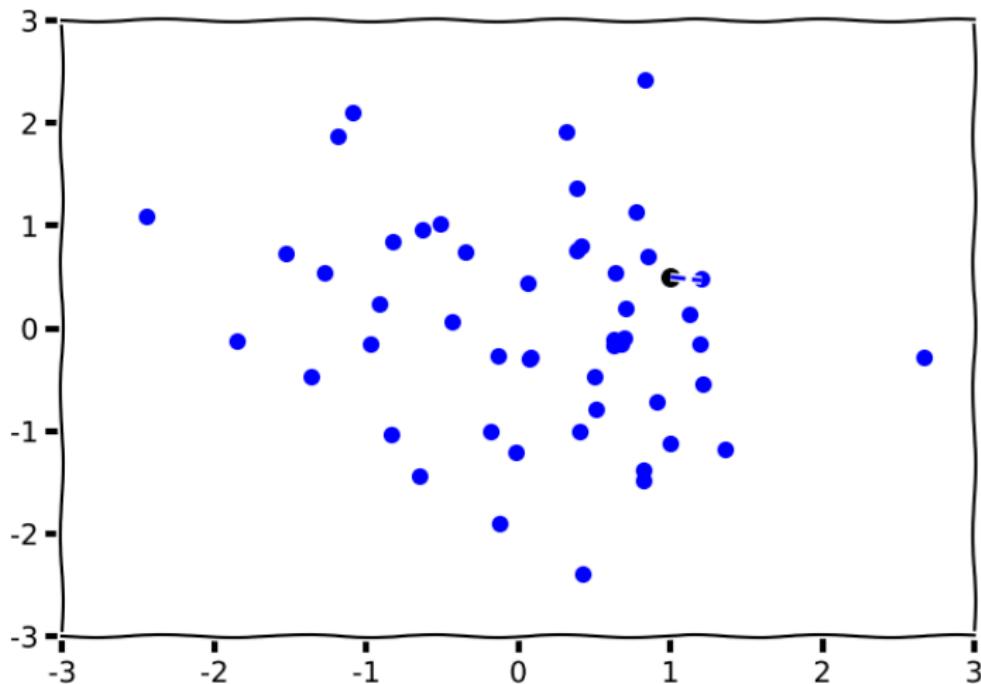
## Nearest Neighbour



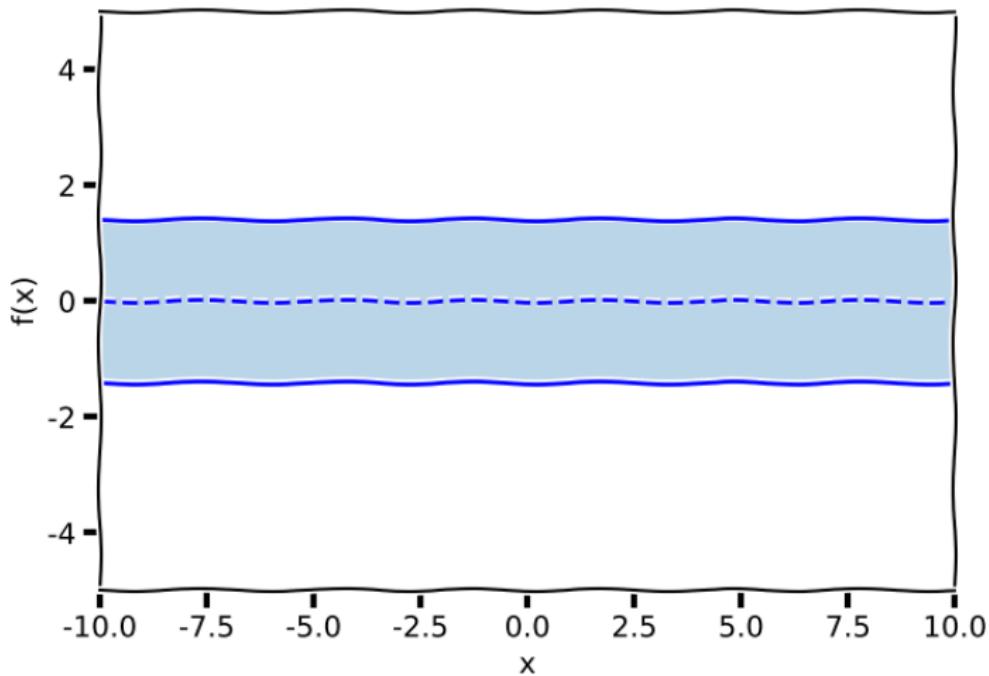
# Nearest Neighbour



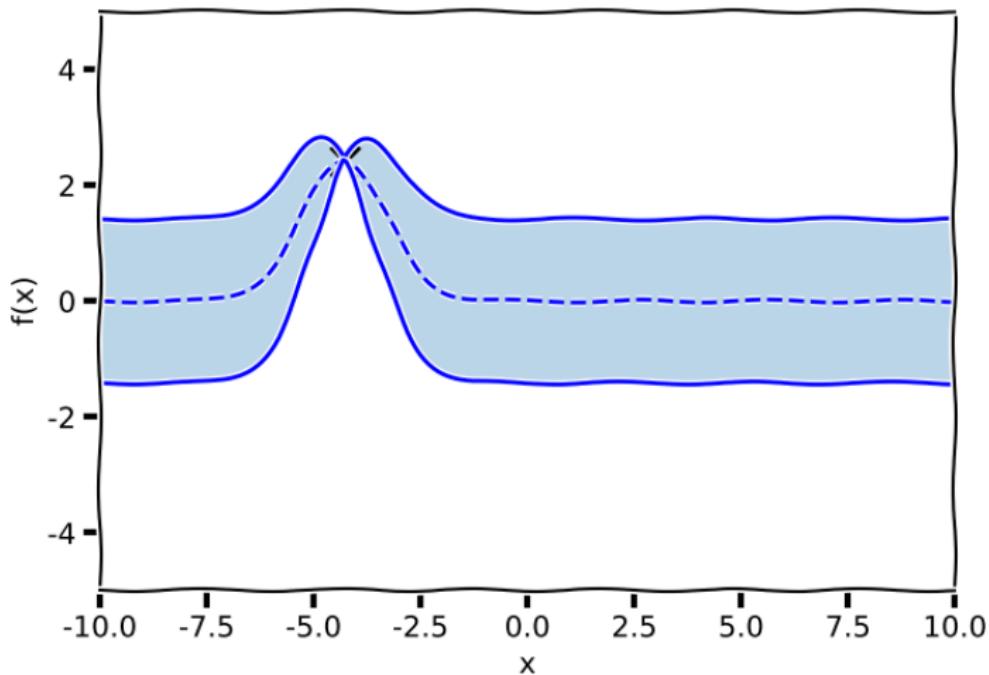
## Nearest Neighbour



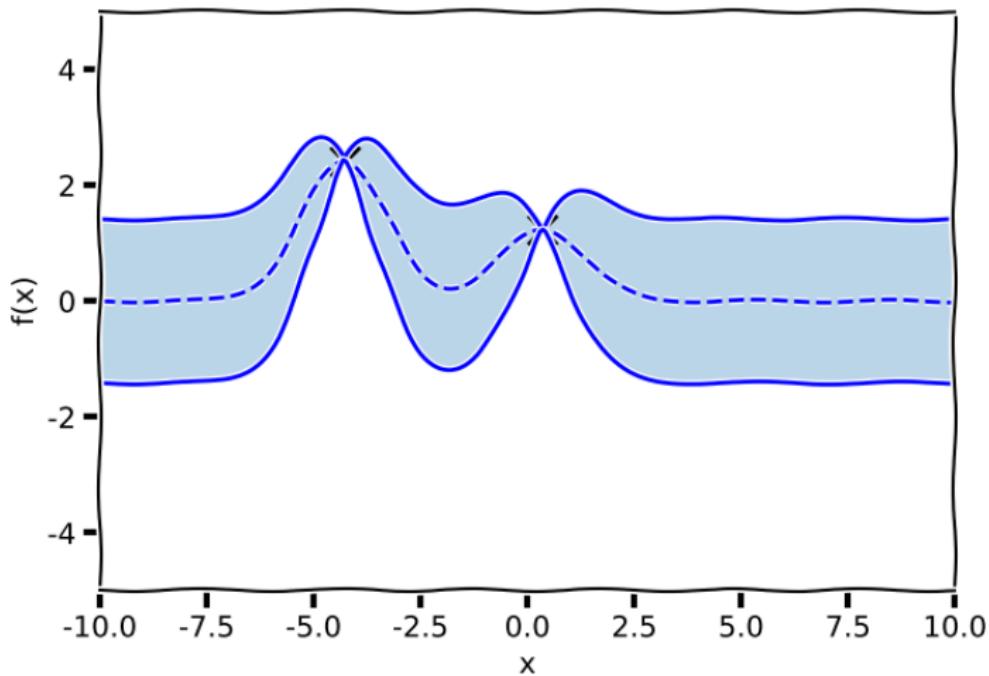
# Gaussian Processes



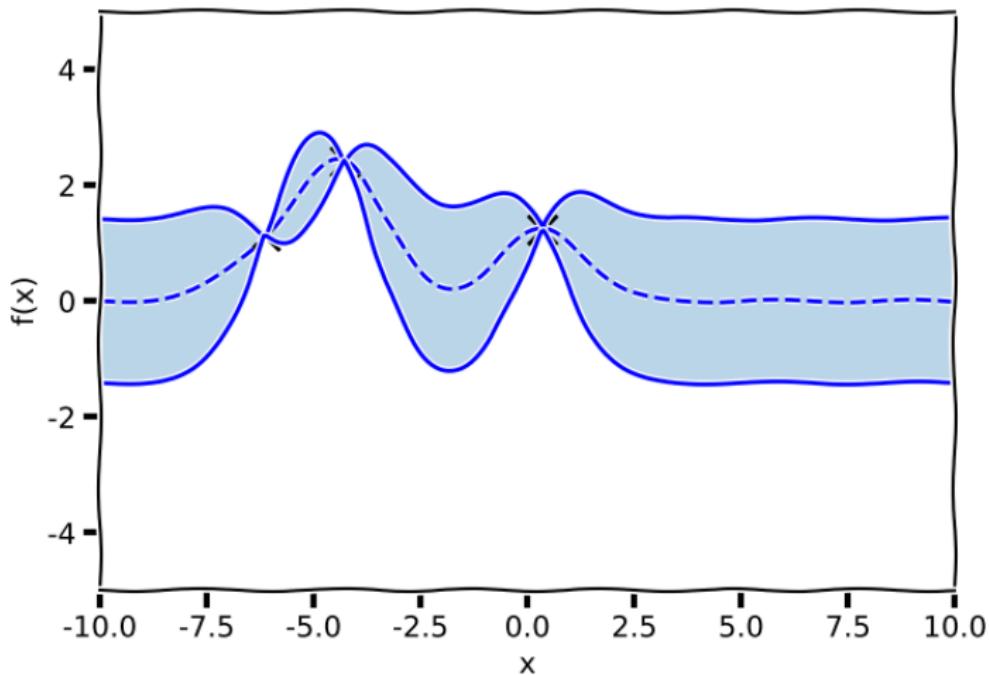
# Gaussian Processes



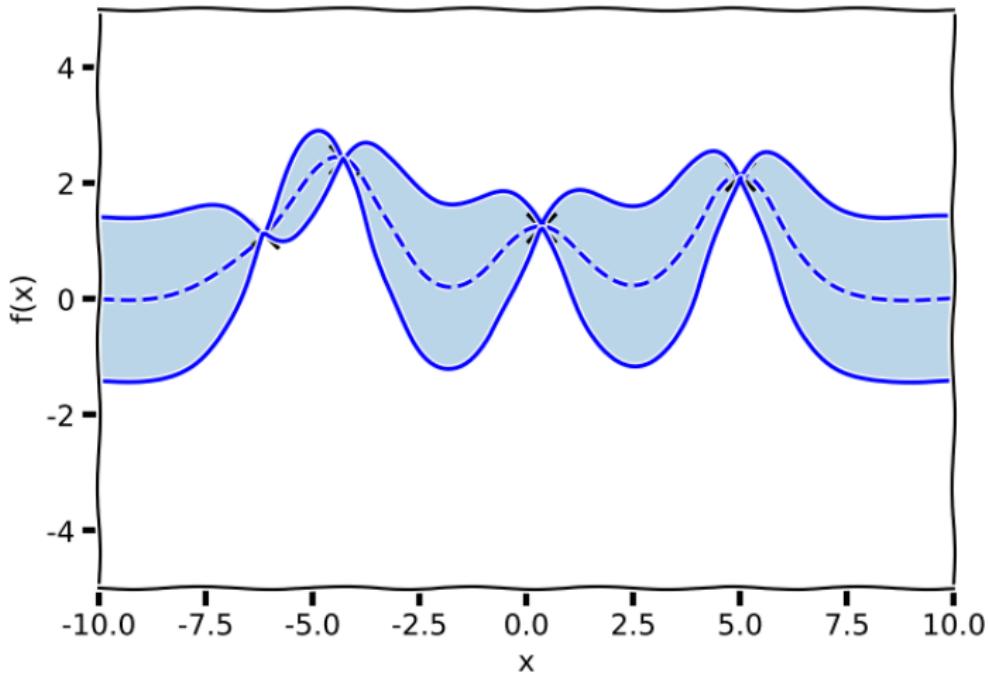
# Gaussian Processes



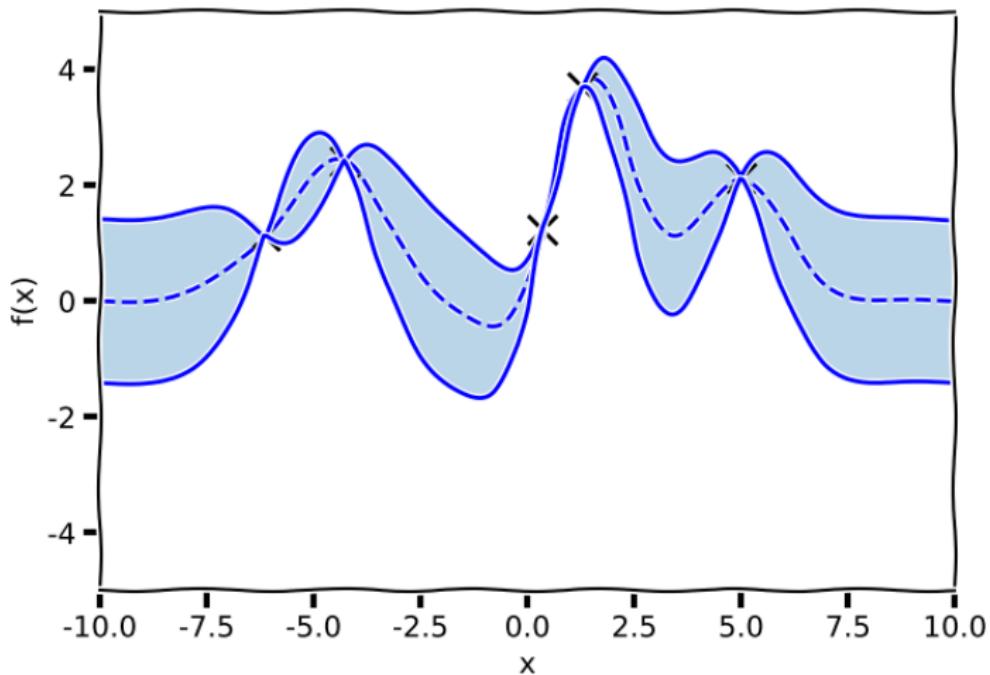
# Gaussian Processes



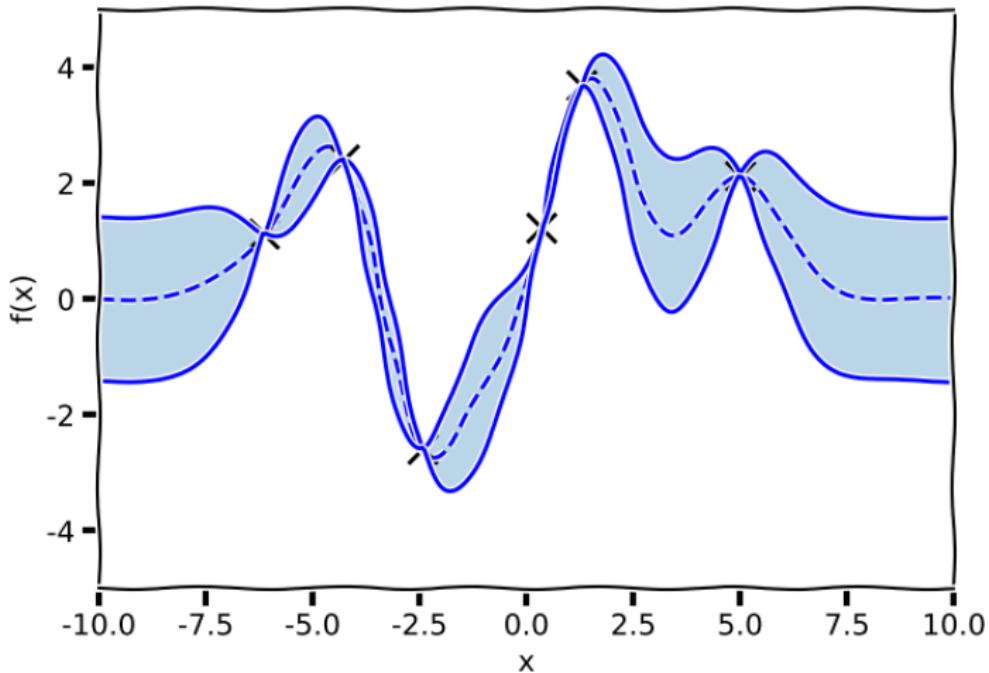
# Gaussian Processes



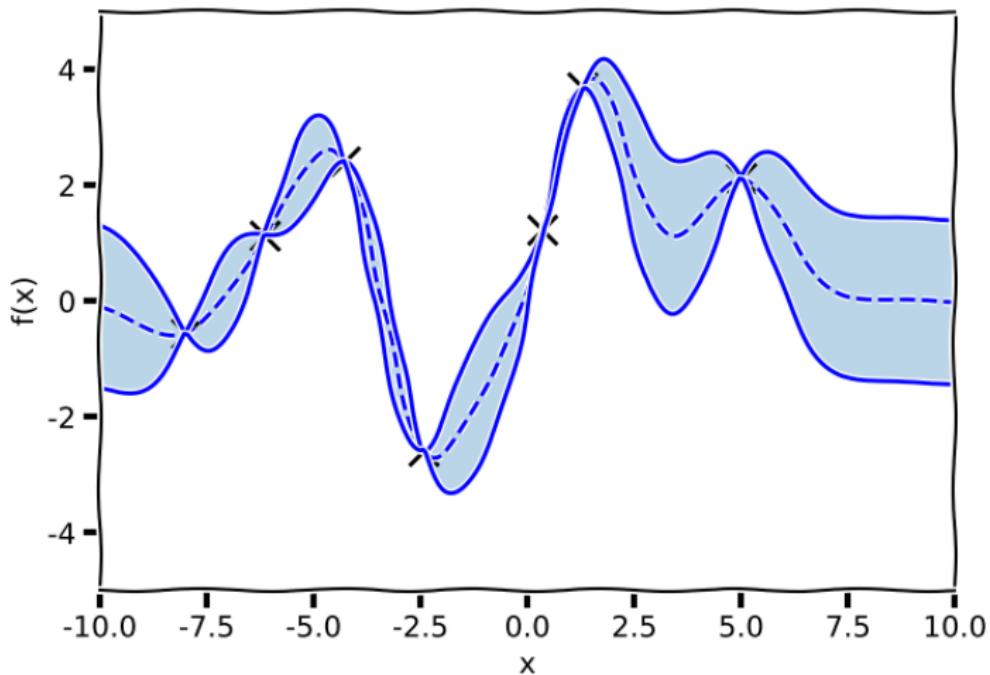
# Gaussian Processes



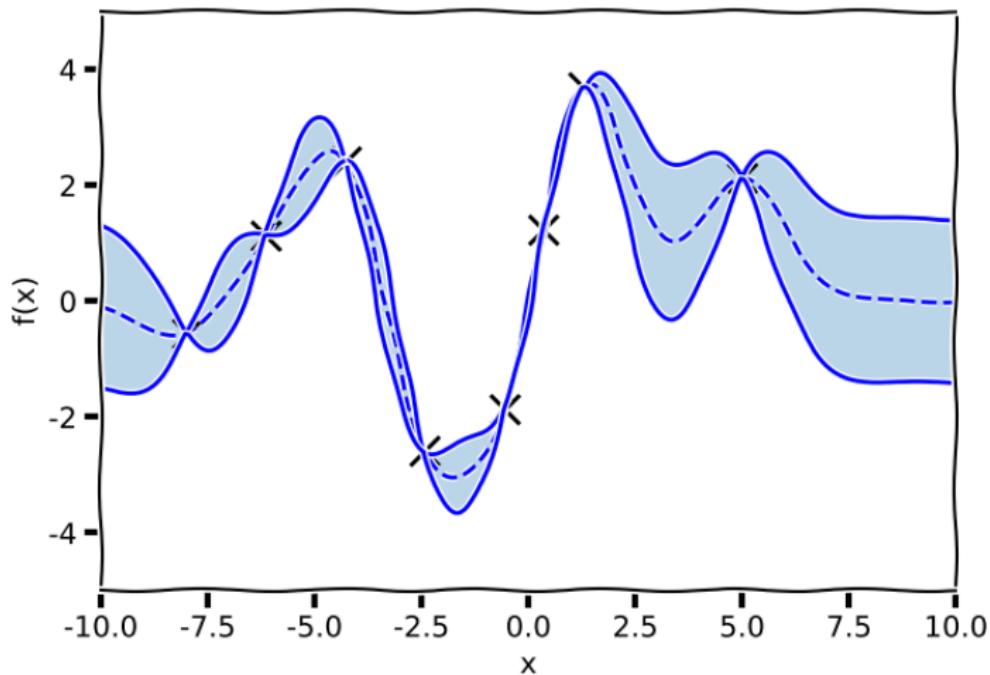
# Gaussian Processes



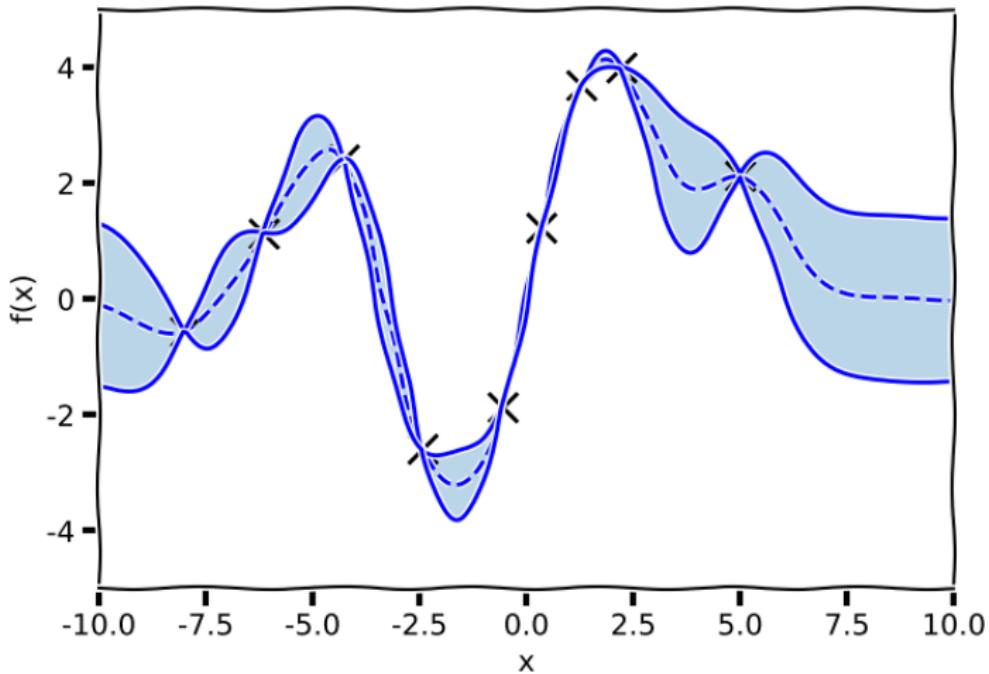
# Gaussian Processes



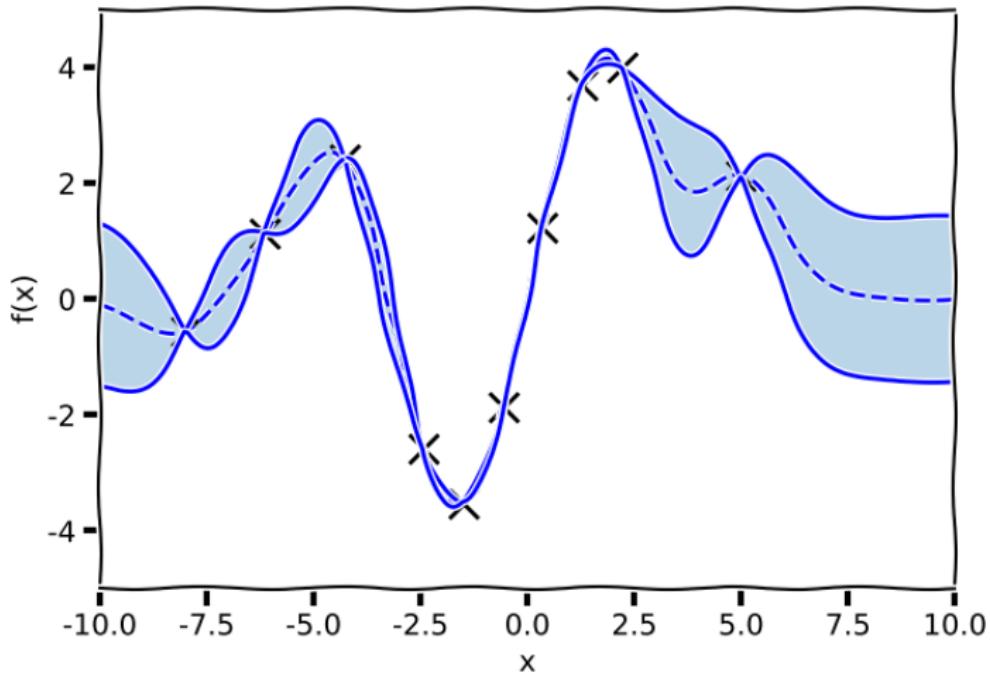
# Gaussian Processes



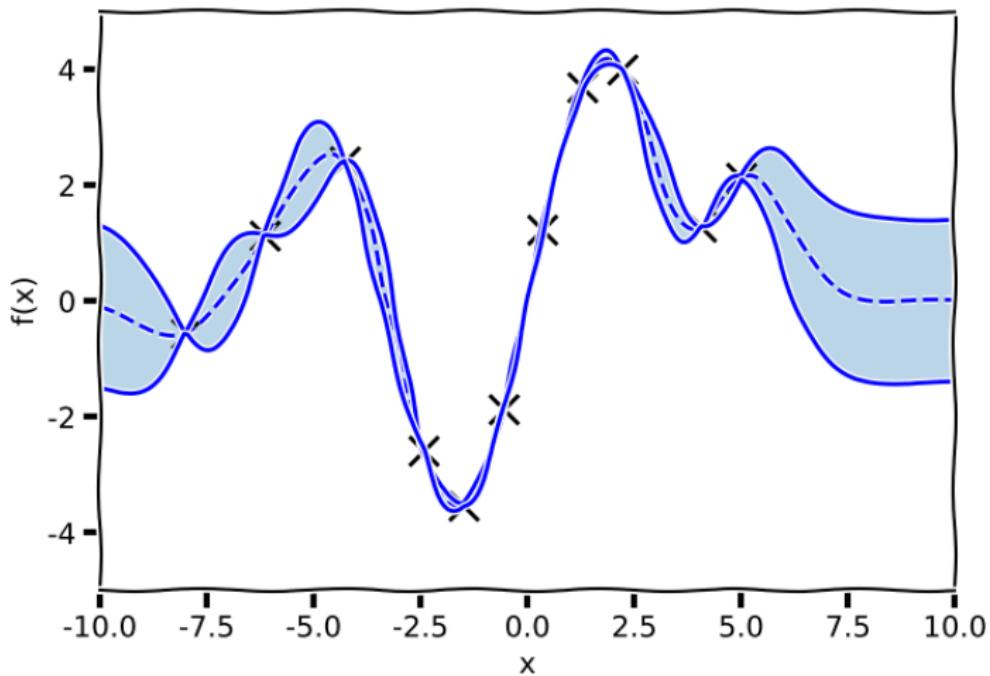
# Gaussian Processes



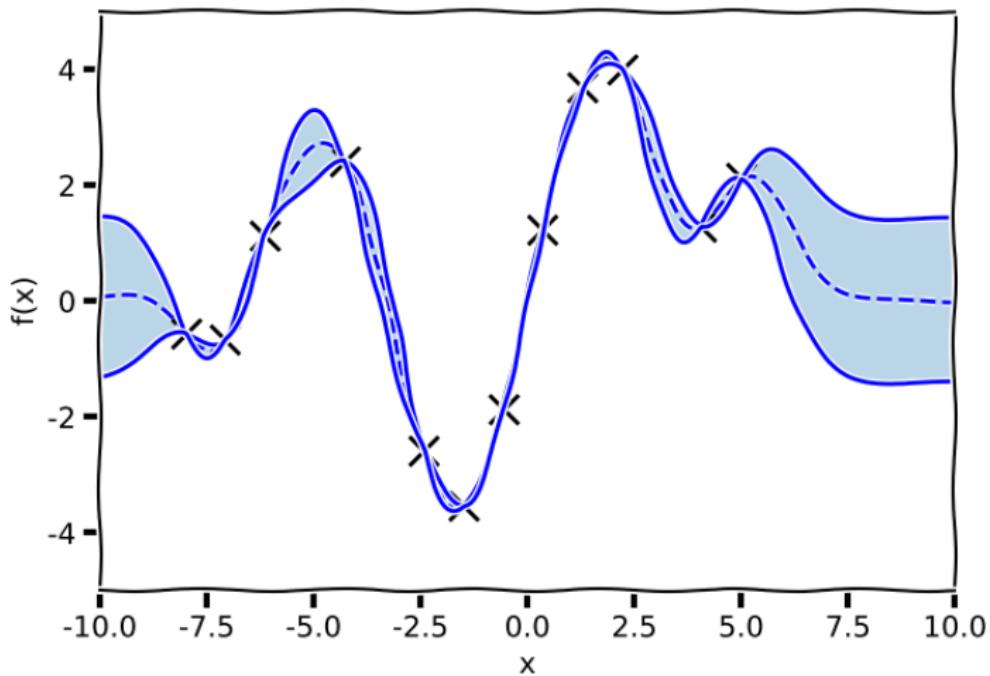
# Gaussian Processes



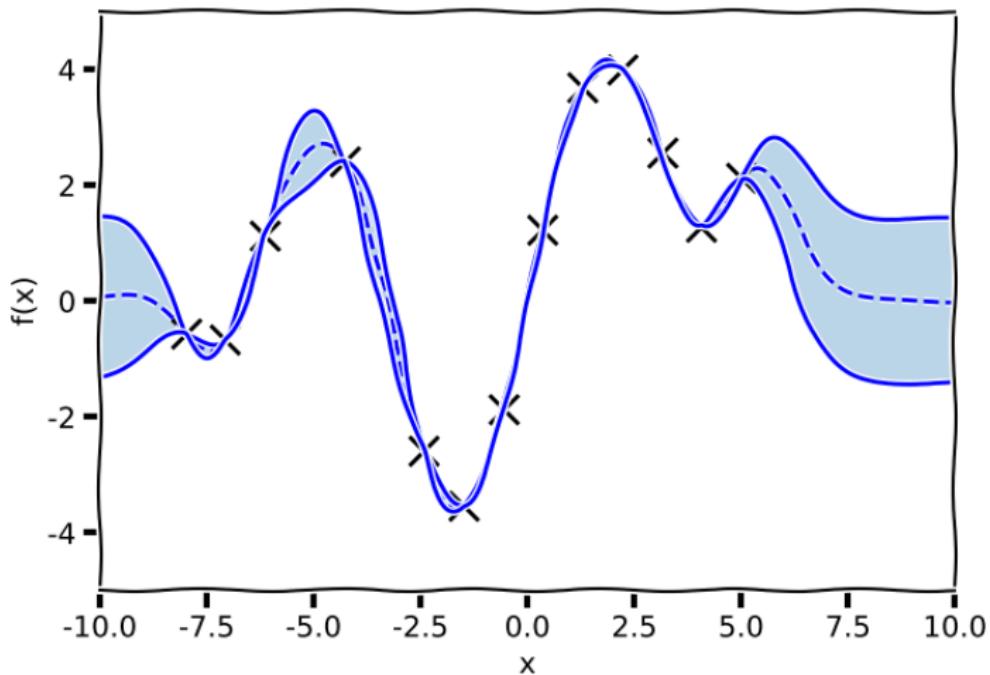
# Gaussian Processes



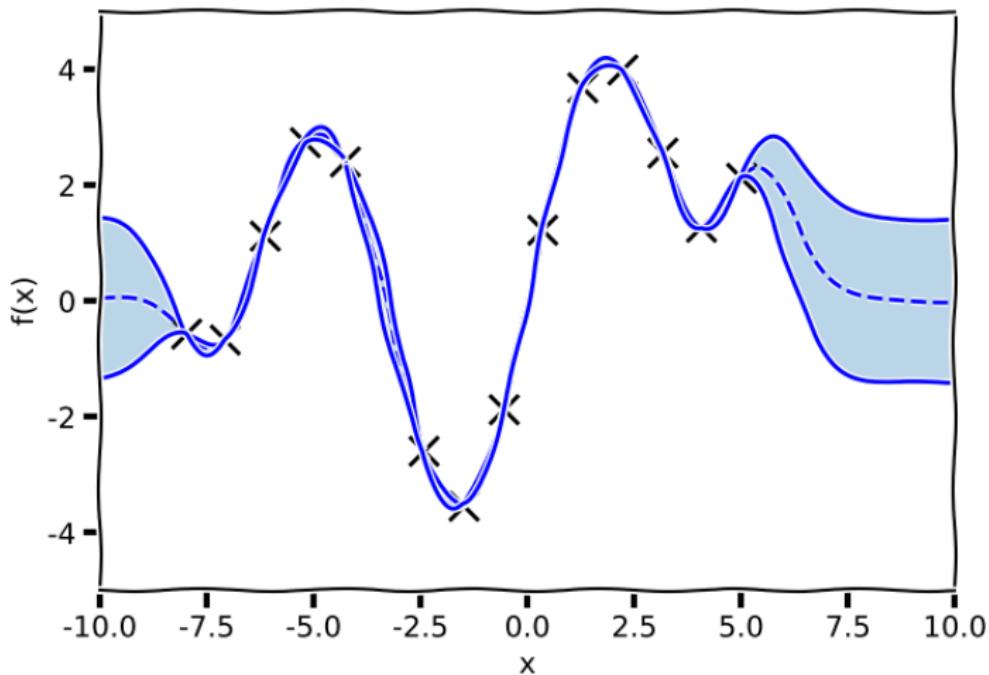
# Gaussian Processes



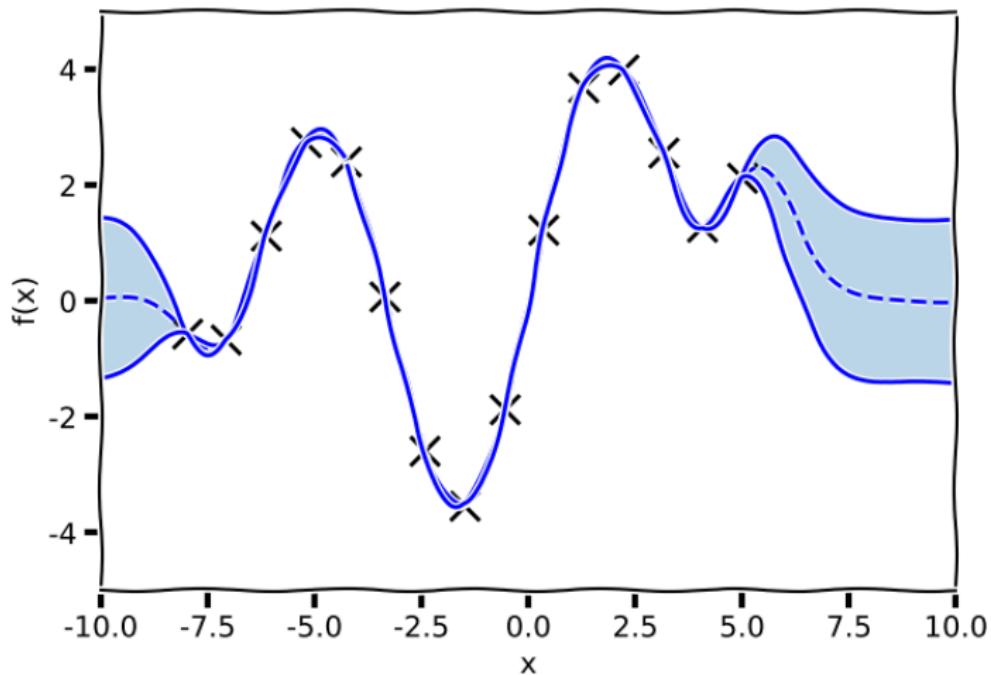
# Gaussian Processes



# Gaussian Processes



# Gaussian Processes



## **Definition (Bayesian Non-parametrics)**

Is a Bayesian model on a  $\infty$ -dimensional parameter space

## Process → Distribution → value

---

- Each evaluation of a process is a distribution

$$\mathcal{N}(0, \Sigma) \sim \mathcal{N}(0, k(\mathbf{X}, \mathbf{X}))$$

- Each evaluation of a process is a distribution

$$\mathcal{N}(0, \Sigma) \sim \mathcal{N}(0, k(\mathbf{X}, \mathbf{X}))$$

- Each evaluation of a distribution is a value

$$y \sim \mathcal{N}(y|0, \Sigma)$$

- Each evaluation of a process is a distribution

$$\mathcal{N}(0, \Sigma) \sim \mathcal{N}(0, k(\mathbf{X}, \mathbf{X}))$$

- Each evaluation of a distribution is a value

$$y \sim \mathcal{N}(y|0, \Sigma)$$

- Parametric models are defined by distributions and non-parametric by processes

# Gaussian Process

---

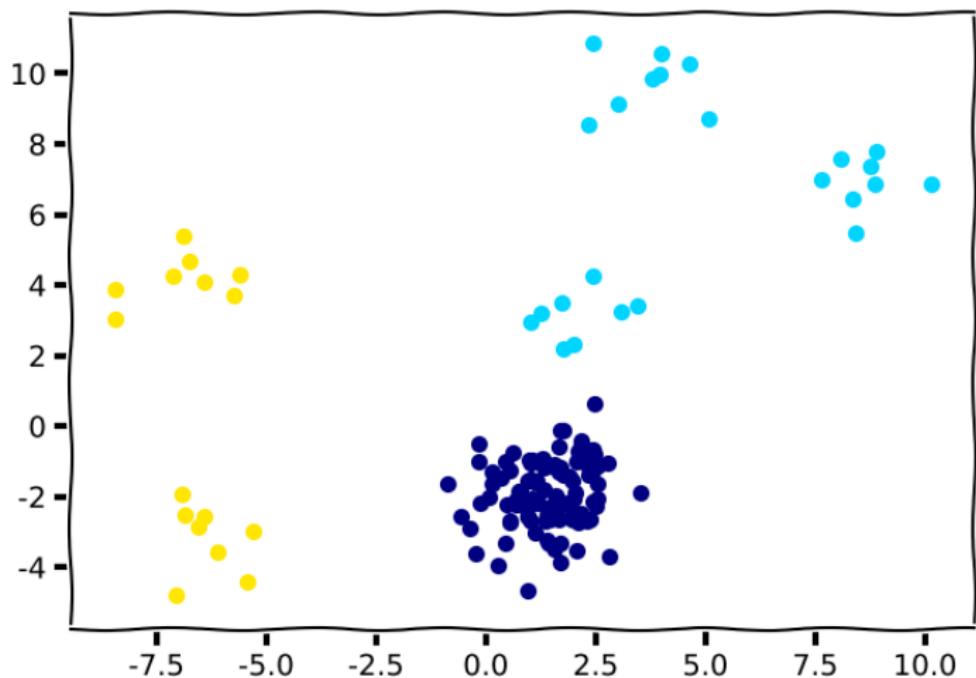
- Formulate process
- Evaluate process at specific location  $x \rightarrow$  distribution
- Evaluate distribution at any location  $y$
- GP is defined over uncountable infinite space

# Gaussian Process

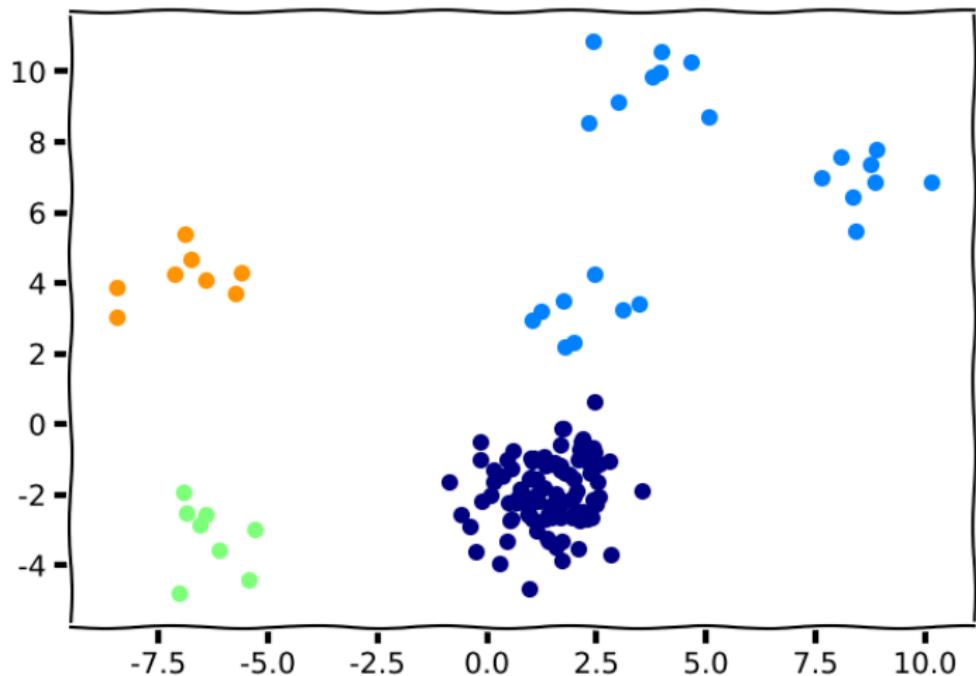
---

- Formulate process
- Evaluate process at specific location  $x \rightarrow$  distribution
- Evaluate distribution at any location  $y$
- GP is defined over uncountable infinite space
- *What about countable objects?*

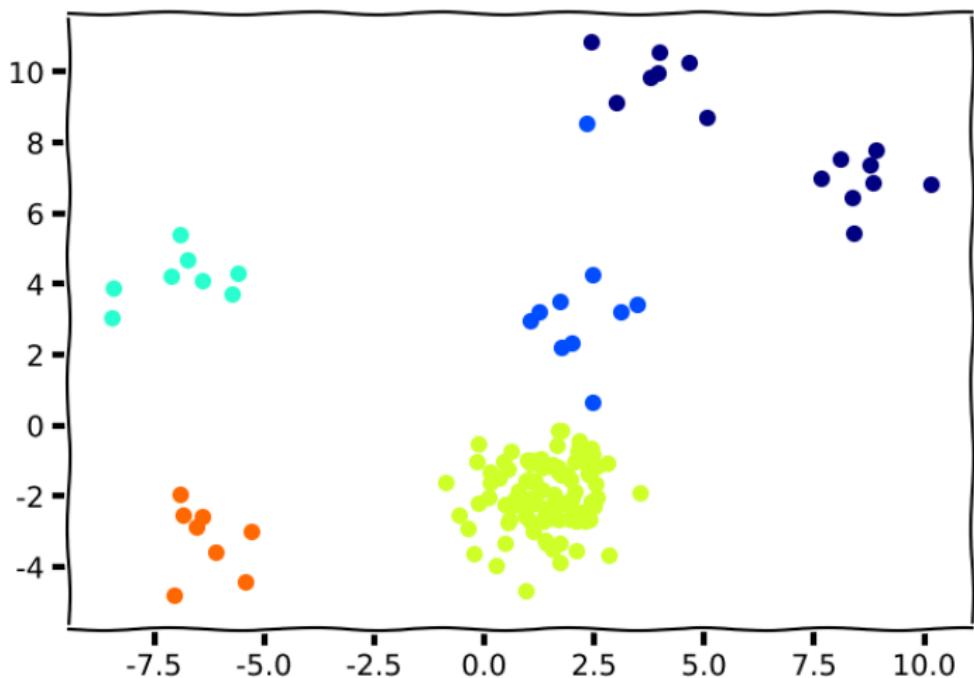
# Gaussian Mixture Model



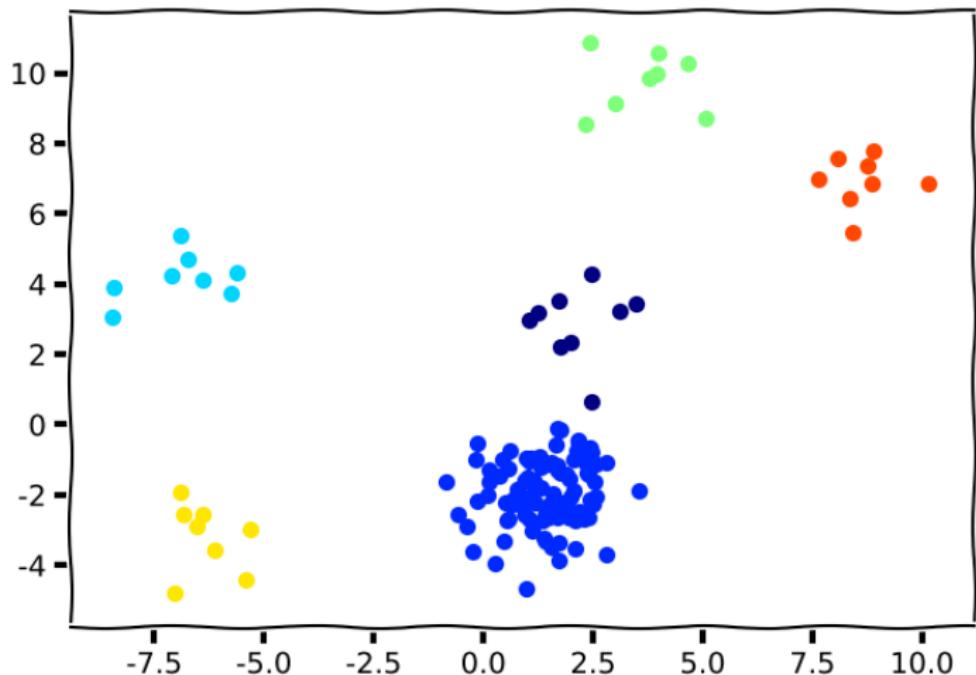
# Gaussian Mixture Model



# Gaussian Mixture Model



# Gaussian Mixture Model



# Gaussian Mixture Models

---

$$p(\mathbf{X}) = \sum_{k=1}^K p(\mathbf{X}|k)p(k) = \sum_{k=1}^K \mathcal{N}(\mathbf{X}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)p(k)$$

- Represent the probability of  $\mathbf{X}$  as a combination or *mixture* of distributions
- What should  $K$  be?
- Can we make  $K$  infinite?

## Gaussian Process

$$p(\mathbf{y}|\mathbf{X}, \theta) = \int p(\mathbf{y}|f)p(f|\mathbf{X}, \theta)df$$

## Infinite Mixture Model

$$p(\mathbf{X}) = \sum_{k=1}^{\infty} p(\mathbf{X}|k)p(k) = \sum_{k=1}^{\infty} \mathcal{N}(\mathbf{X}|\boldsymbol{\mu}_k, \Sigma_k)p(k)$$

# Generative Models

---

- When we build models we describe how the data has been generated

## Unsupervised Linear Regression

1. Sample (pick) a weight

## Clustering

# Generative Models

---

- When we build models we describe how the data has been generated

## Unsupervised Linear Regression

1. Sample (pick) a weight
2. Sample (pick) a input location

## Clustering

# Generative Models

---

- When we build models we describe how the data has been generated

## Unsupervised Linear Regression

1. Sample (pick) a weight
2. Sample (pick) a input location
3. Generate observation

## Clustering

# Generative Models

---

- When we build models we describe how the data has been generated

## Unsupervised Linear Regression

1. Sample (pick) a weight
2. Sample (pick) a input location
3. Generate observation

## Clustering

1. Sample cluster identity

# Generative Models

---

- When we build models we describe how the data has been generated

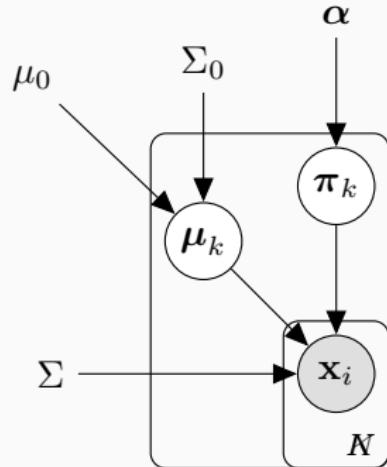
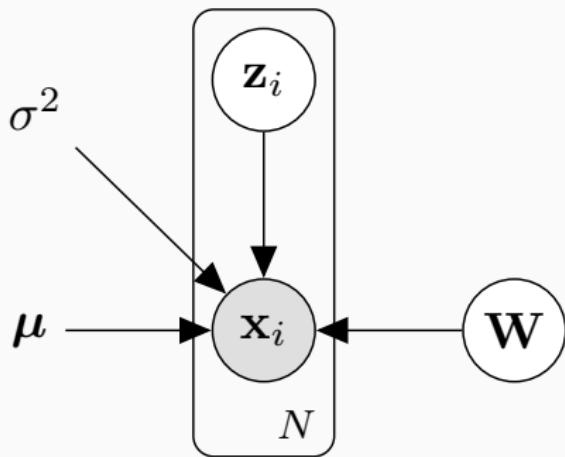
## Unsupervised Linear Regression

1. Sample (pick) a weight
2. Sample (pick) a input location
3. Generate observation

## Clustering

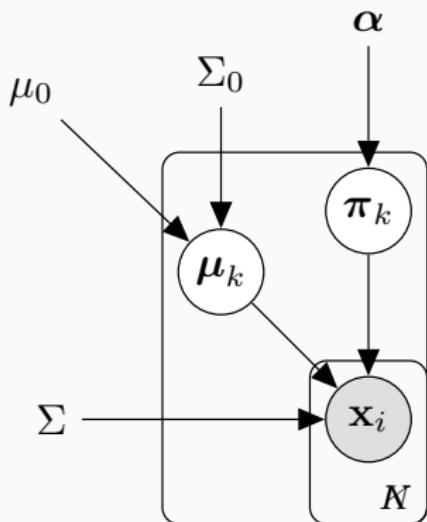
1. Sample cluster identity
2. Sample point from cluster

# Generative Models



- Graphical model clearly shows generative procedure

# Gaussian Mixture Model



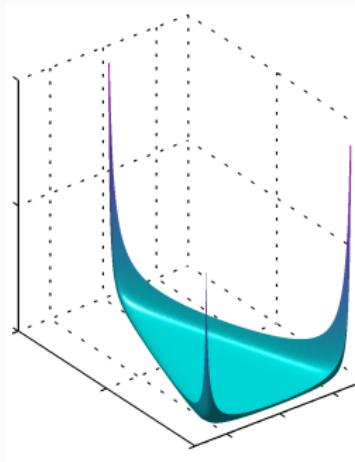
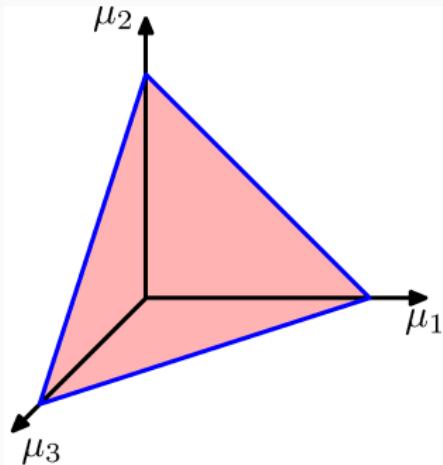
1. Sample proportions
2. Sample cluster id given proportions
3. Sample cluster mean
4. Sample data

## Distributions over partitionings [1] Ch 2.2.0

$$\text{Mult}(m_1, m_2, \dots, m_K | \mu, N) = \binom{N}{m_1, m_2, \dots, m_K} \prod_{k=1}^K \mu^{m_k}$$

- Joint distribution over  $m_1, m_2, \dots, m_K$
- The parameter  $\mu$  says how likely each component is

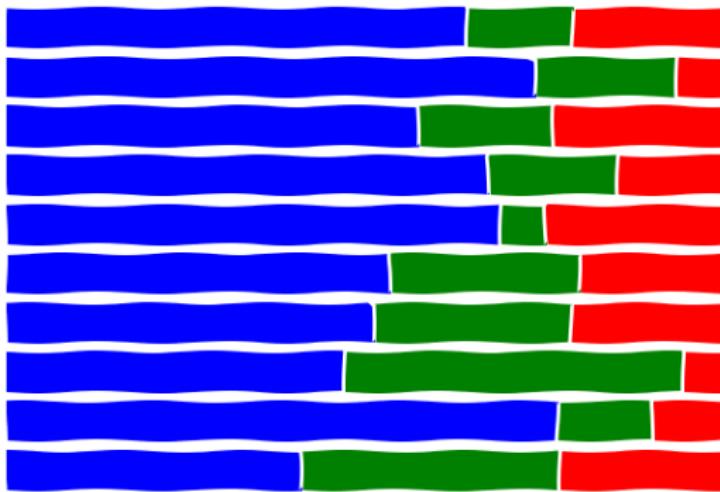
# Dirichlet Distribution



- Conjugate prior to multinomial

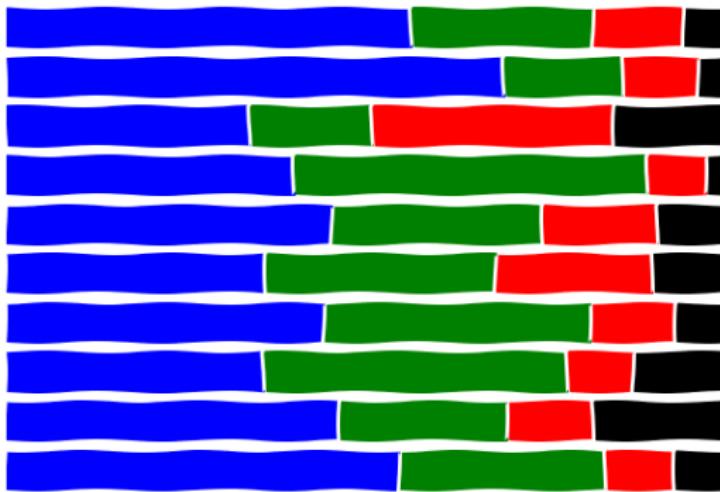
$$\text{Dir}(\boldsymbol{\mu}|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1) \cdot \dots \cdot \Gamma(\alpha_K)} \prod_{k=1}^K \mu_k^{\alpha_k - 1}$$

# Dirichlet Distribution



$\text{Dir}(10, 5, 3)$

# Dirichlet Distribution



$\text{Dir}(7, 5, 3, 2)$

## Dirichlet Processes

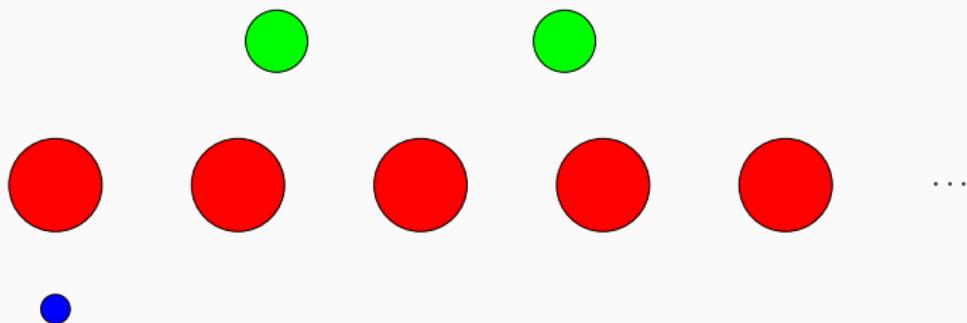
---

## Dirichlet Process

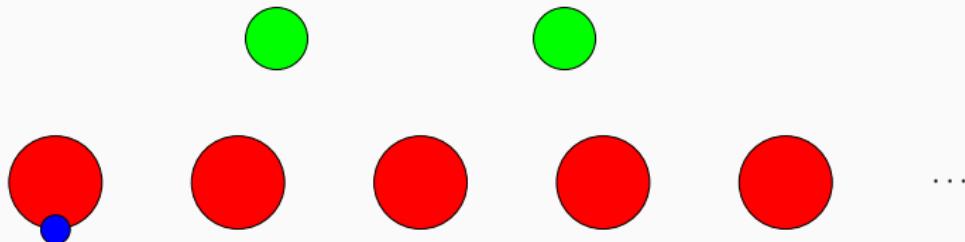
---

- Is the infinite dimensional generalisation of a Dirichlet distribution
  - just as Gaussian process is of Gaussian distribution
- Generates a partitioning of (possibly) infinite number of elements
- Not as intuitive to write down
- Best written constructively

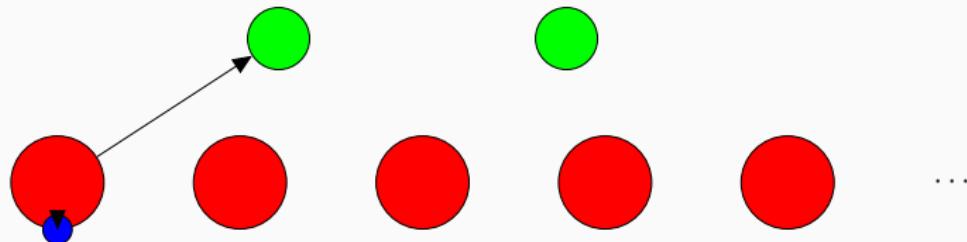
# Chinese Restaurant Process



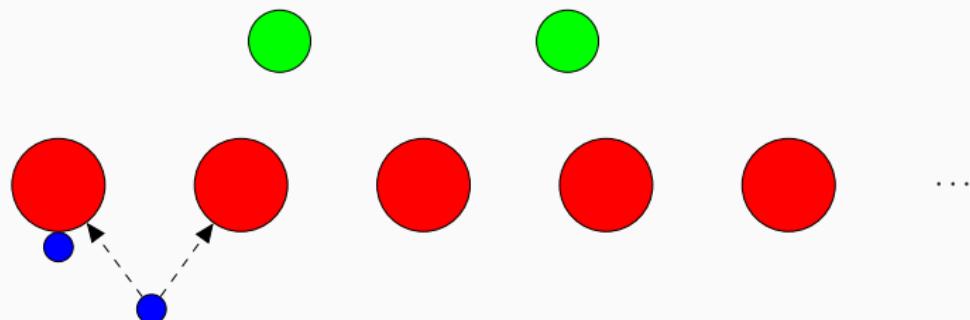
# Chinese Restaurant Process



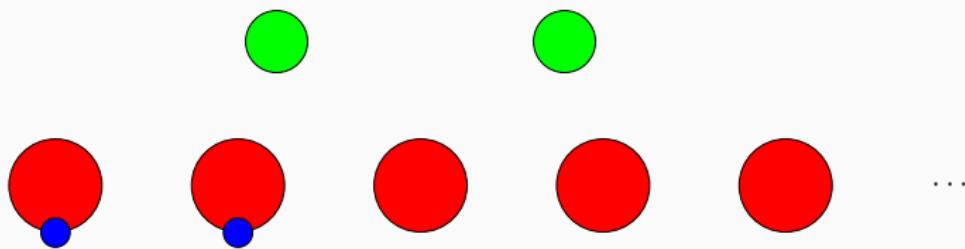
# Chinese Restaurant Process



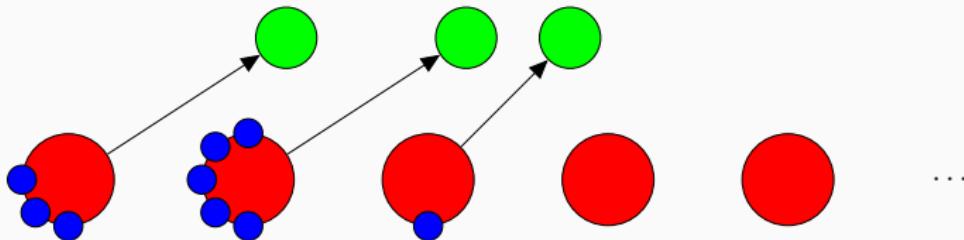
# Chinese Restaurant Process



# Chinese Restaurant Process

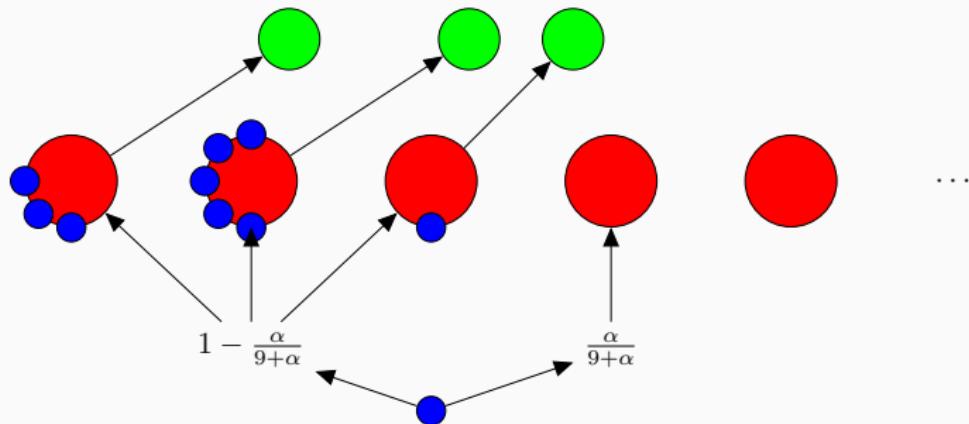


# Chinese Restaurant Process

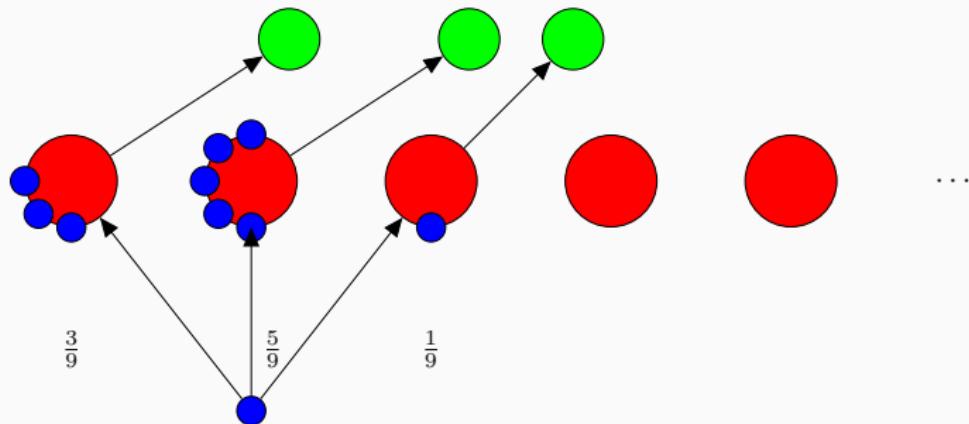


- Go to new table  $\frac{\alpha}{N-1+\alpha}$
- If not choose table as  $\frac{n_i}{N}$  where  $n_i$  number of diners at table  $i$

# Chinese Restaurant Process



# Chinese Restaurant Process



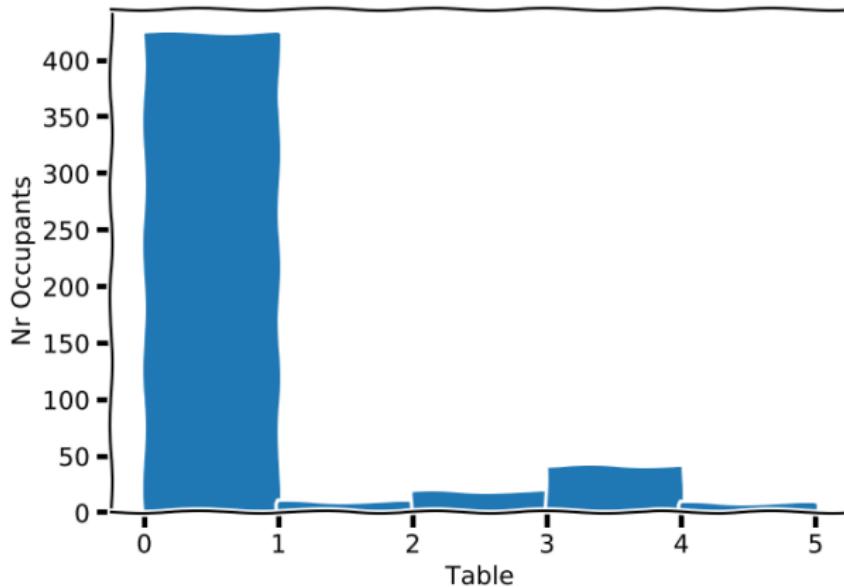
# Chinese Restaurant Process Code

---

```
def chrp(N, alpha):
    table_assignments = np.zeros(N)
    next_open_table = 0
    for i in range(0,N):
        r = np.random.random()
        if r < (alpha/(i+alpha)):
            table_assignments[i] = next_open_table
            next_open_table += 1
        else:
            index = int(np.round((i-1)*np.random.random()))
            table_assignments[i] = table_assignments[index]

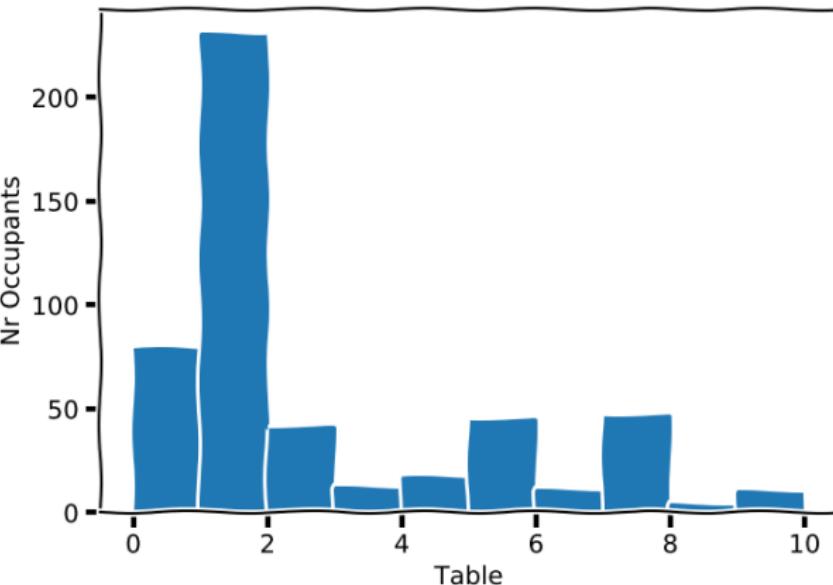
    return table_assignments
```

# Chinese Restaurant Process



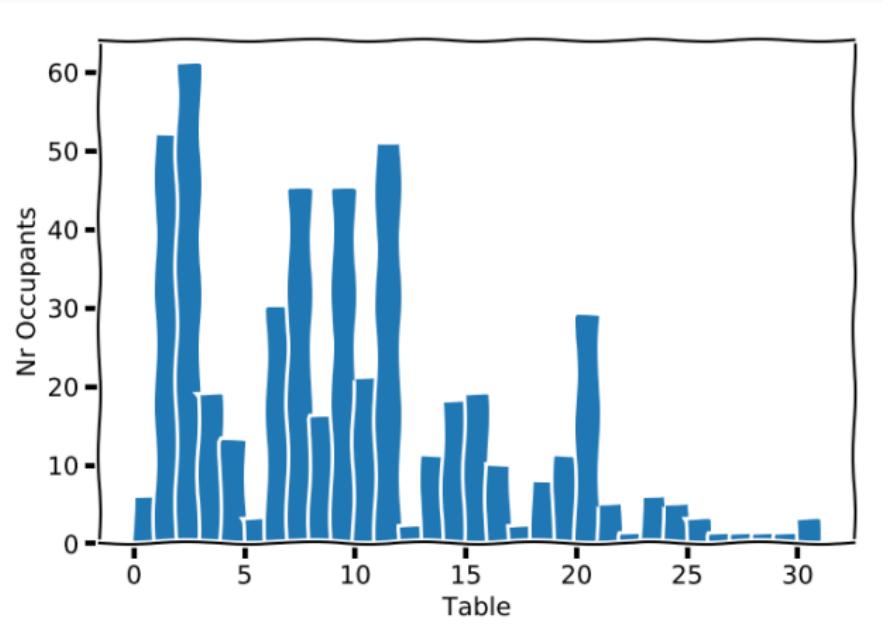
$$N = 500 \quad \alpha = 1.0$$

# Chinese Restaurant Process



$$N = 500 \quad \alpha = 2.0$$

# Chinese Restaurant Process



$$N = 500 \quad \alpha = 10.0$$

# Chinese Restaurant Process

---

N	$\alpha$	$\mu_K$	$\sigma_K$
500	3	14.3	8.81
500	5	21.6	15.64
500	10	39.1	27.29
500	20	64.3	63.8
500	100	180	69.69

# Infinite Mixture Model

---

1. Pick first data-point

# Infinite Mixture Model

---

1. Pick first data-point
2. Pick the first mixture

# Infinite Mixture Model

---

1. Pick first data-point
2. Pick the first mixture
3. Pick parameters associated with this mixture  $\mu_k, \sigma_k$

# Infinite Mixture Model

---

1. Pick first data-point
2. Pick the first mixture
3. Pick parameters associated with this mixture  $\mu_k, \sigma_k$
4. Next data-point

# Infinite Mixture Model

---

1. Pick first data-point
2. Pick the first mixture
3. Pick parameters associated with this mixture  $\mu_k, \sigma_k$
4. Next data-point
5. Associate with a new mixture or create new

# Infinite Mixture Model

---

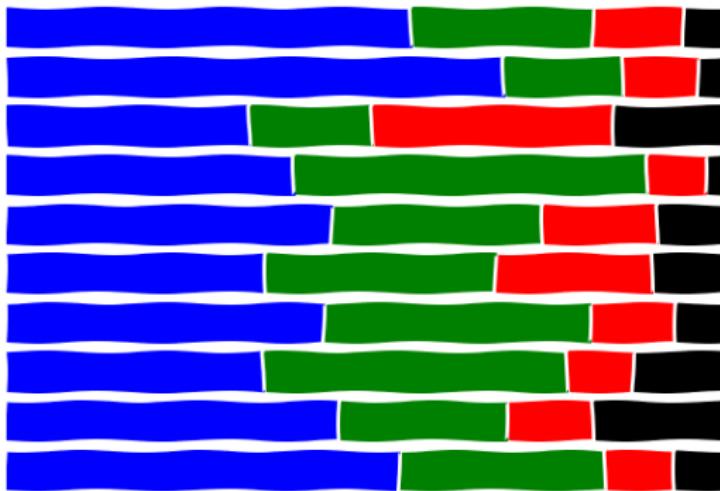
1. Pick first data-point
2. Pick the first mixture
3. Pick parameters associated with this mixture  $\mu_k, \sigma_k$
4. Next data-point
5. Associate with a new mixture or create new
6. If new, pick new parameters

# Infinite Mixture Model

---

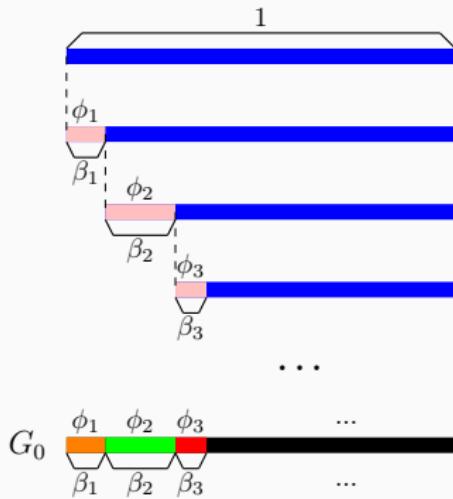
1. Pick first data-point
2. Pick the first mixture
3. Pick parameters associated with this mixture  $\mu_k, \sigma_k$
4. Next data-point
5. Associate with a new mixture or create new
6. If new, pick new parameters
7. Repeat

# Dirichlet Distribution



$\text{Dir}(7, 5, 3, 2)$

# Stick-Breaking



$$G \sim \text{DP}(\mathcal{H}, \alpha)$$

$$\hat{\beta}_k \sim \text{Beta}(1, \alpha)$$

$$\beta_k = \hat{\beta}_k \prod_{l=1}^{k-1} (1 - \hat{\beta}_l)$$

$$\Phi \sim \mathcal{H}$$

$$G_0 = \sum_{k=1}^{\infty} \beta_k \Phi_k$$

## This is all really different

---

- A probability measure is a measure on how much we believe in a specific setting of a variable

## This is all really different

---

- A probability measure is a measure on how much we believe in a specific setting of a variable
- We have derived a formulation that provides measure on any partition

## This is all really different

---

- A probability measure is a measure on how much we believe in a specific setting of a variable
- We have derived a formulation that provides measure on any partition
  - We can now search for the configuration that is most likely: ML or MAP etc.

## This is all really different

---

- A probability measure is a measure on how much we believe in a specific setting of a variable
- We have derived a formulation that provides measure on any partition
  - We can now search for the configuration that is most likely: ML or MAP etc.
  - We can try to derive the posterior over the partition

## This is all really different

---

- A probability measure is a measure on how much we believe in a specific setting of a variable
- We have derived a formulation that provides measure on any partition
  - We can now search for the configuration that is most likely: ML or MAP etc.
  - We can try to derive the posterior over the partition
- *We are so far only looking at how to formulate models*

## Summary

---

## Summary

---

- Dirichlet processes are priors over countably infinite sets

## Summary

---

- Dirichlet processes are priors over countably infinite sets
- Allows for models dealing with infinite partitions

## Summary

---

- Dirichlet processes are priors over countably infinite sets
- Allows for models dealing with infinite partitions
  - Infinite Gaussian Mixture Models

# Summary

---

- Dirichlet processes are priors over countably infinite sets
- Allows for models dealing with infinite partitions
  - Infinite Gaussian Mixture Models
- Take home message: generative models

# Summary

---

- Dirichlet processes are priors over countably infinite sets
- Allows for models dealing with infinite partitions
  - Infinite Gaussian Mixture Models
- Take home message: generative models
- Tomorrow: Latent Dirichlet Allocation

eof

## References

---

 Christopher M. Bishop.

*Pattern Recognition and Machine Learning (Information  
Science and Statistics).*

Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.