# ECE 884 Deep Learning

Lecture 5: Parametric Models

02/02/2021

# Review of last lecture

- Function form#1: nonparametric models

# Today’s lecture

- Function form#2: parametric linear models
- Function form#3: parametric nonlinear models (DL models)
  - Understand why deep learning models are powerful and importance of massive computation and large data sets.

# Today’s lecture

- Function form#2: parametric linear models

# Parametric Approach

Image

Array of **32x32x3** numbers
(3072 numbers total)

$f(\mathbf{x}, \mathbf{W})$

**10** numbers giving
class scores

W

parameters
or weights

Credit: Justin

# Parametric Approach: Linear Classifier

$$f(x,W) = Wx$$

Image

Array of **32x32x3** numbers
(3072 numbers total)

$f(\mathbf{x},\mathbf{W})$

**10** numbers giving
class scores

W
parameters
or weights

Credit: Justin

# Parametric Approach: Linear Classifier

$$\underbrace{\boxed{f(x,W)}}_{(10,)} = \underbrace{\boxed{W}}_{(10,\,3072)}\overbrace{\boxed{x}}^{(3072,)}$$

Image

Array of **32x32x3** numbers
(3072 numbers total)

→ f(**x**,**W**) →

**10** numbers giving
class scores

↑

W

parameters
or weights

Credit: Justin

# Parametric Approach: Linear Classifier

(3072,)

$$\boxed{f(x,W)} = \boxed{W}\boxed{x} + \boxed{b}$$

(10,)

(10,) (10, 3072)

Image

Array of **32x32x3** numbers
(3072 numbers total)

$\longrightarrow$ f(**x**,**W**) $\longrightarrow$ **10** numbers giving class scores

$\uparrow$

W

parameters
or weights

Credit: Justin

# Example for 2x2 image, 3 classes (cat/dog/ship)

Stretch pixels into column

$f(x,W) = Wx + b$

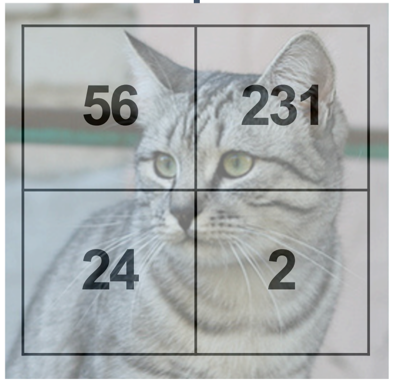

Input image
(2, 2)

(4,)

# Example for 2x2 image, 3 classes (cat/dog/ship)

Stretch pixels into column

$f(x,W) = Wx + b$




Input image
(2, 2)

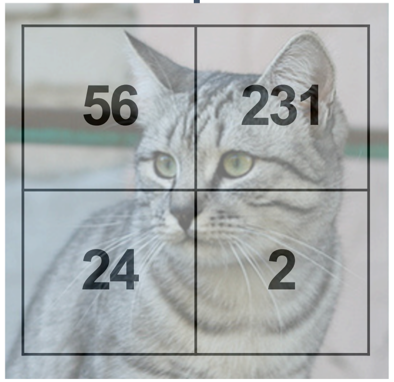
$$\underbrace{\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix}}_{W\ (3,\ 4)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix}}_{(4,)} + \underbrace{\begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix}}_{b\ (3,)} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$

Credit: Justin

# Linear Classifier: Algebraic Viewpoint

Stretch pixels into column

$$f(x,W) = Wx + b$$




Input image
(2, 2)

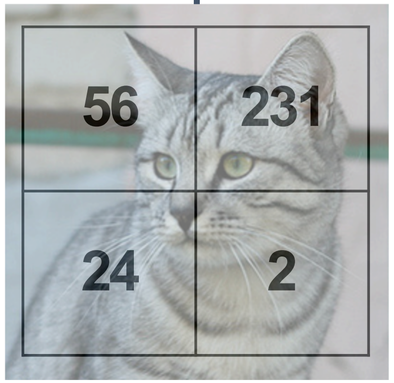
$$\underbrace{\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix}}_{W\ (3,\ 4)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix}}_{(4,)} + \underbrace{\begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix}}_{\substack{b \\ (3,)}} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$

Credit: Justin

# Linear Classifier: Bias Trick

Add extra one to data vector; bias is absorbed into last column of weight matrix

Stretch pixels into column




Input image
(2, 2)

$$\underbrace{\left[\begin{array}{cccc|c} 0.2 & -0.5 & 0.1 & 2.0 & 1.1 \\ 1.5 & 1.3 & 2.1 & 0.0 & 3.2 \\ 0 & 0.25 & 0.2 & -0.3 & -1.2 \end{array}\right]}_{W\ (3,\ 5)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \\ 1 \end{bmatrix}}_{(5,)} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$

Credit: Justin

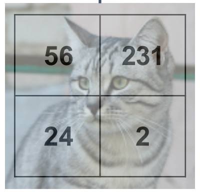
# Interpreting a Linear Classifier

Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!

## Algebraic Viewpoint

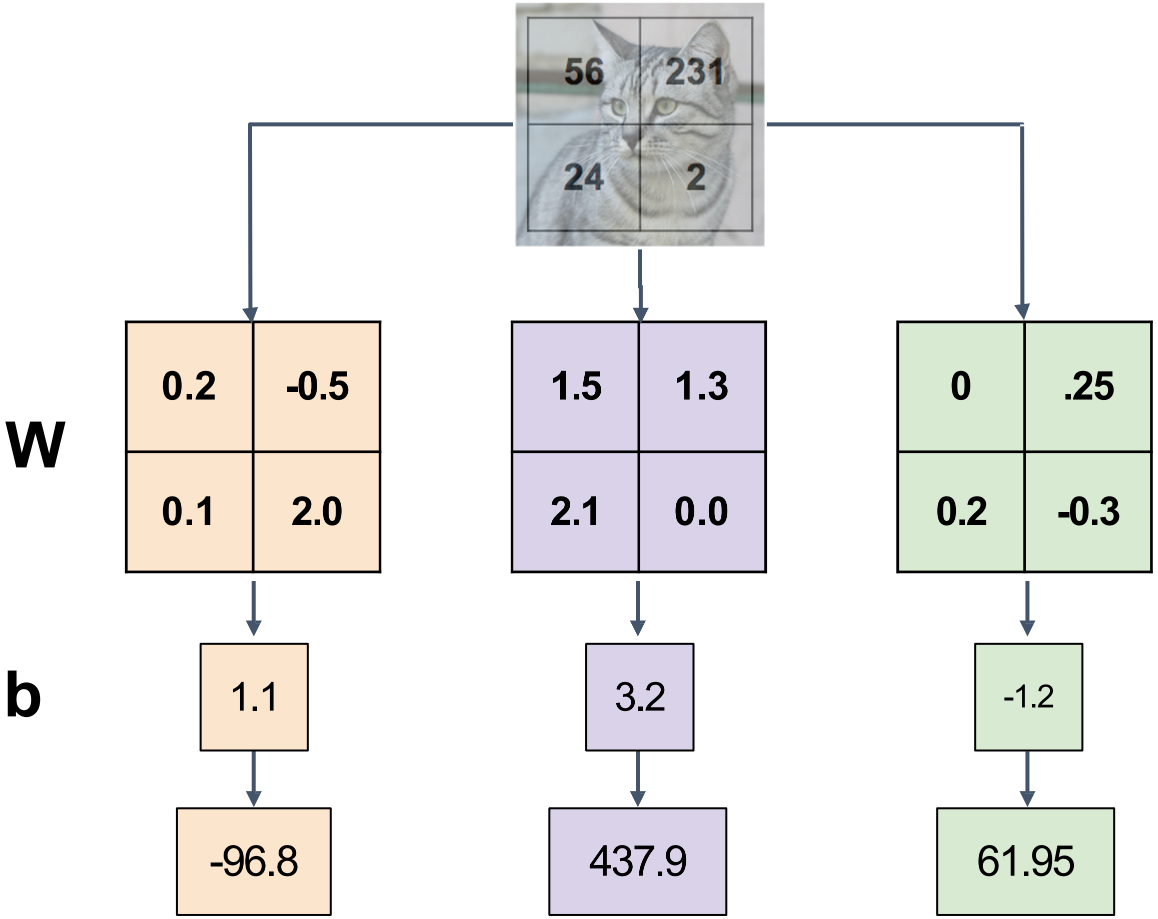
$$f(x,W) = Wx + b$$

Stretch pixels into column




Input image
(2, 2)

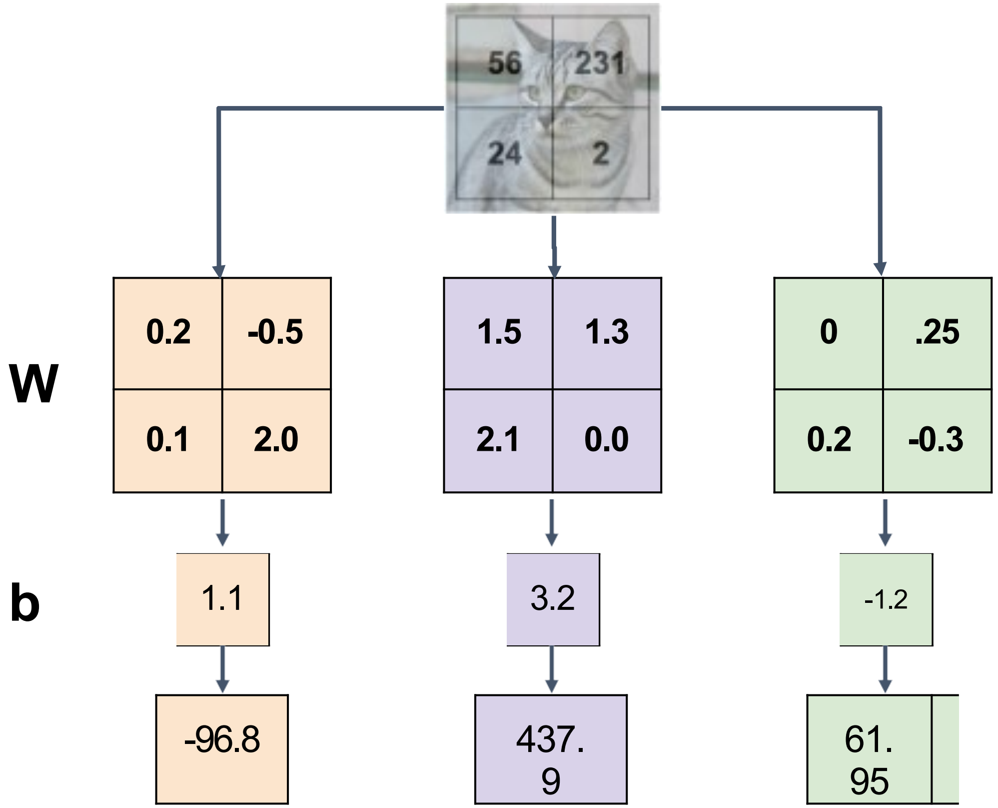
$$\underbrace{\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix}}_{W\ (3,\ 4)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix}}_{(4,)} + \underbrace{\begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix}}_{b\ (3,)} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$




Credit: Justin

# Interpreting an Linear Classifier: Visual Viewpoint

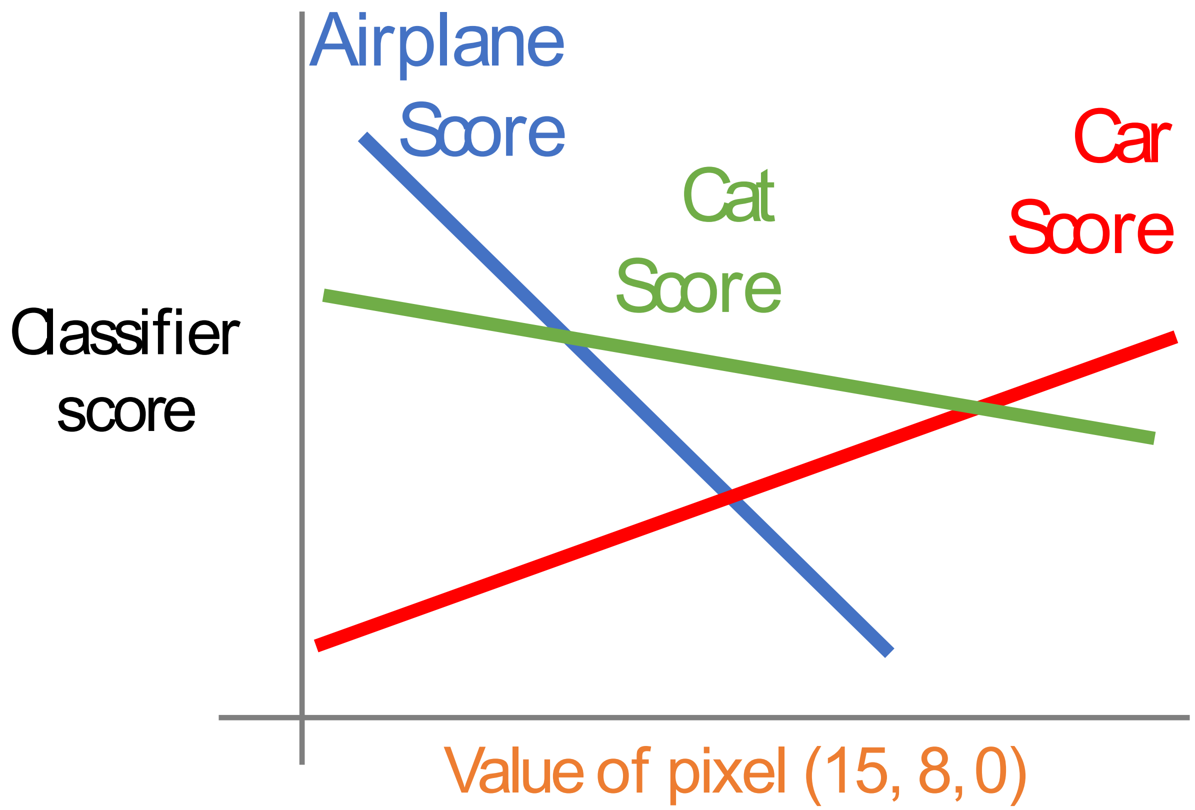
Linear classifier has one "template" per category





Credit: Justin

# Interpreting a Linear Classifier: Geometric Viewpoint

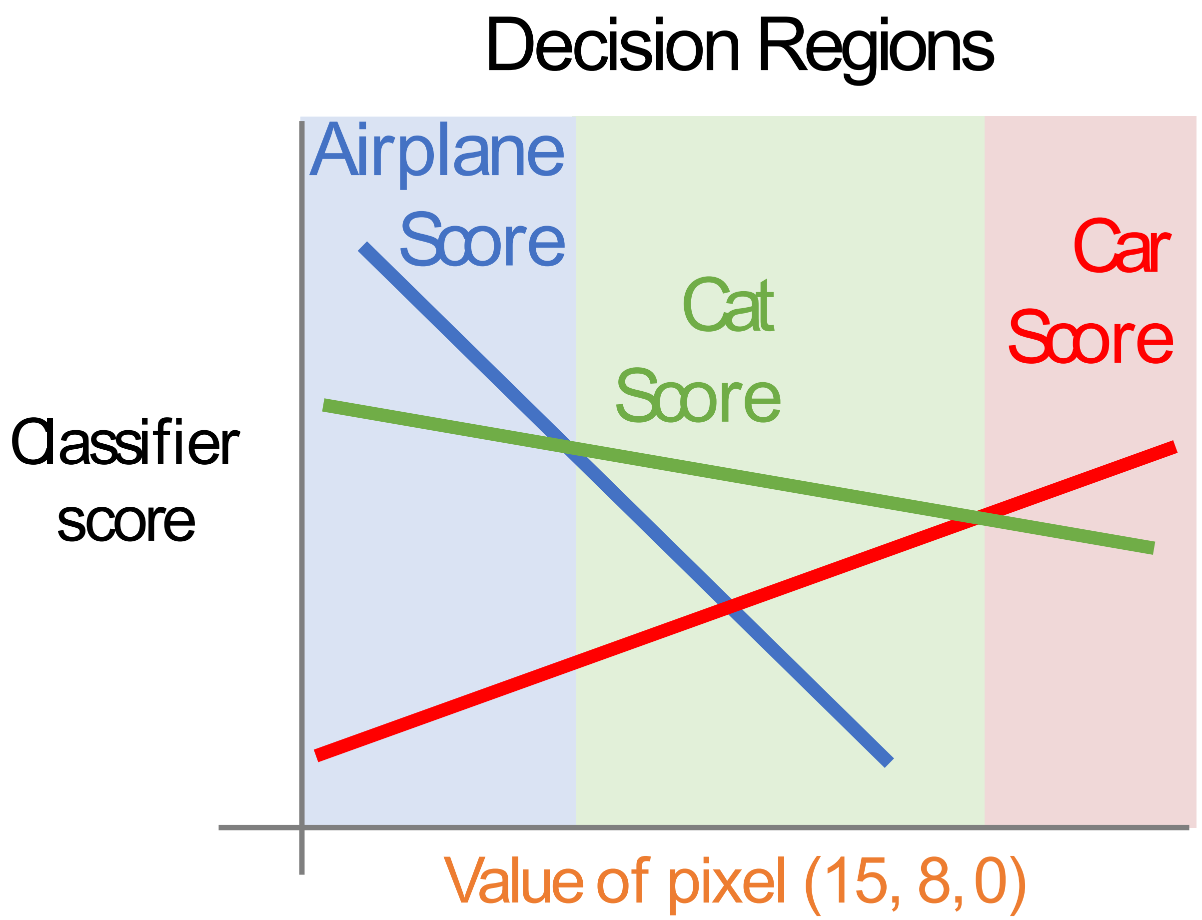



$$f(x,W) = Wx + b$$

Array of **32x32x3** numbers
(3072 numbers total)

Credit: Justin

# Interpreting a Linear Classifier: Geometric Viewpoint




$$f(x,W) = Wx + b$$

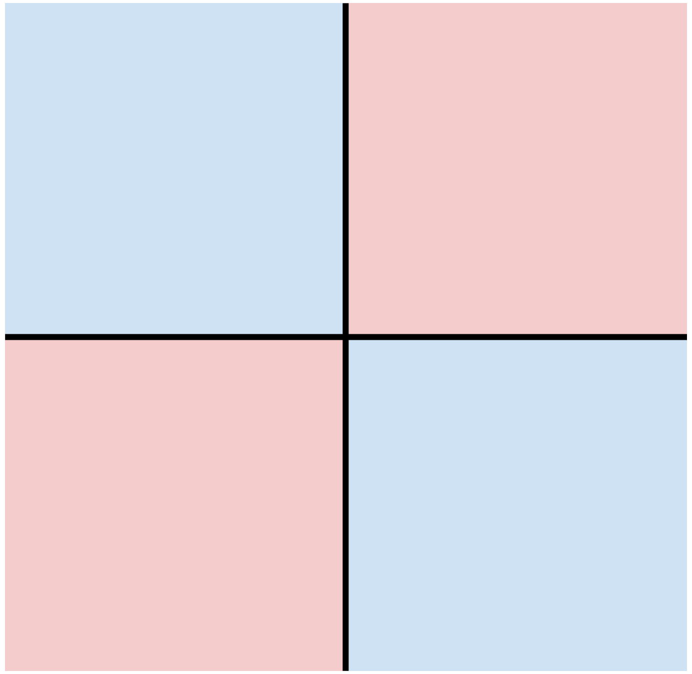
Array of **32x32x3** numbers
(3072 numbers total)

Credit: Justin

# Hard Cases for a Linear Classifier

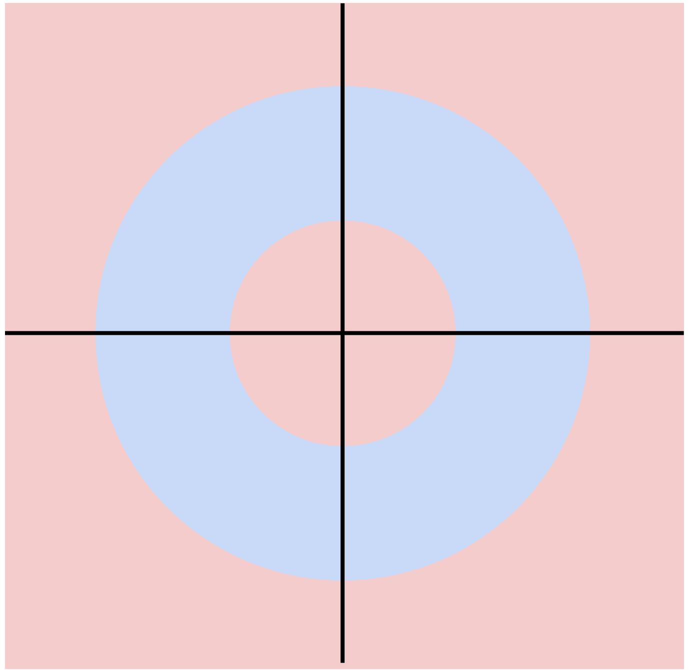
**Class 1**:
First and third quadrants

**Class 2**:
Second and fourth quadrants



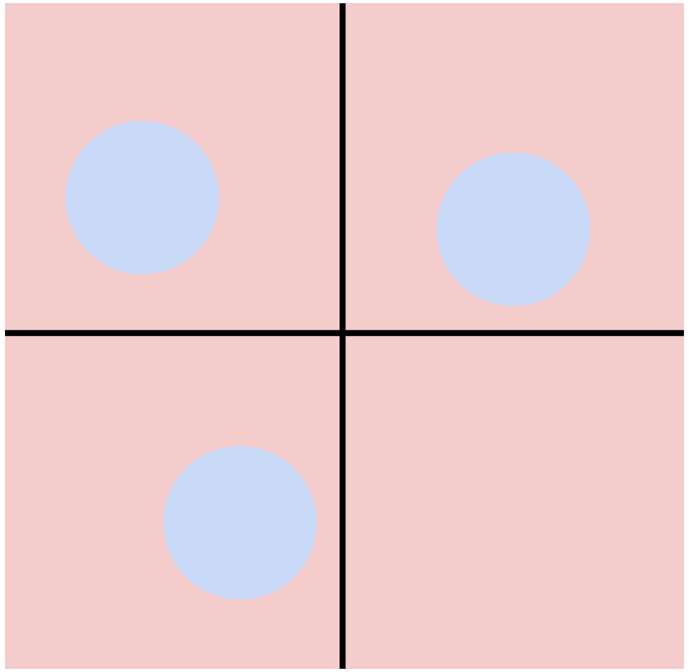
**Class 1**:
1 <= L2 norm <= 2

**Class 2**:
Everything else



**Class 1**:
Three modes

**Class 2**:
Everything else



Credit: Justin

# Non-parametric model

+ As the number of training samples goes to infinity, nearest neighbor can represent any(*) function!

- You need to carry all the data: curse of dimensionality!

# Linear parametric model

+ We have a function now, and we do not need to carry all the data!

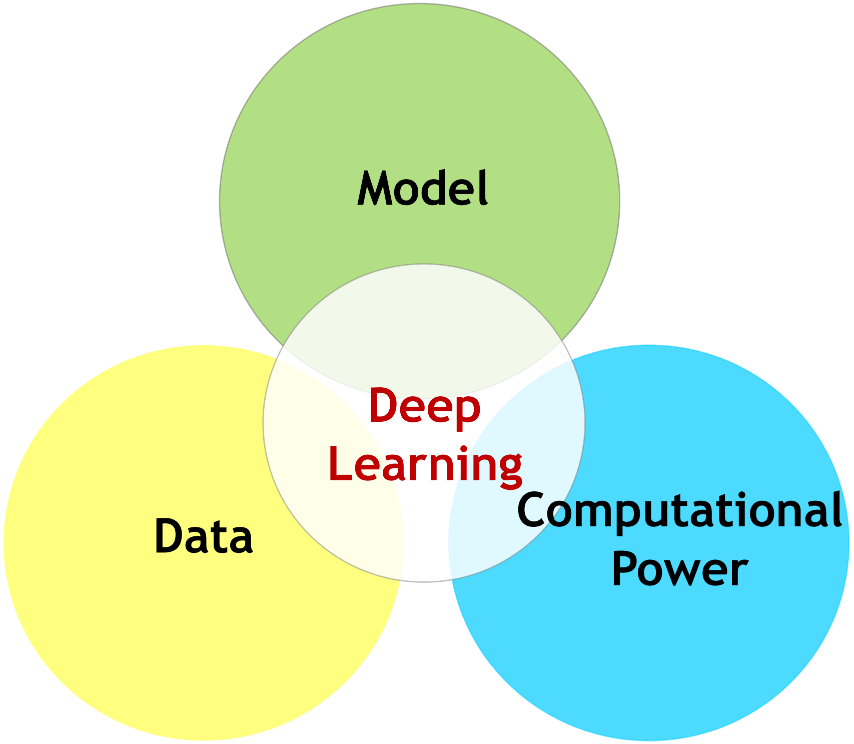
- Its capability is limited!

Can we combine the advantages of both worlds and get rid of their drawbacks?

# Today’s lecture

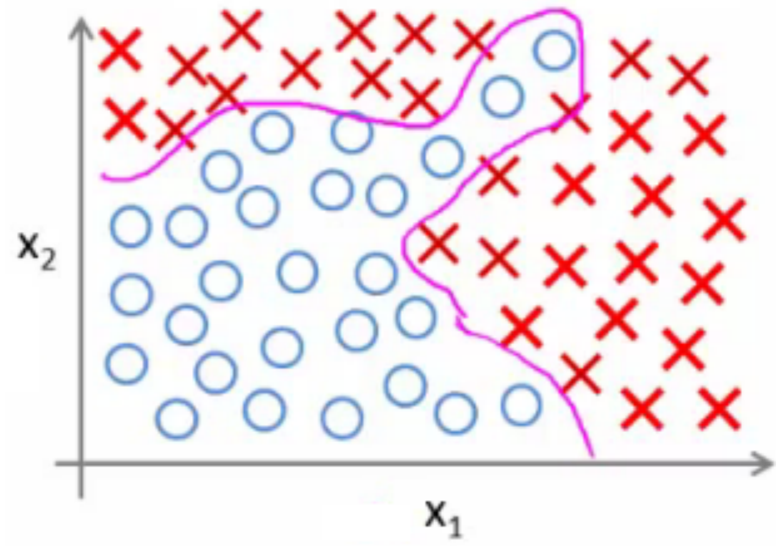
- Function form#3: parametric nonlinear models (DL models)
    - Understand why deep learning models are powerful and importance of massive computation and large data sets.

# Model + Data + Computational Power

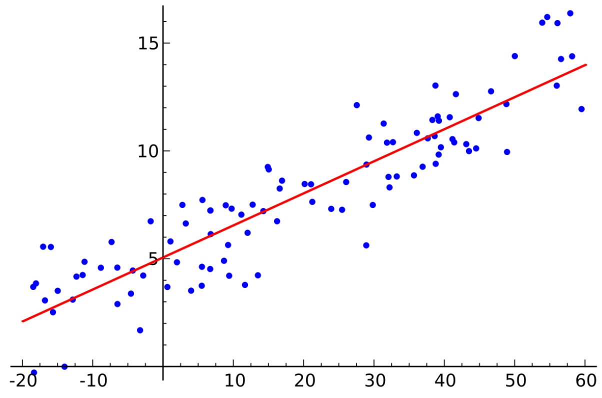


# Model

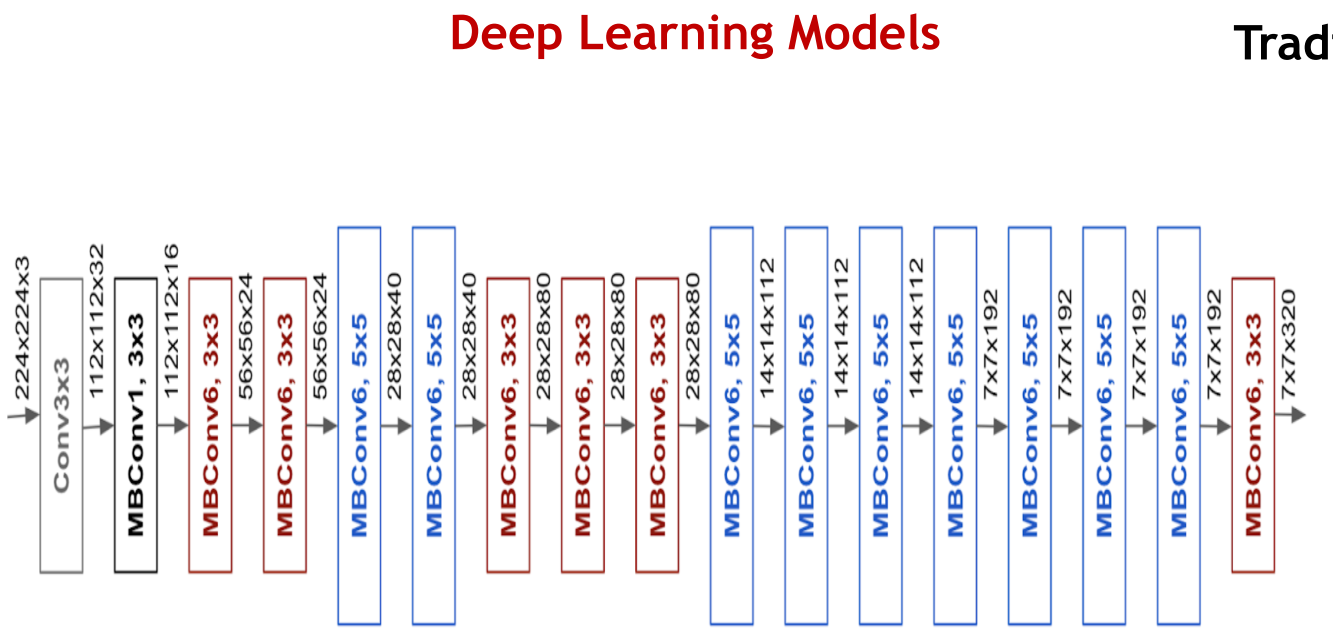
**Deep Learning Models**



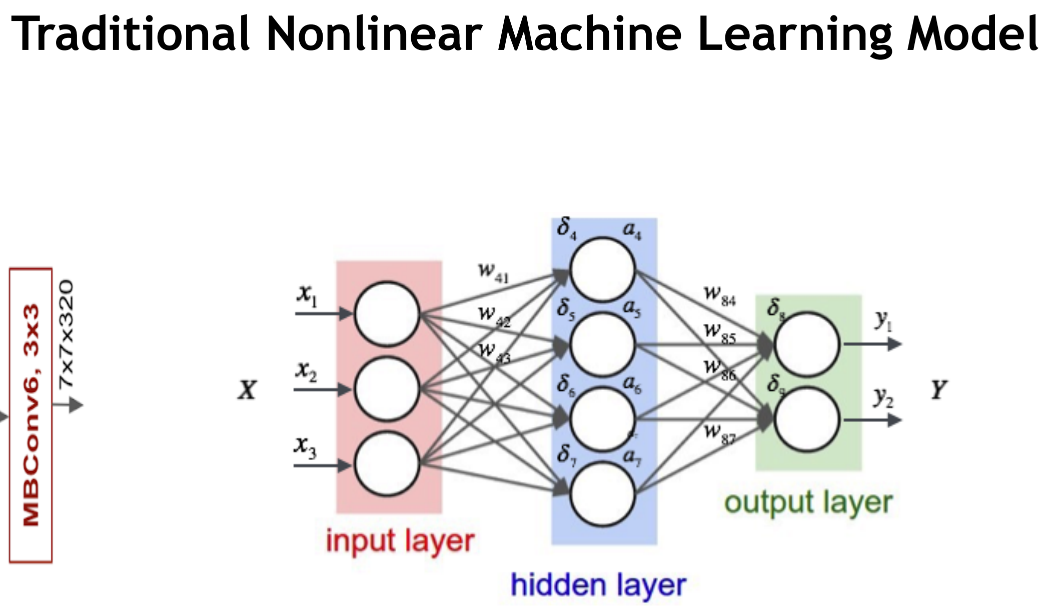

**Traditional Linear Machine Learning Model**



# Model



Deep Learning Models
224x224x3
Conv3x3
112x112x32
MBConv1, 3x3
112x112x16
MBConv6, 3x3
56x56x24
MBConv6, 3x3
56x56x24
MBConv6, 5x5
28x28x40
MBConv6, 5x5
28x28x40
MBConv6, 3x3
28x28x80
MBConv6, 3x3
28x28x80
MBConv6, 3x3
28x28x80
MBConv6, 5x5
14x14x112
MBConv6, 5x5
14x14x112
MBConv6, 5x5
14x14x112
MBConv6, 5x5
7x7x192
MBConv6, 5x5
7x7x192
MBConv6, 5x5
7x7x192
MBConv6, 5x5
7x7x192
MBConv6, 3x3
7x7x320

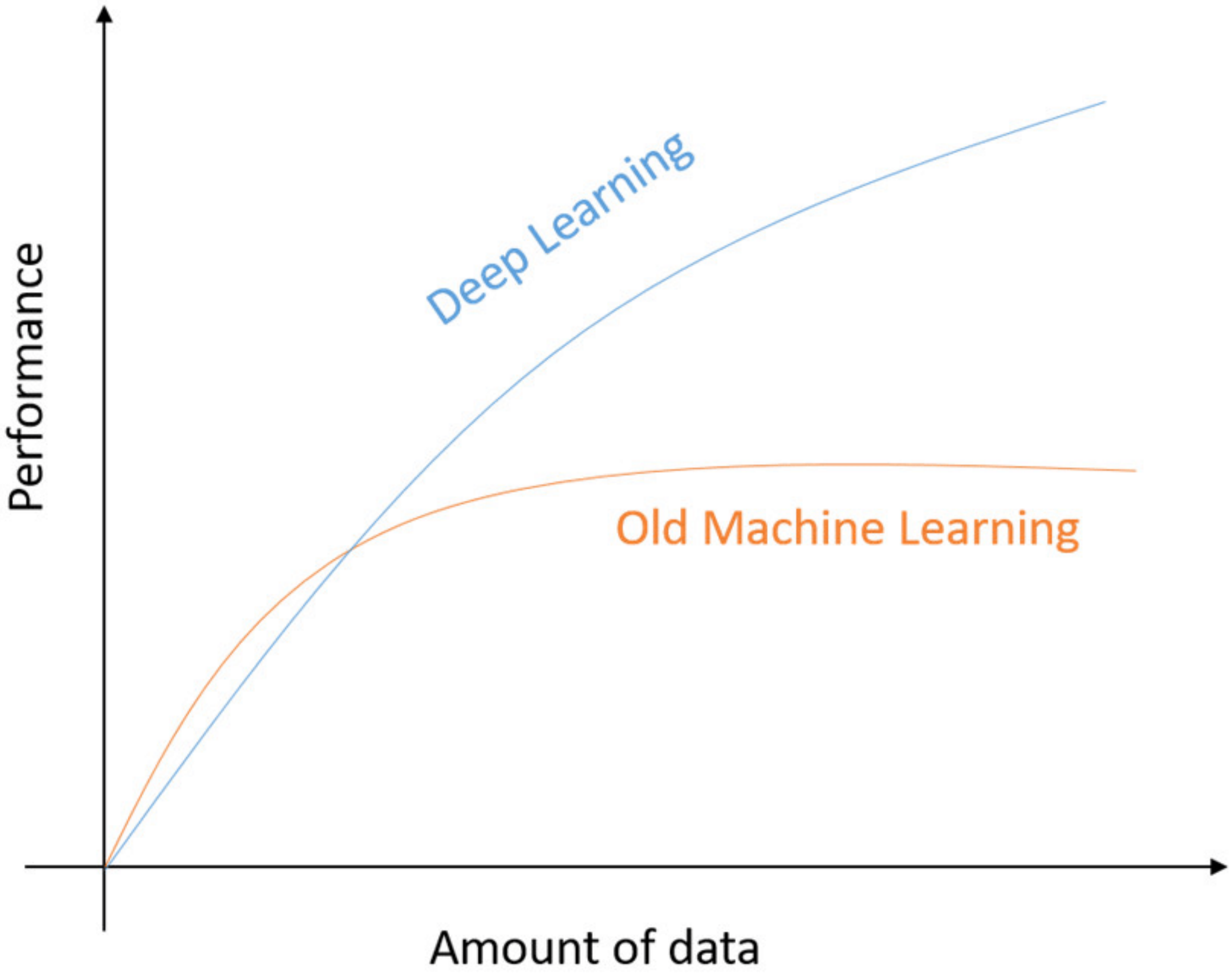
Traditional Nonlinear Machine Learning Model
X
Y
input layer
hidden layer
output layer

# Model

**Deep Learning Models**

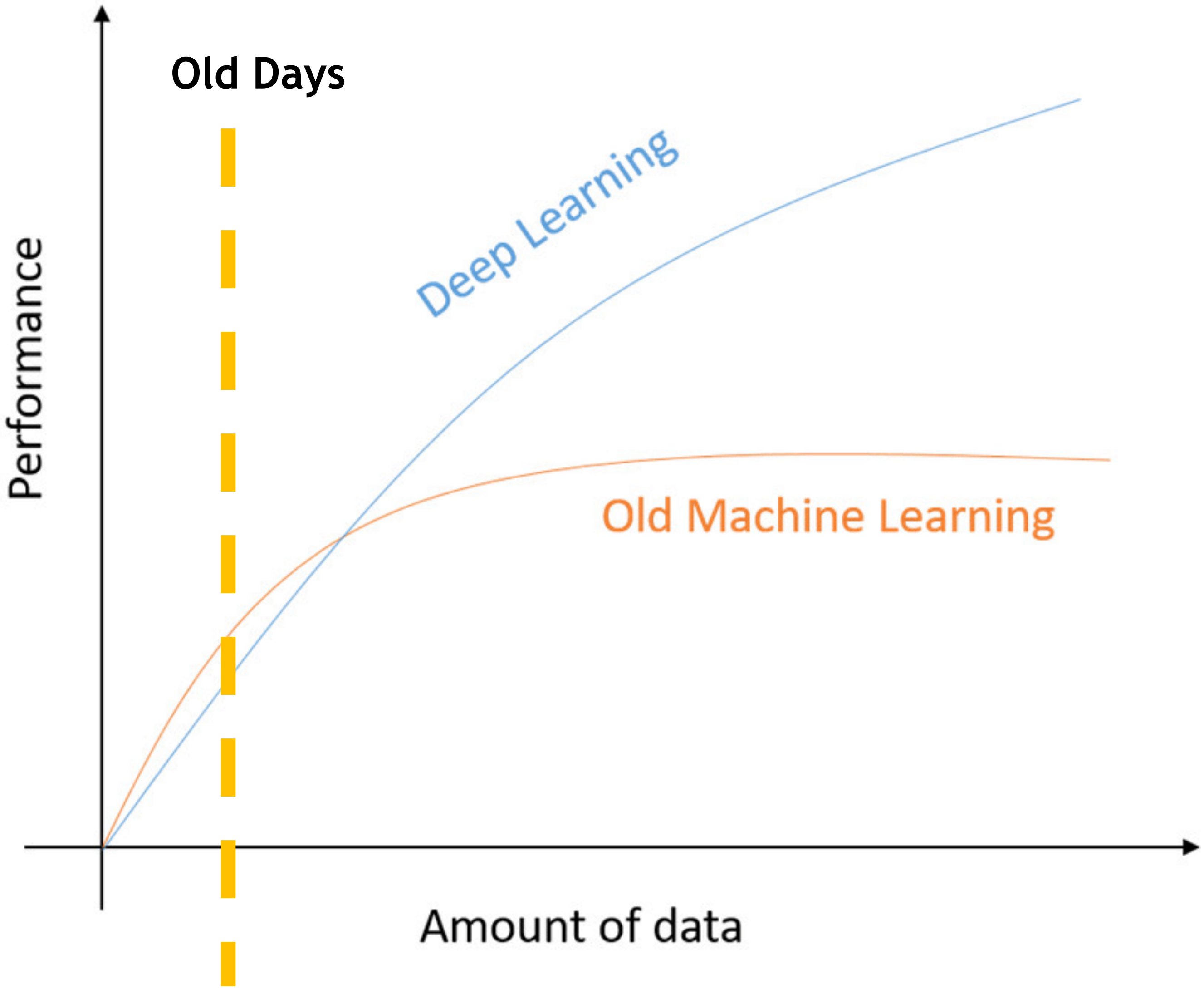
**Traditional Machine Learning Model**

# Data

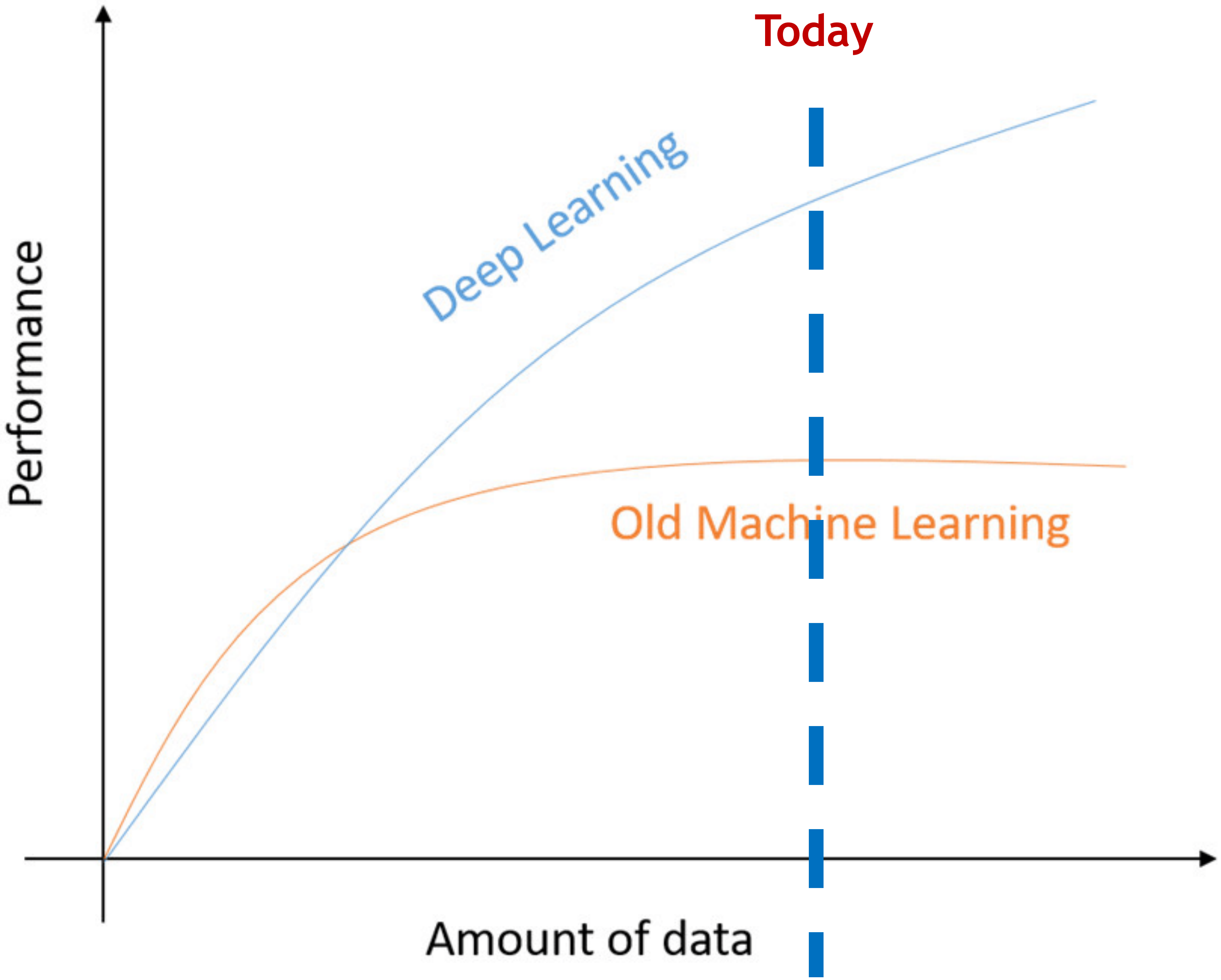


Performance
Deep Learning
Old Machine Learning
Amount of data

# Data


Old Days
Deep Learning
Old Machine Learning
Performance
Amount of data

# Data



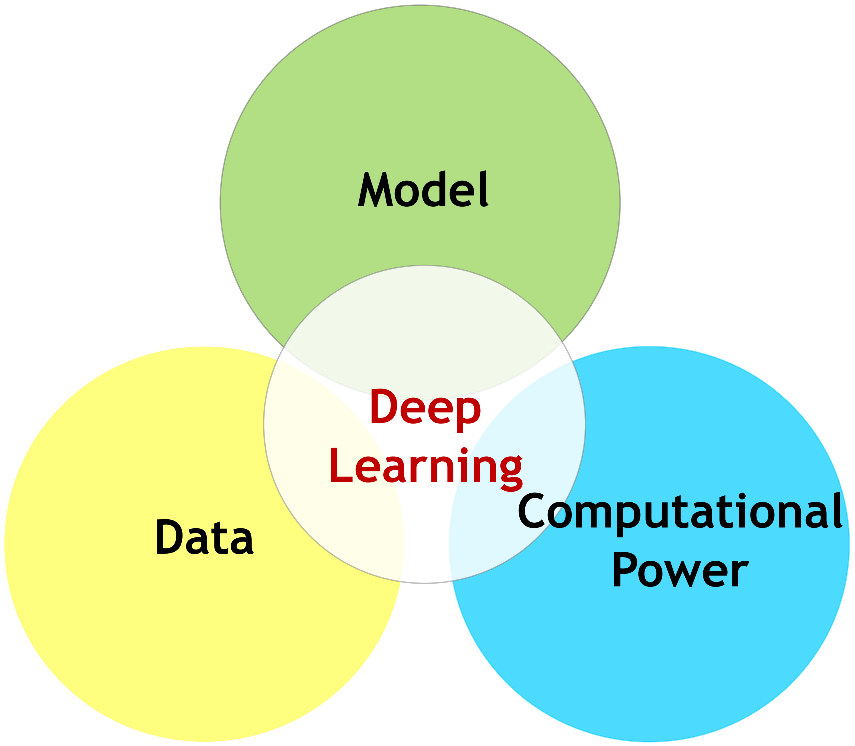
Today
Deep Learning
Performance
Old Machine Learning
Amount of data

# Computational Power

**Old Days**

**Today**

# Model + Data + Computational Power -> Success of Deep Learning



Any Question ?