

Python Fundamentals - Part 2

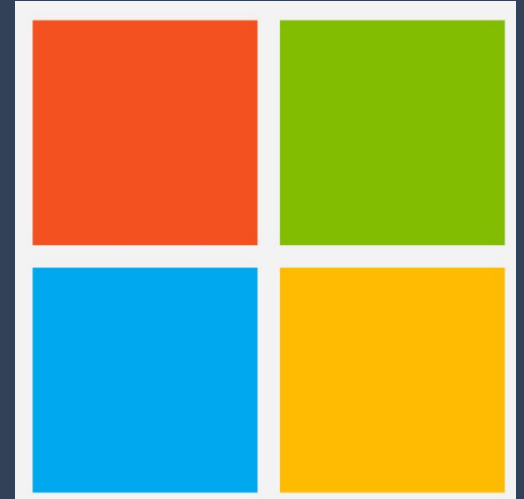
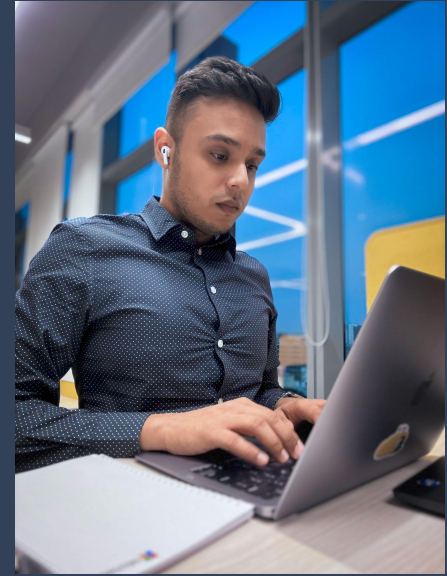


Instructor: Anshaj Khare

[Anshaj Khare | LinkedIn](#)

Introduction – Anshaj Khare

- ✓ I have been working on data science since 2017
- ✓ I've worked for 4 startups before joining Microsoft
- ✓ I have been working at Microsoft for about 2.5 years.
- ✓ In my career, I've worked as a backend engineer, a devops engineer, a data engineer and as a Machine learning engineer

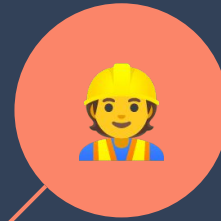


**Welcome to today's class, before we begin,
pop into the chat:**

Your Name



Role



Location

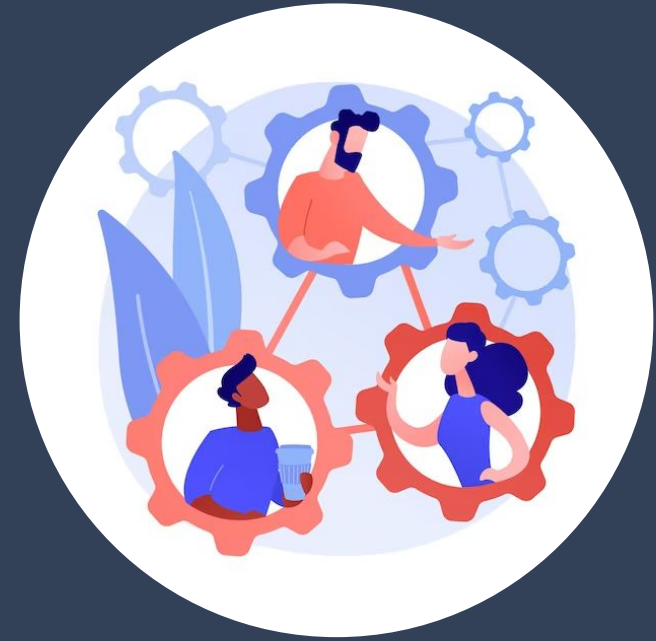


Company



Optimize Your Experience

- ✓ Interact with your instructors via Sunday live class.
- ✓ Don't be shy to speak up and get clarifications !



- ✓ Solve your assignments, MCQs and other assessments and get feedback (Thursday review session)
- ✓ Use your resources! It's your experience – what you put in is what you'll get out.

Today's Agenda



Break

Break

Break

1



Machine
Learning

2



Scikit
Learn

3



Deep
Learning

4



Tensors
&
PyTorch

How is Today's Content Relevant to My Job Role?

- **PMs:** Gain foundational knowledge of AI and ML concepts that will enable you to identify and prioritize AI features in product development.
- **TPMs:** Build an understanding of ML and DL frameworks like Scikit Learn and PyTorch, essential for managing AI-driven projects effectively.
- **SDEs:** Develop the ability to implement and optimize ML and DL models using Scikit Learn and PyTorch, crucial for integrating AI into software applications.
- **Engineering Managers:** Learn how to guide your engineering teams in applying ML and DL technologies, ensuring your projects leverage AI efficiently.
- **DevOps Engineers:** Understand the infrastructure and deployment considerations for ML and DL models, preparing you to support AI solutions in production environments.

Frequently Asked Question - 1

I don't have to code in my job role, why am I learning Python Libraries like Scikit Learn, PyTorch here? How will it help me?

Don't worry, you don't have to learn Python to the extent of a Python Developer. You only need to be able to comprehend the code that would be used in the upcoming modules in the course.

In GenAI, Python is just a tool! With this Crash Course, you would understand just enough on how to operate the tool and that's all you need!

Frequently Asked Question - 2

I have never coded in Python before. Will I become an expert in Python after completing this module?

This is a Python Crash Course - tailored specifically to equip you with enough knowledge to help you navigate through the Applied GenAI Course smoothly.

Generative AI technologies are implemented through Python, and you will encounter multiple coding demos in Python throughout the course. This Python Crash Course includes just enough details so that the coding demos in the further modules are comprehensible.

If you have never coded in Python before, you may need to refer to external resources too. If you need help with the resources, please contact studentsupport@interviewkickstart.com

If you are already familiar with Python programming, this module may seem like a basic refresher for you - and that's exactly what it's meant to be!

Today's Agenda



Break

Break

Break

1



Machine
Learning

2



Scikit
Learn

3



Deep
Learning

4



Tensors
&
PyTorch

Machine Learning

Welcome to the World of Machine Learning!

- Machine Learning (ML) is the backbone of modern AI systems, allowing computers to learn from data and improve their performance without being explicitly programmed.

Why It Matters for GenAI?

- In the context of Generative AI, ML is crucial for creating models that can generate new content, understand patterns, and make decisions autonomously.

Types of Machine Learning

1. Supervised Learning

- The model learns from labeled data, where the correct output is provided during training.
- Regression, Classification.

2. Unsupervised Learning

- The model learns from unlabeled data and tries to find hidden patterns or groupings.
- Clustering: Grouping similar data points together (e.g., customer segmentation).

3. Semi-Supervised Learning

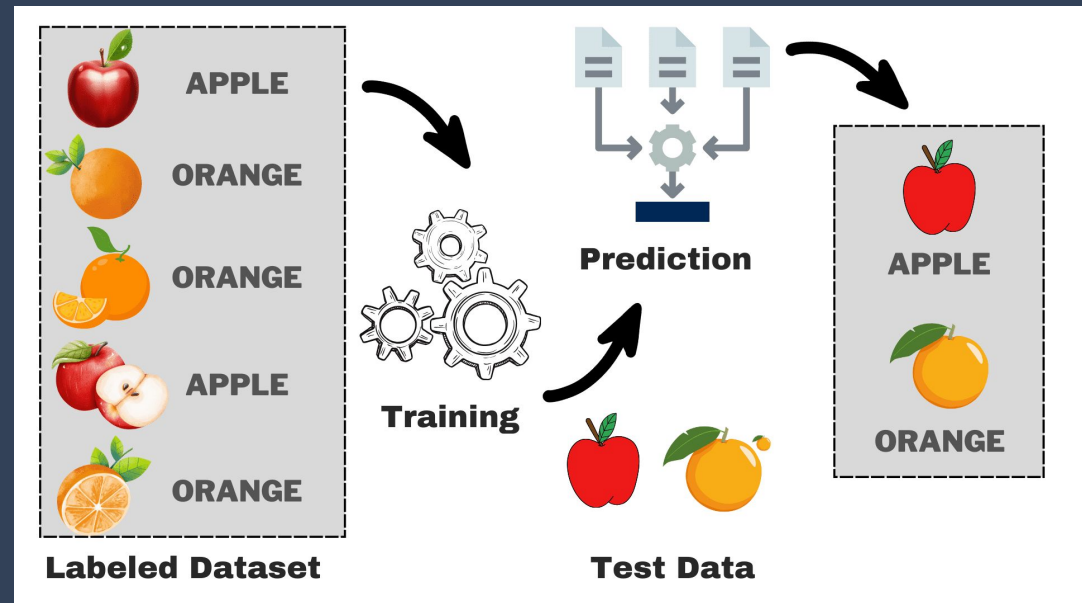
- The model is trained on a mix of labeled and unlabeled data. Often, a small amount of labeled data and a large amount of unlabeled data are used.

4. Reinforcement Learning

- The model learns by interacting with an environment, receiving rewards or penalties based on its actions, and optimizing its strategy to maximize cumulative rewards.

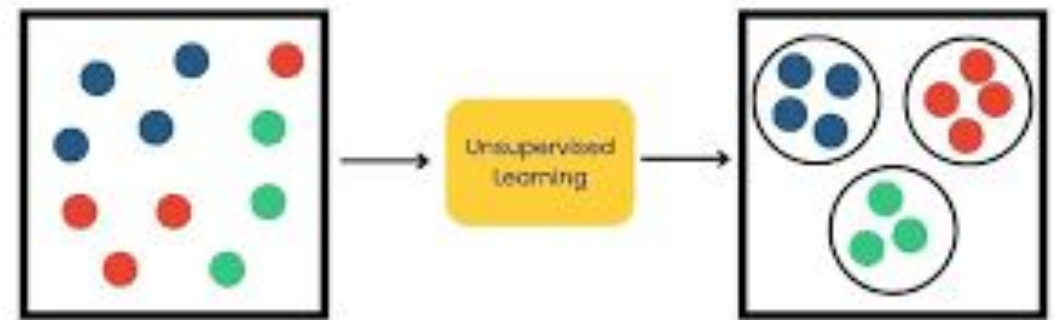
Supervised Learning

- Supervised Learning involves training a model on a labeled dataset, where the correct output is known. The model learns to map inputs to the correct outputs.
- Common examples include spam detection, where emails are labeled as 'spam' or 'not spam,' and image classification, where images are labeled by the objects they contain."



Unsupervised Learning

- Unsupervised Learning involves training a model on data without labeled outputs. The model tries to learn the underlying structure of the data.
- Examples include customer segmentation, where customers are grouped based on similar behaviors, and anomaly detection, where outliers are identified.



Regression

Let's start with an example!

House Price Prediction



[Source](#)

What is Regression?

- Regression is a type of supervised learning where the model predicts continuous outcomes based on input data.
- Examples: Predicting house prices based on features like size and location, or forecasting sales figures.

A quick poll: Which factor do you think most influences house prices? (A) Location (B) Size (C) Age of the house.

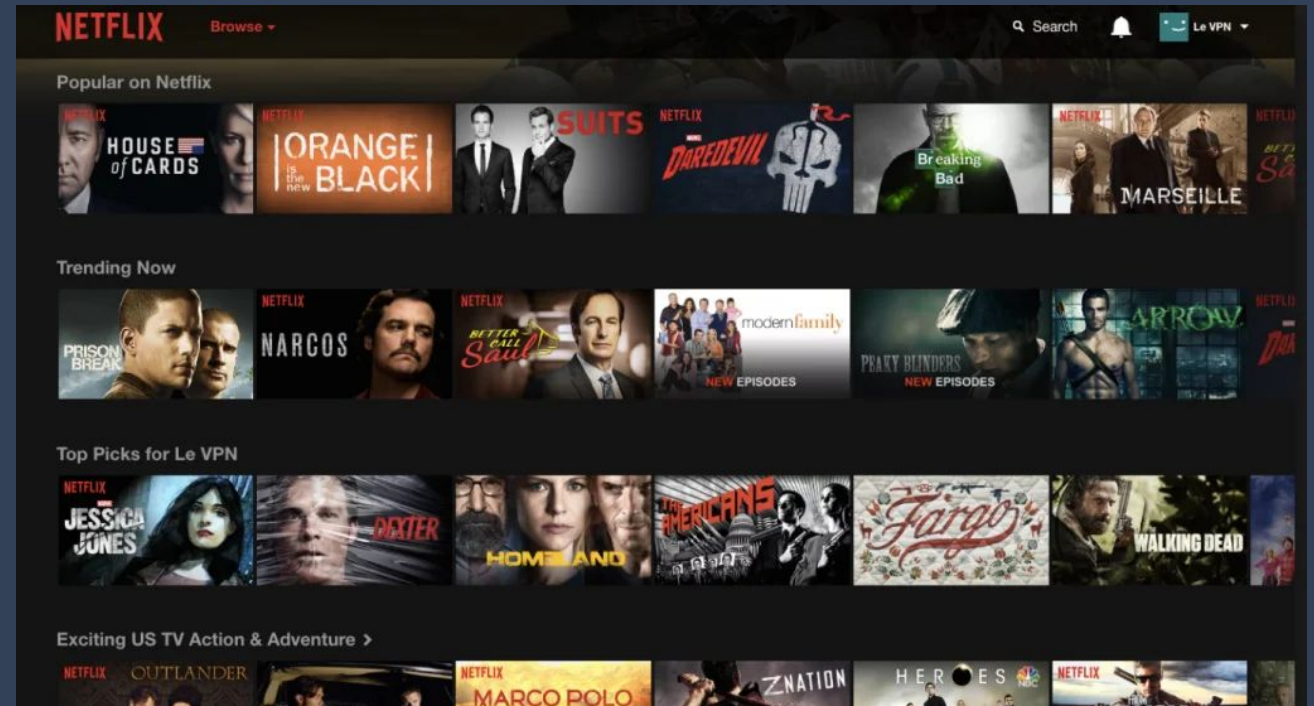
Classification

Let's start with some examples!

Email inbox spam filter



Streaming services: 'your recommendation'



What is Classification?

Classification:

Problem setting where the goal is to predict discrete labels (categories) for given inputs

Regression vs Classification:

Classification predicts *discrete labels*

regression predicts *continuous numerical values*

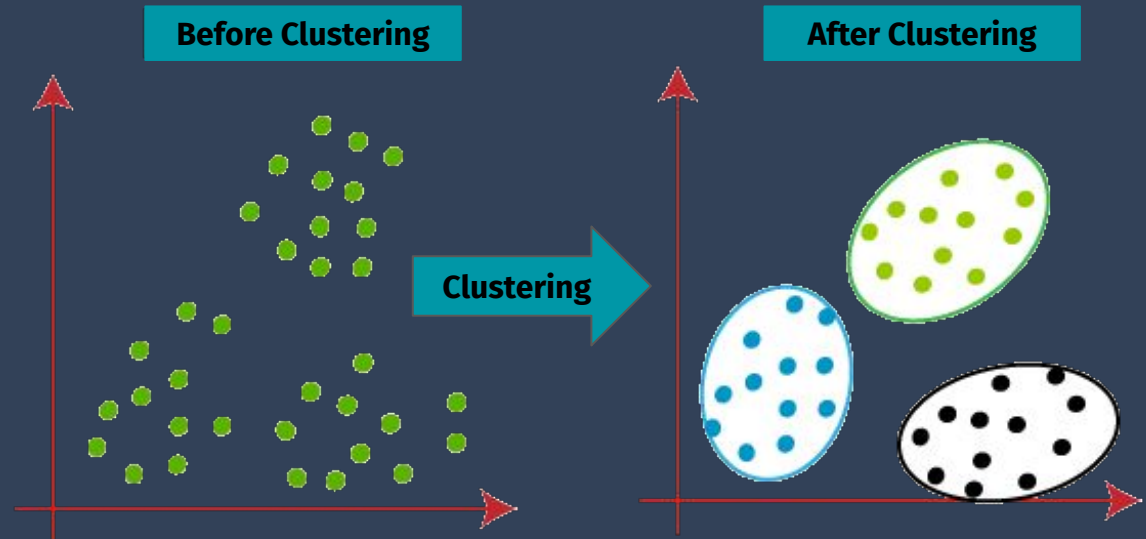
Example

- **Classification:** Email spam (spam or not spam)
- **Regression:** House price prediction (numerical value)

Clustering

Clustering is an unsupervised learning technique where the model groups similar data points together.

Examples: Market segmentation, where customers are grouped based on purchasing behavior, or social network analysis, where users are grouped based on interaction patterns.





Quiz

Which of the following is an example of a supervised learning task?

- A** Clustering customers based on purchasing behavior.
- B** Predicting house prices based on features like size and location.
- C** Grouping users in a social network based on interaction patterns.
- D** Analyzing trends in unstructured data.

Quiz - Solution

Which of the following is an example of a supervised learning task?

- A** Clustering customers based on purchasing behavior.
- B** Predicting house prices based on features like size and location.
- C** Grouping users in a social network based on interaction patterns.
- D** Analyzing trends in unstructured data.

Today's Agenda



Scikit Learn

Introduction

- Scikit-learn is a powerful Python library widely used for machine learning and data mining.
- Scikit-learn offers simple and efficient tools for data analysis and modeling, making it an essential tool in data science and machine learning.

Key python library: scikit-learn.org



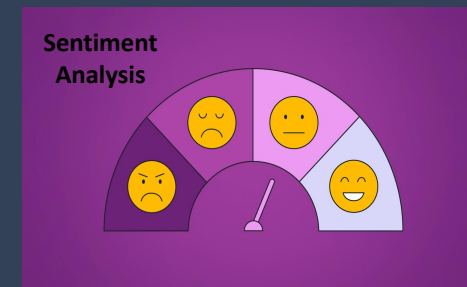
Why Scikit-Learn?

- **User-Friendly API:** Simple and consistent interface for various ML algorithms.
- **Comprehensive Documentation:** Extensive resources and examples for easy learning.
- **Integration with Other Libraries:** Seamlessly works with NumPy, SciPy, and Pandas.
- **Wide Range of Algorithms:** Includes classification, regression, clustering, etc.
- **Versatile Tools:** Supports data preprocessing, model selection, and evaluation.

How Does This Help Me?

Scikit-Learn

- **Simplifies ML Algorithms:** Easy-to-use API and comprehensive resources for quick understanding and application.
- **Seamless Integration:** Works well with NumPy, SciPy, and Pandas, essential for advanced GenAI projects.



```
#hide_output
# We increase `max_iter` to guarantee convergence
from sklearn.linear_model import LogisticRegression

lr_clf = LogisticRegression(max_iter=300)
lr_clf.fit(X_train, y_train)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:186:
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

Increase the number of iterations (max_iter) or scale the data as shown in <https://scikit-learn.org/stable/modules/preprocessing.html>. Please also refer to the documentation for alternative solver options: https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression.

```
LogisticRegression(max_iter=300)
```

Training and Testing in Machine Learning

- In machine learning, the dataset is typically split into two parts: training and testing. The model is trained on the training set and then tested on the testing set to evaluate its performance."
- **Why It's Important:** This process ensures that the model generalizes well to new, unseen data, which is crucial for reliable predictions.

Let's look at a simple example of splitting data and training a model in Scikit-Learn.



```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Demo Time

Summary

01

Overview and role of ML in our Course

02

Regression, Classification and Clustering in ML.

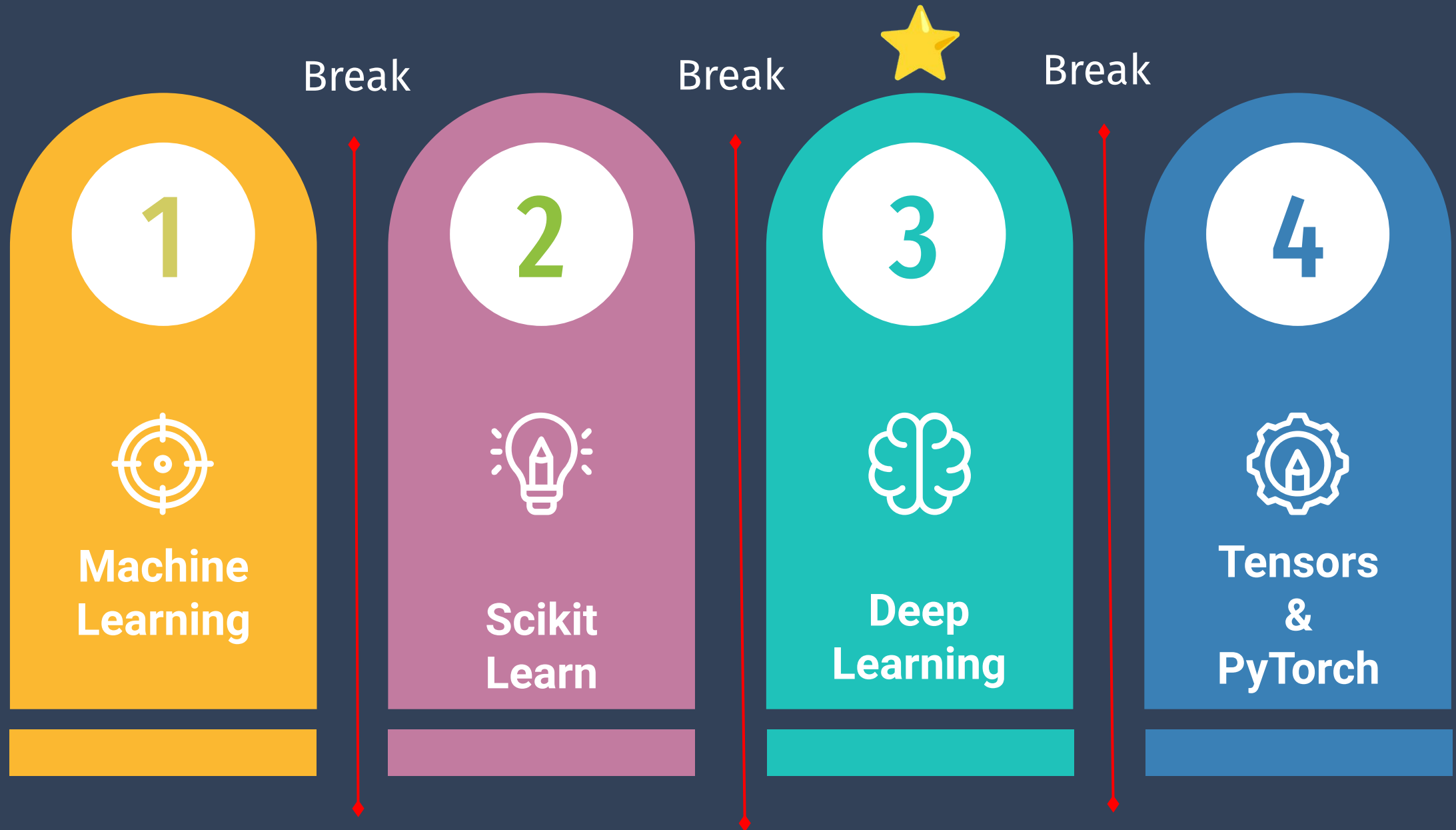
02

Scikit Learn and Train/Test in Scikit Learn

04

Hands-on Exercises and Practical Applications

Today's Agenda



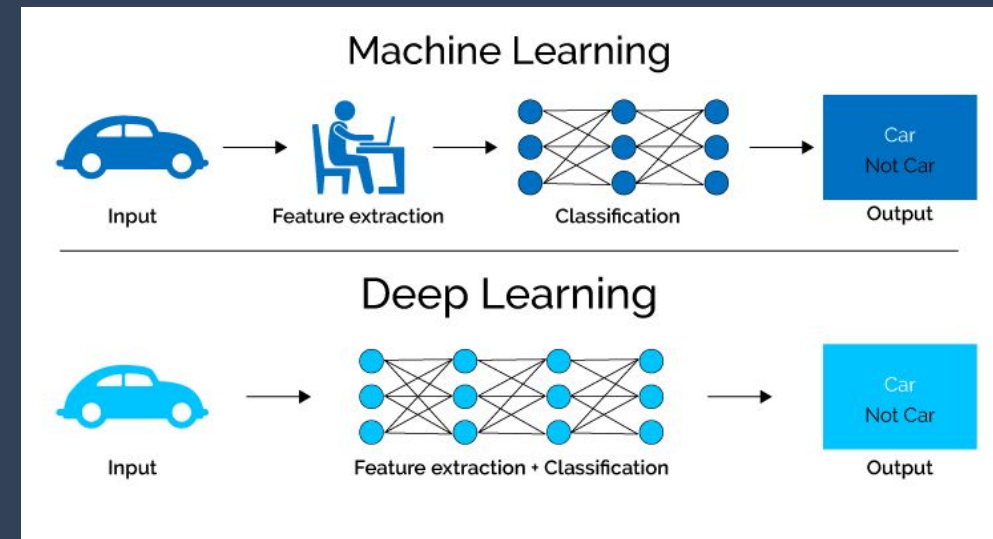
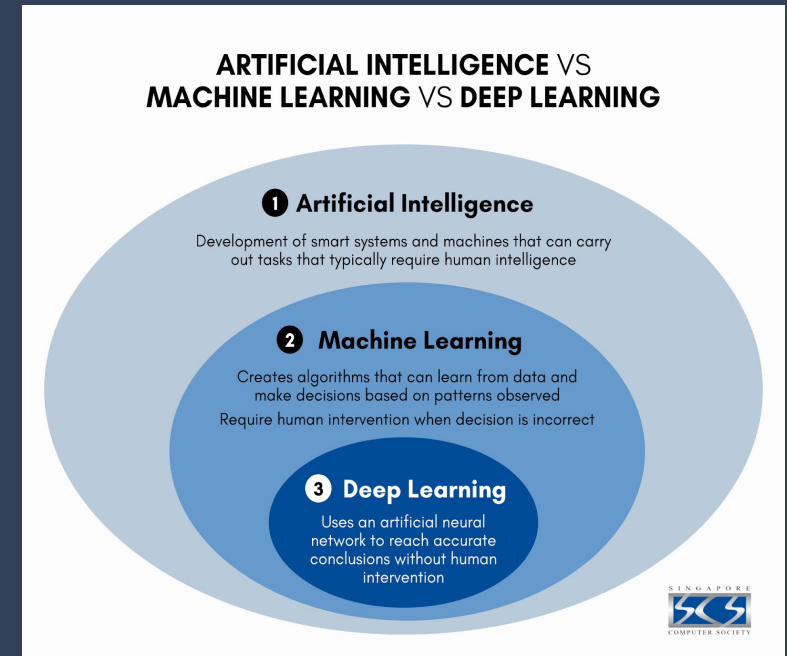
Deep Learning

Motivation

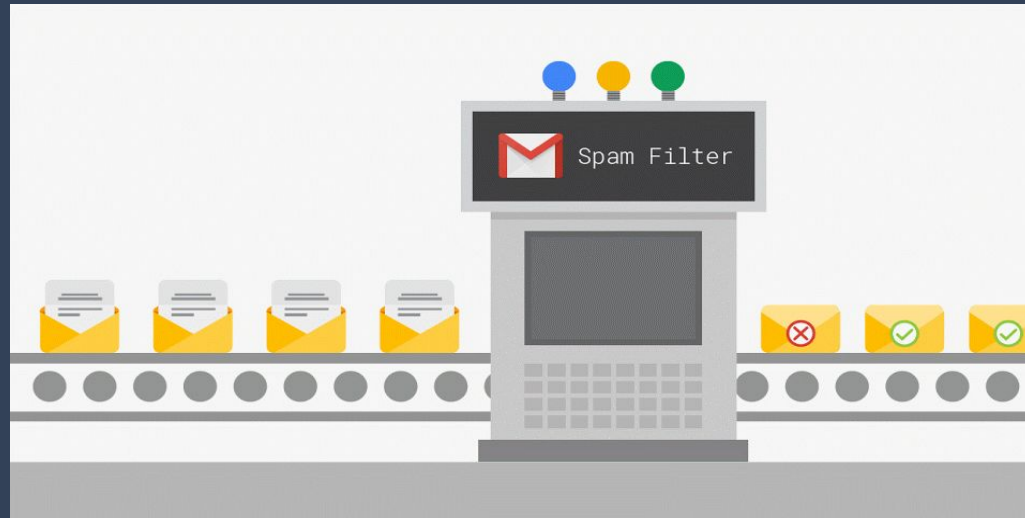
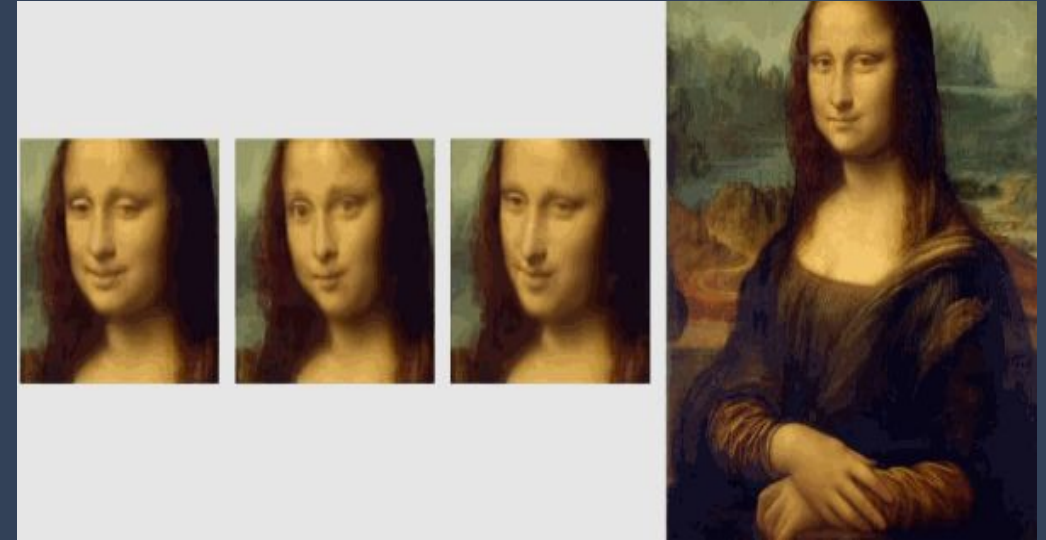
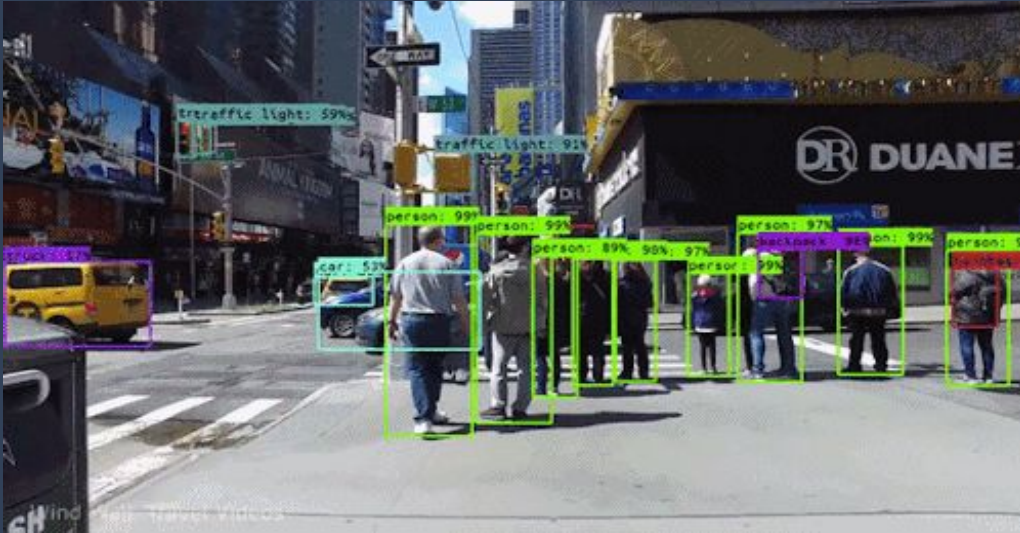
- Deep learning is revolutionizing various industries, including healthcare, finance, and self-driving cars. Understanding the fundamentals of deep learning helps to leverage its potential for solving complex problems and developing innovative applications.
- Deep Learning Models support **Multi Modality** i.e. it enables learning from data including Text, Vision, Audio, Tabular Data, simultaneously, like a human is capable of.
- Latest Applications like **ChatGPT (LLM), Dalle, Stable Diffusion** have led to a massive penetration into **Generative AI**, for example you can now create a movie without actors, without scriptwriter, without musicians (how cool is that)

Deep Learning

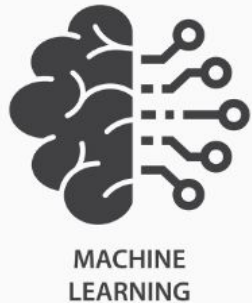
- Subset of machine learning
- Uses artificial neural network architectures to learn the complex relation between the input data and output variable
- Motivated from the complex biological neural structure of the brain
- Supports multi-modality capturing information from multiple inputs and types of data including Video, Images Audio.



Real-life application of deep learning

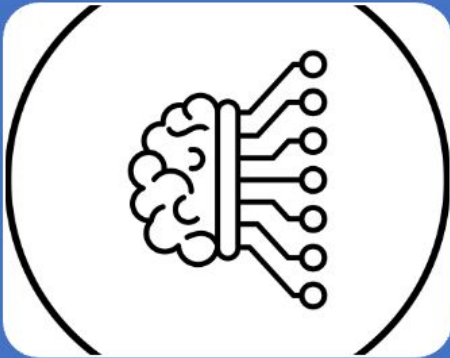


Machine Learning vs Deep Learning



ML

- ML is a subset of AI focused on the development of algorithms that allow computers to learn from and make predictions or decisions based on data. Instead of being explicitly programmed to perform a task, ML models improve their performance as they are exposed to more data.



DL

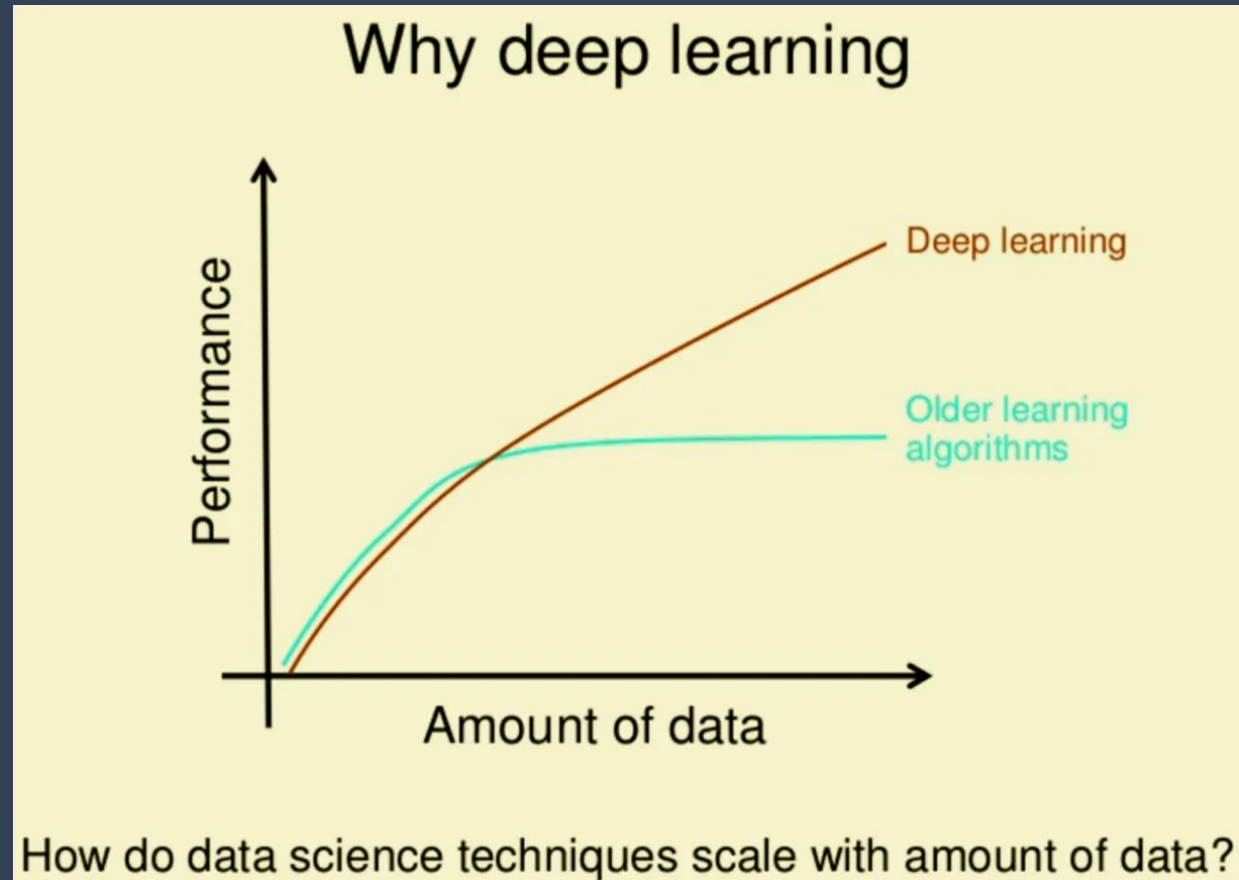
- DL (deep learning) is a subset of ML that uses neural networks with many layers (hence “deep”) to analyse various factors of data. These models are particularly effective for tasks like image and speech recognition. They can automatically discover representations from data, making them powerful for complex tasks.

How to identify deep learning problems?

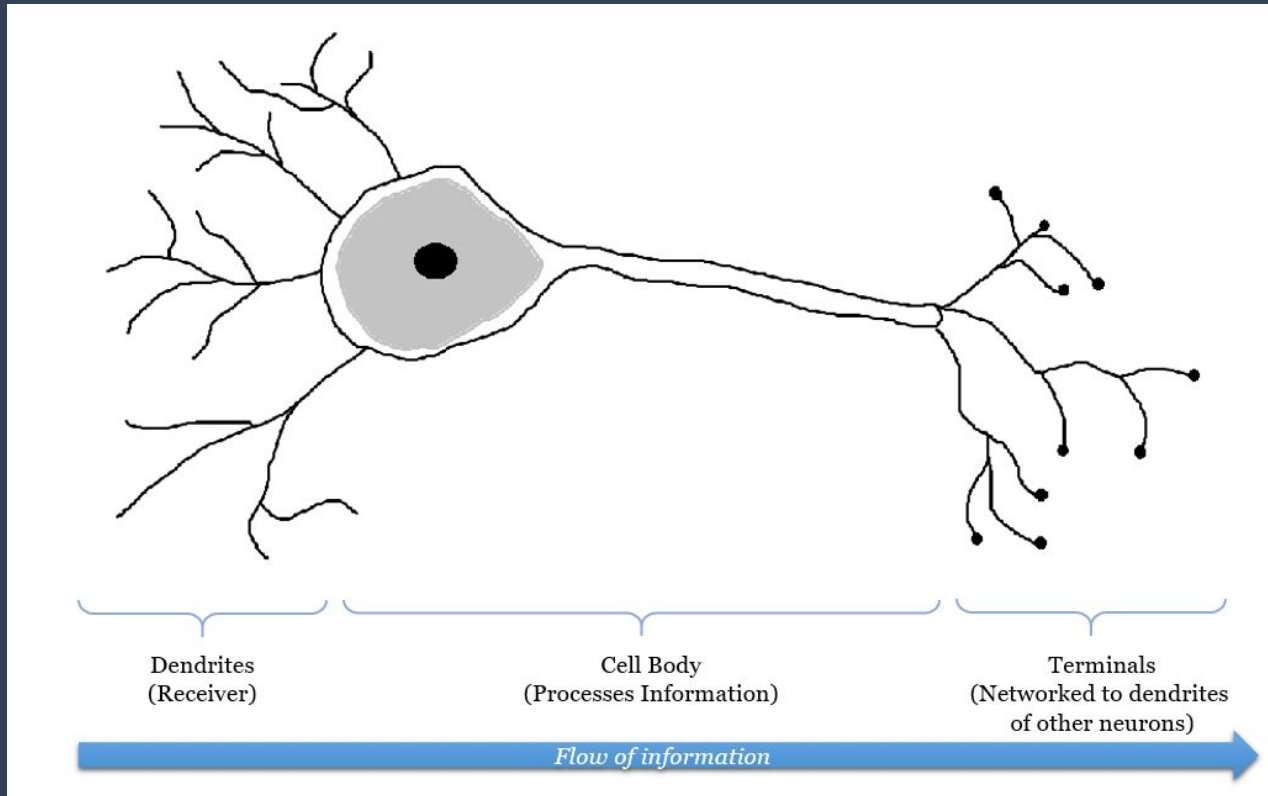
Large amount of high-dimensional, unstructured data, and you require the model to automatically learn intricate patterns and representations from the data

How to identify deep learning problems - Answer

1. *Unstructured or large amount of data*

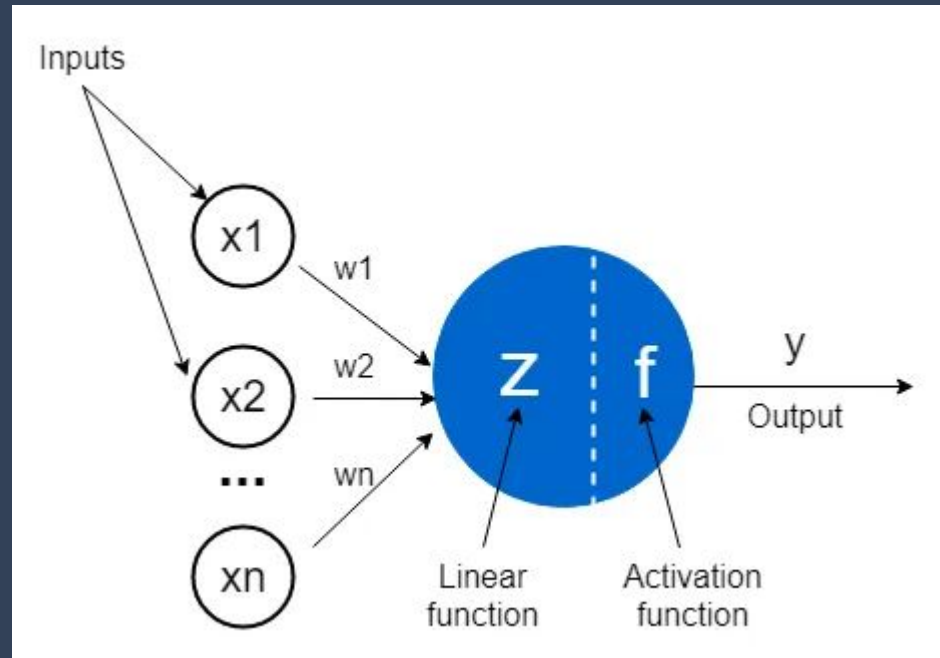


Biological Neuron



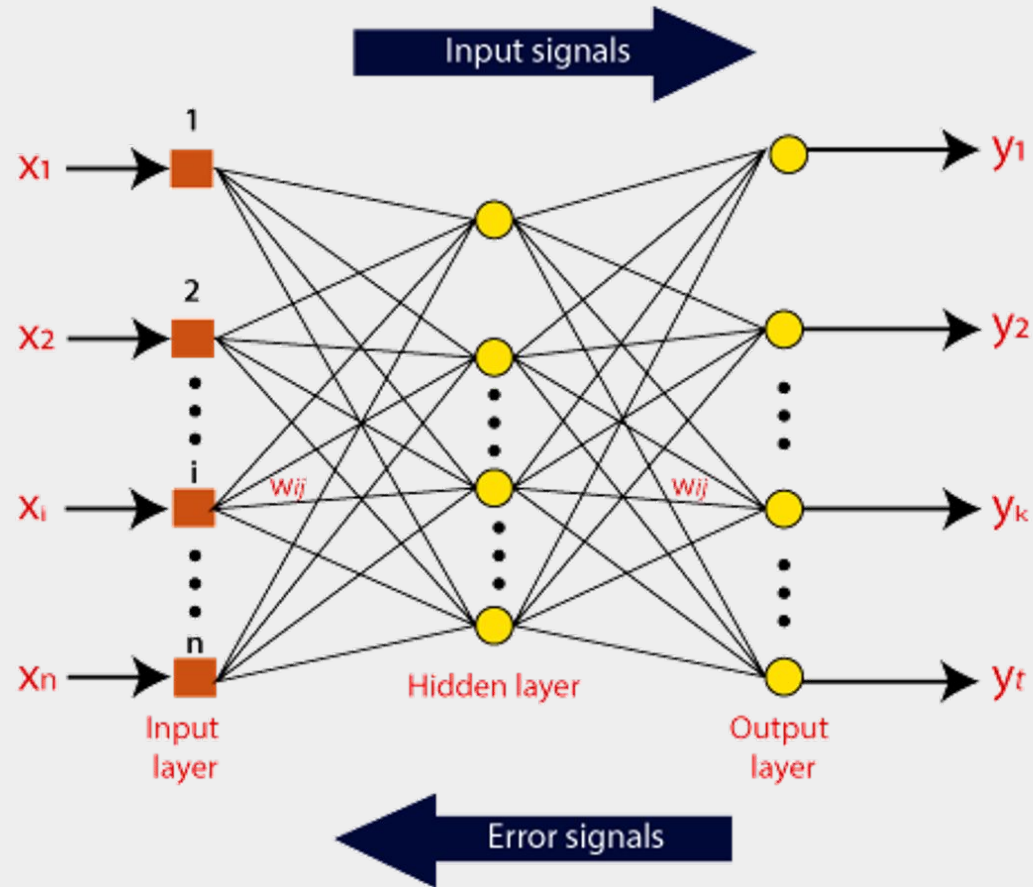
- Our brain has a large network of interlinked neurons, which act as a highway for information to be transmitted from point A to point B.
- To send different kinds of information from A to B, the brain activates a different sets of neurons, and so essentially uses a different route to get from A to B. This is how a typical neuron might look like.
- At each neuron, its dendrites receive incoming signals sent by other neurons. If the neuron receives a high enough level of signals within a certain period of time, the neuron sends an electrical pulse into the terminals. These outgoing signals are then received by other neurons.

Artificial Neuron



- An **artificial neuron** is a mathematical function conceived as a model of biological neurons, a neural network . Artificial neurons are elementary units in an artificial neural network.
- ANN is made of three layers namely input layer, output layer, and hidden layers.
- There must be a connection from the nodes in the input layer with the nodes in the hidden layer and from each hidden layer node with the nodes of the output layer.

Artificial Neural Networks



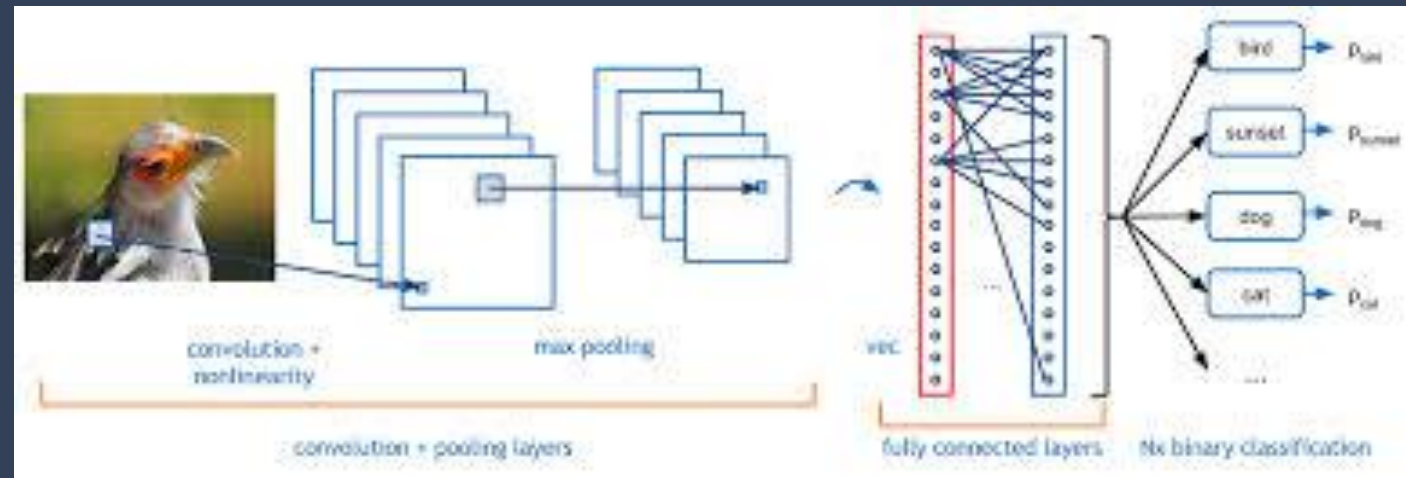
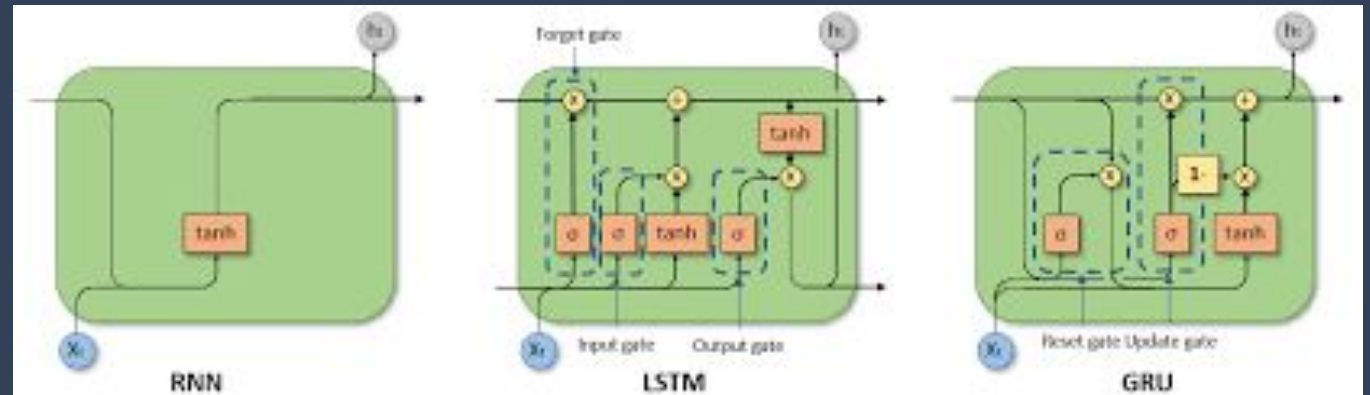
- Artificial Neural Network (ANN) is a combination of multi-layer neural network with neurons present at each layer.
- A basic ANN architecture consists of computational units and links.
- The links have different weights on themselves depending on the weightage of different connections across the network.

Why Deep Learning?

- **Learning Complex Patterns**
 - Models like Convolutional Neural Networks (CNNs) have learned to recognize complex patterns in images, including objects, animals, and scenes
- **Transfer Learning and Pretrained Models**
 - The pre-trained transformer based BERT/GPT model captures a rich understanding of language, making it a valuable starting point for various NLP tasks
- **Versatility and Adaptability**
 - Deep learning can handle multi-modal data, combining information from multiple sources (e.g., text, images, and audio) to perform tasks like multimedia content understanding or sentiment analysis in social media posts

Types of Neural Networks

- Artificial Neural Network
- Recurrent Neural Network,
- Long Short Term Memory,
- Gated Recurrent Units
- Convolutional Neural Networks
- Transformers
- Graph Neural Networks



Neural Network Limitations

- Neural Networks are not silver bullets.
 - Neural Nets are not immune to error and will make mistakes.
- May not be the most appropriate solution for a given problem.
 - As with all solutions, there is a time and place for neural nets. A simpler solution should always be checked/considered first.
- Require a lot of data.
 - Neural Networks can easily overfit. If there is not enough data, model will not be able to generalize to new data at inference.
- DS/ML teams may not be able to deploy NN.
 - Given the computational complexity of neural networks, companies may not have the capability to deploy models with current infrastructure.


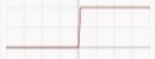









What is Gradient Descent?

- Gradient Descent is an optimization algorithm used to minimize the error in a neural network by iteratively adjusting the model's weights.
- **How It Works:**
 - Initial Weights: The process starts with initial random weights.
 - Error Calculation: The model's output is compared to the actual output to calculate the error.
 - Weight Update: Weights are adjusted to reduce the error using the gradient (slope) of the error function.
 - Iteration: This process is repeated iteratively until the error is minimized

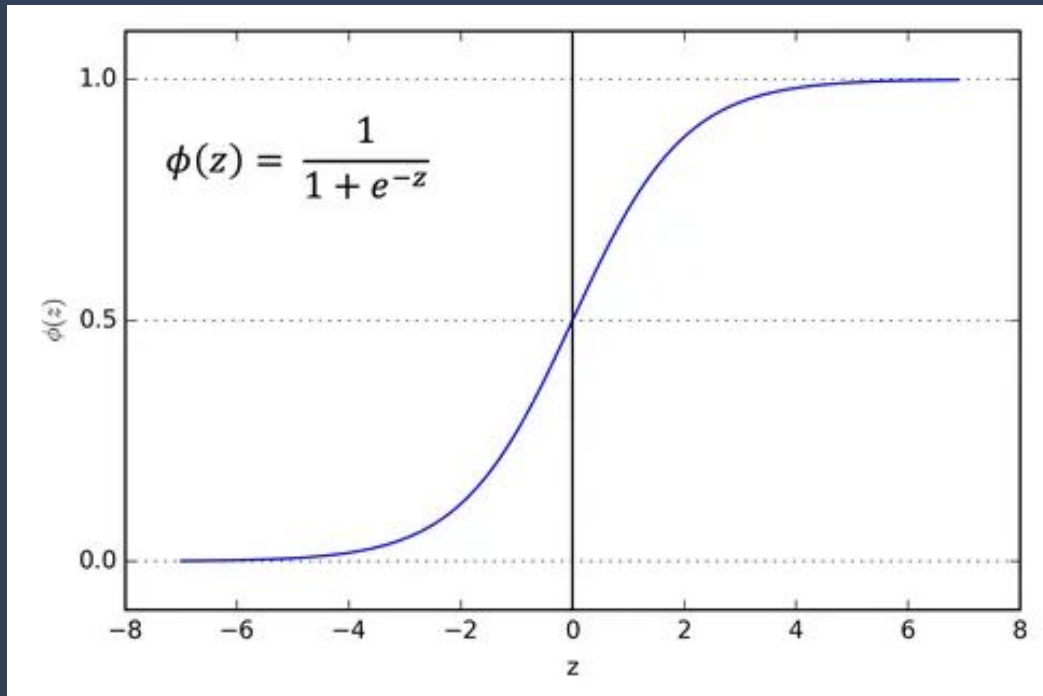


Activation Functions

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) ^[2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) ^[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

- The activation function calculates a weighted total and then adds bias to it to decide whether a neuron should be activated or not.
- The Activation Function's goal is to **introduce non-linearity** into a neuron's output.

Sigmoid



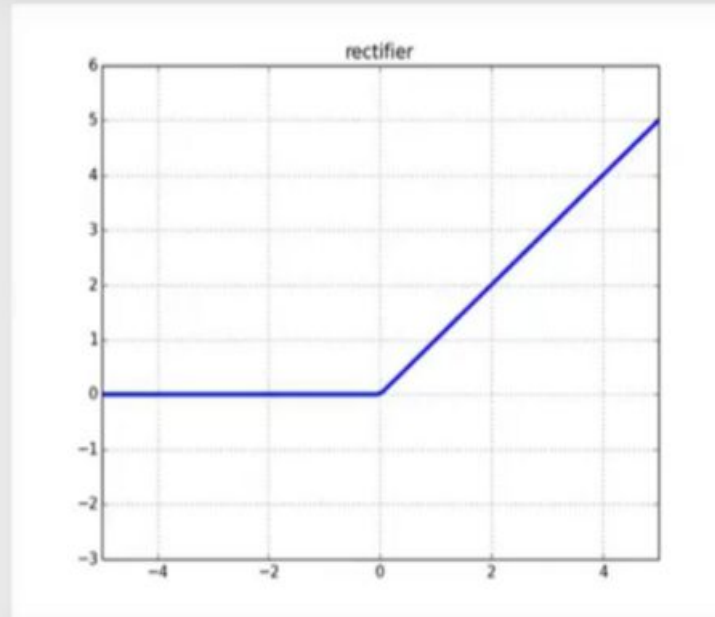
This function takes any real value as input and **outputs values in the range of 0 to 1**.

The larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to 0.0.

ReLu

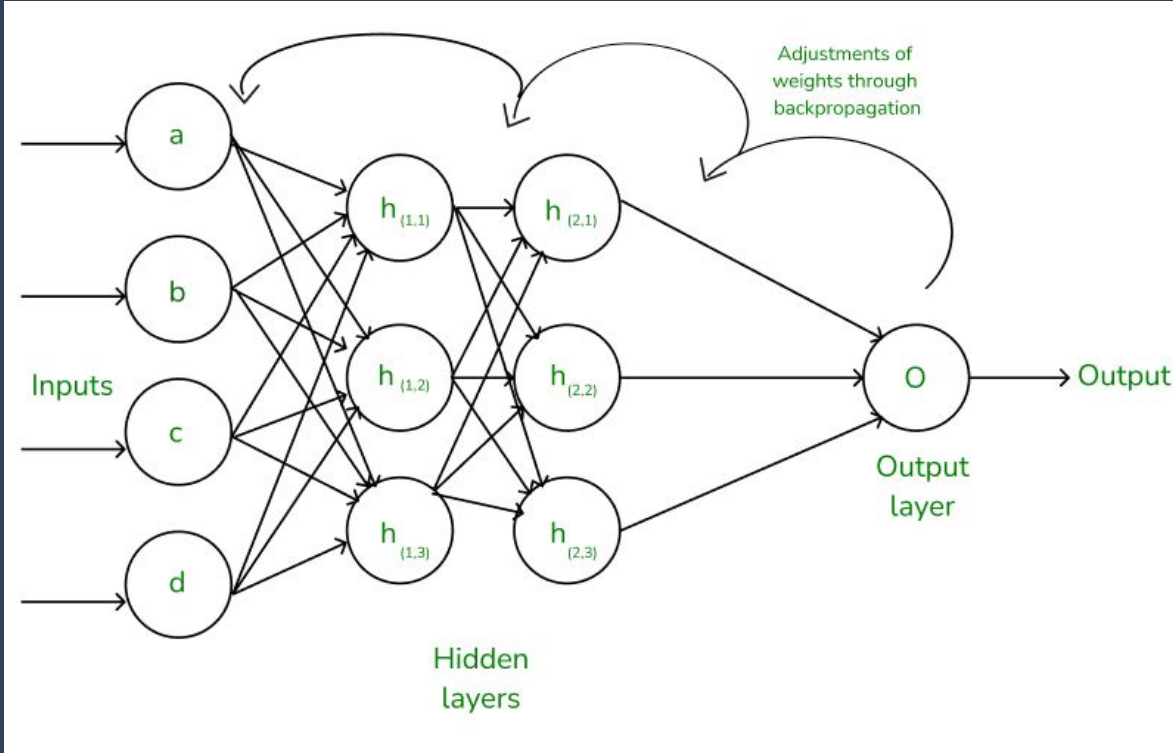
ReLU

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$



- ReLU is half rectified (from bottom). $f(z)$ is zero when z is less than zero and $f(z)$ is equal to z when z is above or equal to zero.
- The function and its derivative **both are monotonic**.
- **Issue:** All the negative values become zero immediately which decreases the ability of the model to fit or train from the data properly.
- In other words, any negative input affects the resulting graph by not mapping those values appropriately.

Backward Propagation



- Backpropagation is the essence of neural network training.
- It is the method of **fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch** (i.e., iteration).
- Proper tuning of the weights allows you to reduce error rates and make the model reliable by increasing its generalization .
- The two main formulas in backpropagation are
 - Chain Rule**
 - Weight Updates**



Quiz

What type of problems are typically well-suited for deep learning models?

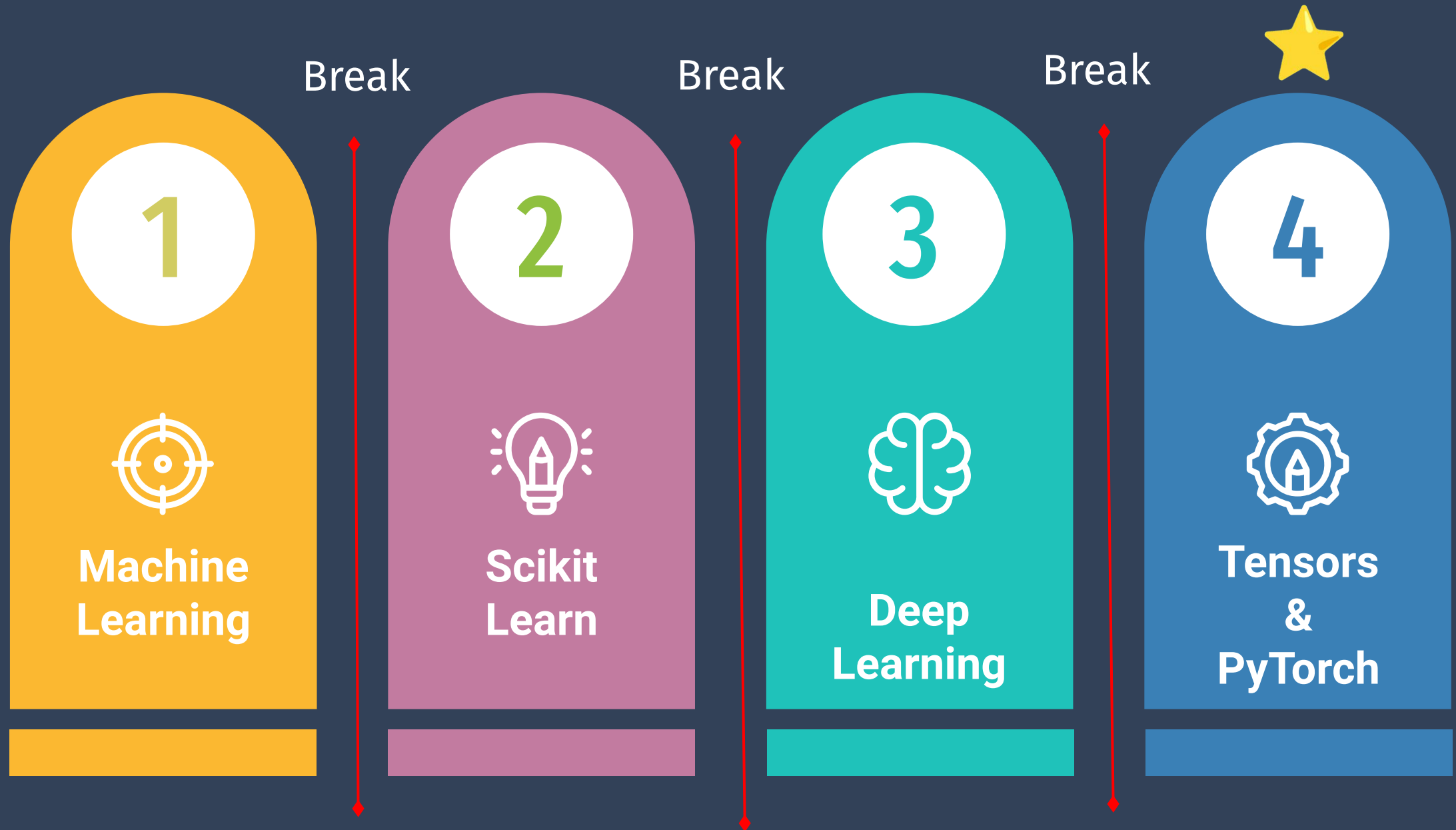
- A** Structured data with a small number of features.
- B** Tasks that require learning complex patterns from large amounts of high-dimensional, unstructured data.
- C** Simple classification tasks with binary outcomes.
- D** Problems where data is scarce and highly structured.

Quiz

What type of problems are typically well-suited for deep learning models?

- A** Structured data with a small number of features.
- B** Tasks that require learning complex patterns from large amounts of high-dimensional, unstructured data.
- C** Simple classification tasks with binary outcomes.
- D** Problems where data is scarce and highly structured.

Today's Agenda



Tensors

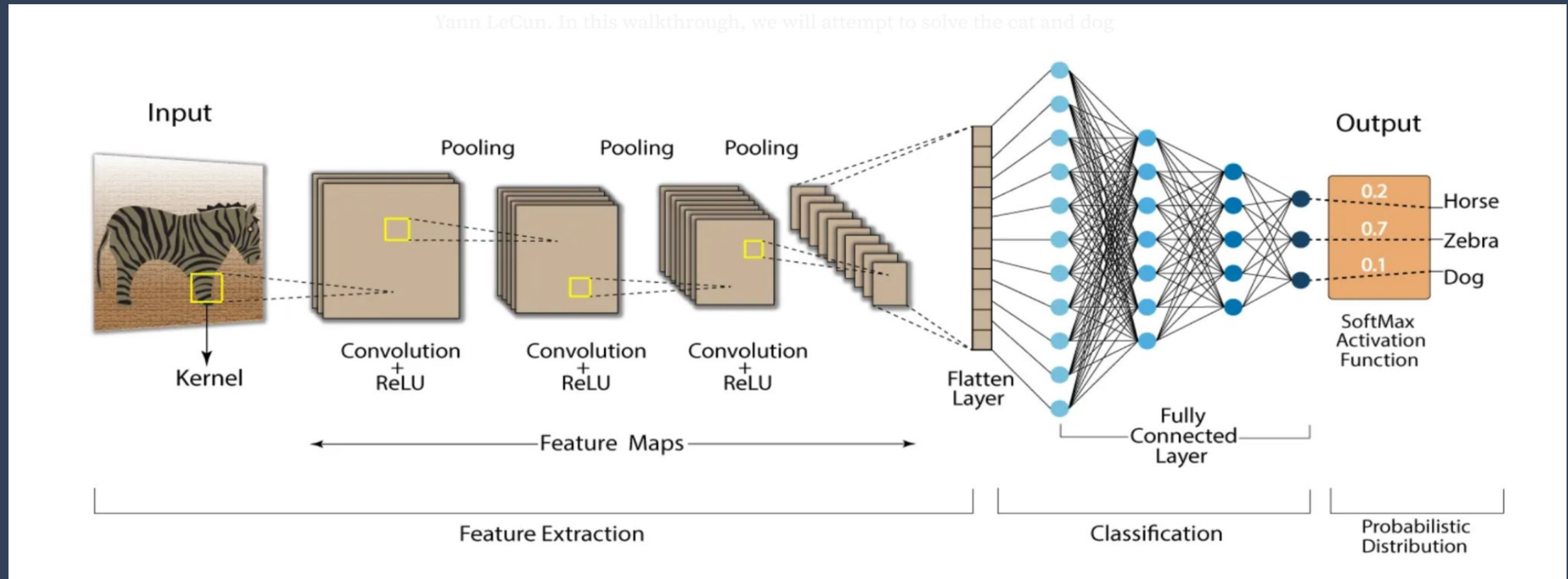
What are Tensors?

- Mathematical representation of data (any data - image/text/audio converted to numerical format) used in DL
- Tensors are mathematical objects used to represent multi-dimensional arrays of numbers.
- **Example:** In image processing, a tensor can represent a grayscale image, where each element is a pixel value.



Where do tensors come into play?

Let's start from the general representation of a deep learning model



PyTorch

What is PyTorch?

PyTorch is an open-source deep learning framework developed by Facebook's AI Research lab (FAIR).

PyTorch is known for its intuitive and easy-to-use API, which makes it a favorite among researchers and developers for building and training neural networks.



Features of PyTorch

- **Tensors and Dynamic Graphs:** Similar to NumPy arrays, GPU support for faster computation.
- **Autograd:** Computes gradients for backpropagation automatically.
- **TorchScript:** Creates serializable and optimizable models for production.
- **Optimizers and Loss Functions:** Built-in optimization algorithms and loss functions.

Advantages of PyTorch

- Rich Ecosystem
- Includes libraries like TorchVision, TorchText, and TorchRL.
- Provides tools and pre-trained models.
- GPU Acceleration
- Integrates with CUDA for faster deep learning computations.

How Does This Help Me?

Tensors

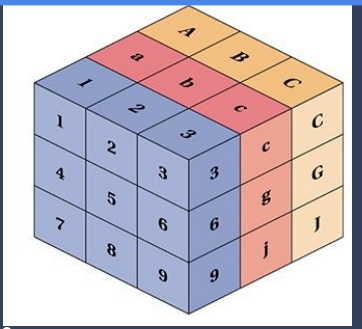
- **Tensors are the basic building blocks for all neural network computations**
- **Tensors integrate well with libraries like PyTorch and TensorFlow**

PyTorch

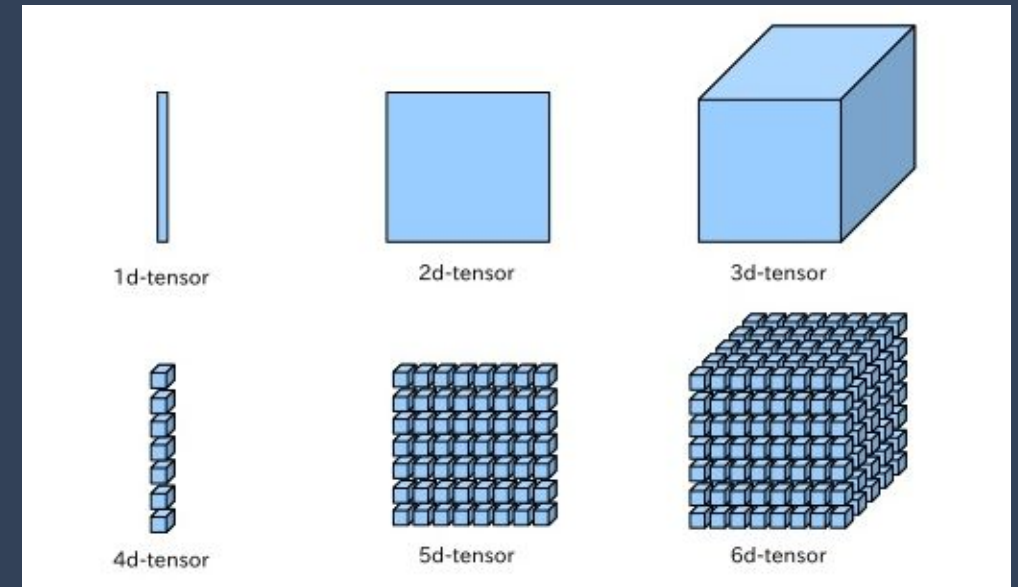
- **PyTorch offers dynamic computational graphs, which makes it easier to modify the network architecture on-the-fly.**
- **PyTorch provides high flexibility for research and deployment, ensuring efficient performance on both CPU and GPU.**

Tensor Operations

Common Tensor Operations



- **Creation:** Initializing tensors with predefined values or random data.
- **Indexing & Slicing:** Accessing and modifying specific elements or subsections of tensors.
- **Reshaping:** Changing the shape of a tensor without altering its data.
- **Arithmetic Operations:** Performing element-wise and matrix operations such as addition, subtraction, multiplication, and division.



Tensor Operations

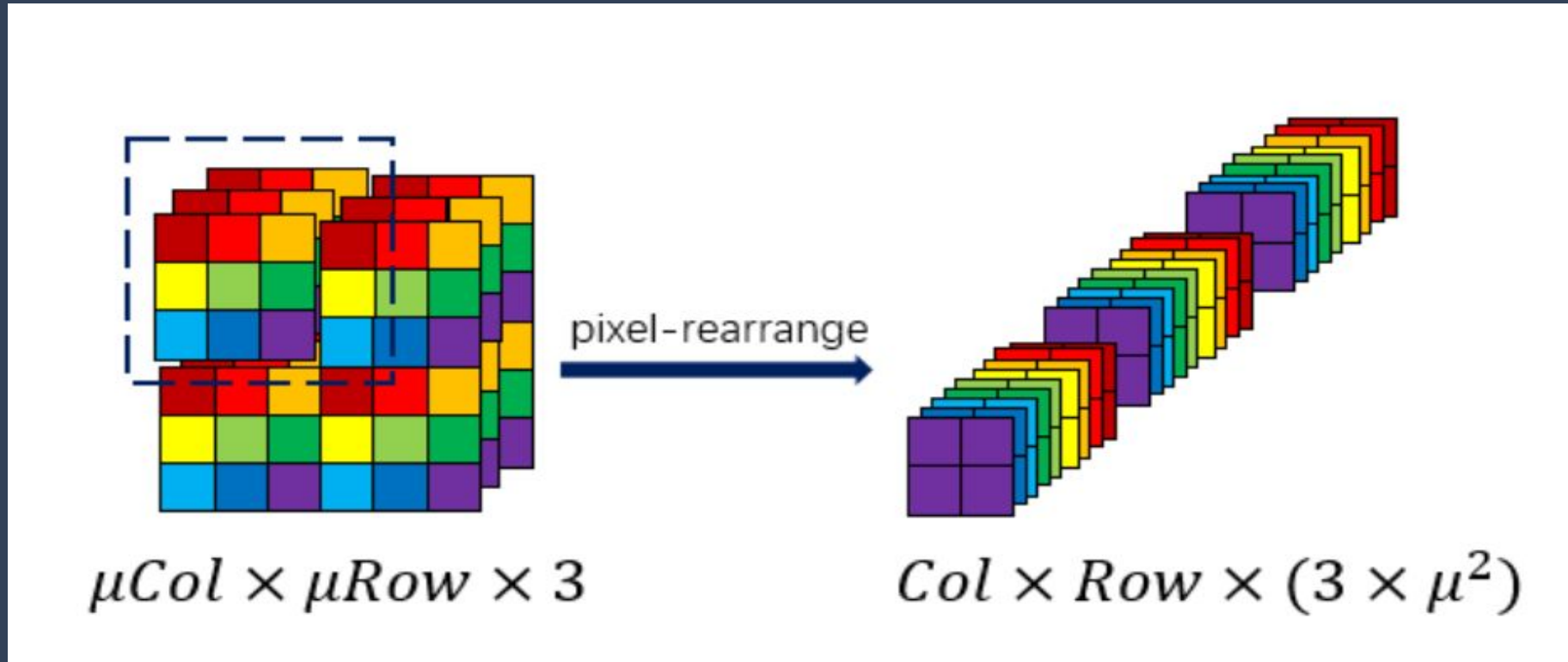
Demo Time

Applications

- Computer Vision: Image recognition, object detection, and segmentation.
- Natural Language Processing: Text generation, sentiment analysis, and translation.
- Speech Recognition: Converting speech to text and understanding spoken language.
- Recommender Systems: Providing personalized content suggestions.
- Robotics: Control systems and autonomous navigation.
- Medical Imaging: Analyzing medical scans and diagnosing conditions.
- Scientific Computing: Simulating physical systems and solving differential equations.

Tensor Reshaping and Visualization

Let's understand what tensor reshaping is



Source

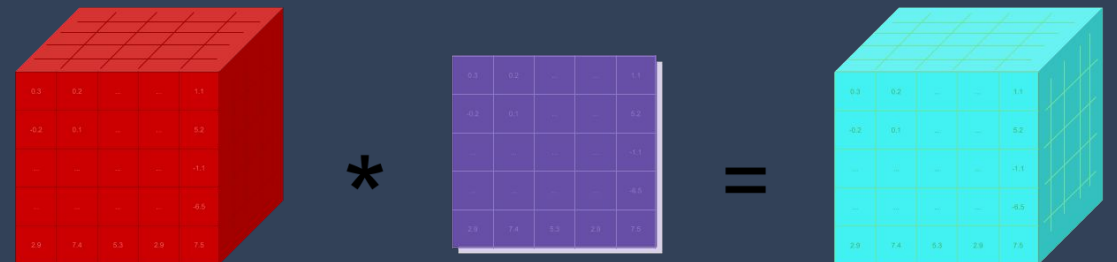
We reshape a tensor either to match dimensions to perform an operation, OR to enable batch processing (eg: multiple tensors into one tensor), OR for memory optimization OR for visualizing a tensor.

Tensor reshaping with reshape()

Most used methods for tensor reshaping:

- view()
- reshape()

Question: view() and reshape() perform the similar action of reshaping a tensor. Why do you think there are two methods for the similar objective?



Tensor reshaping with view()

- This function returns a view of the original tensor with the specified shape.
- **torch.view()** operates on contiguous tensors only.
- If the original tensor is contiguous (meaning the memory is laid out as a single block), view() will return a new tensor with the desired shape.
- If the original tensor is non-contiguous, view() will raise an error.
- It is recommended to use view() when working with contiguous tensors.

Tensor reshaping with reshape()

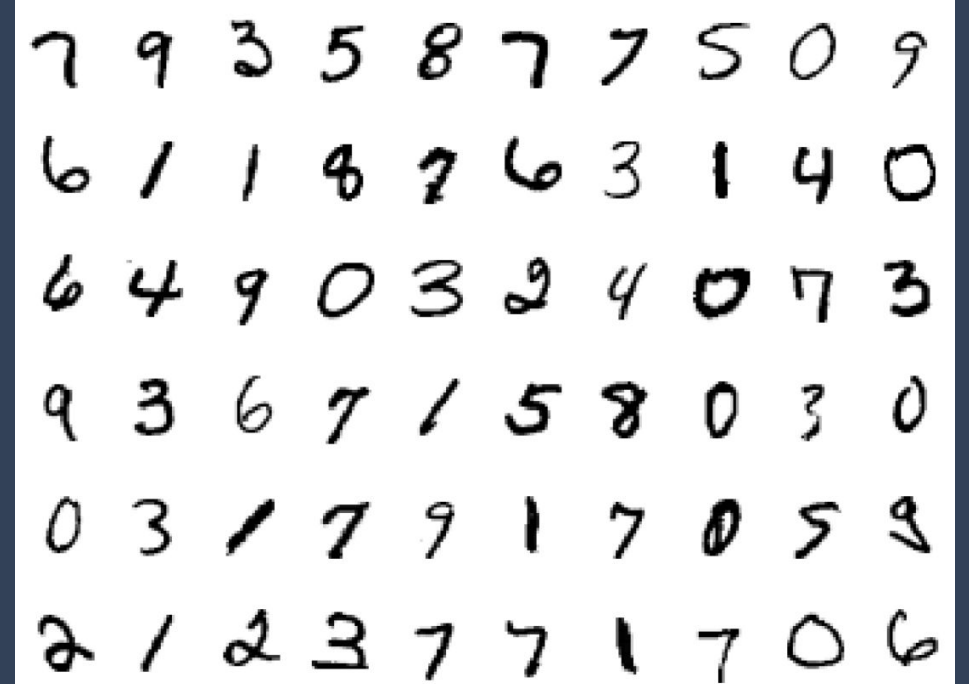
- This function returns a new tensor with the desired shape
- Operates on contiguous and non-contiguous tensors.
- If the desired shape is compatible with the number of elements in the original tensor, reshape() will return the reshaped tensor.
- If the desired shape is incompatible with the number of elements, it will raise an error.

Tensor Reshaping and Visualization

Demo Time

Handwritten Digit Recognition using PyTorch

- Handwritten Digit Recognition involves build and train a simple neural network to detect handwritten digits with PyTorch
- Widely used in postal services, banking, and form data entry.



Handwritten Digit Recognition using PyTorch

Summary

01

What is Deep Learning?

02

Different Components in Deep Learning

03

Tensors and PyTorch

04

Common Tensor Operations

05

Hands-on Exercises and Practical Applications

Next Steps

- [Revise the concepts](#) learnt in today's session through class recording and post-class videos
- Attempt the [Assignment & Quiz questions](#) available on Uplevel
- Register for [Technical Coaching Session](#) through Xpert Connect option on Uplevel
- Attend the [Assignment Review Session](#) to get a walkthrough of the assignment solution

What's Coming Next?

Now that you have grasped the Python Essentials for GenAI, it's time to apply some of those!

The next module is Hands-on with GenAI Models!!

Do remember to read the instructions and buy OpenAI credits for yourself to make the best out of the next session!

Thank
you