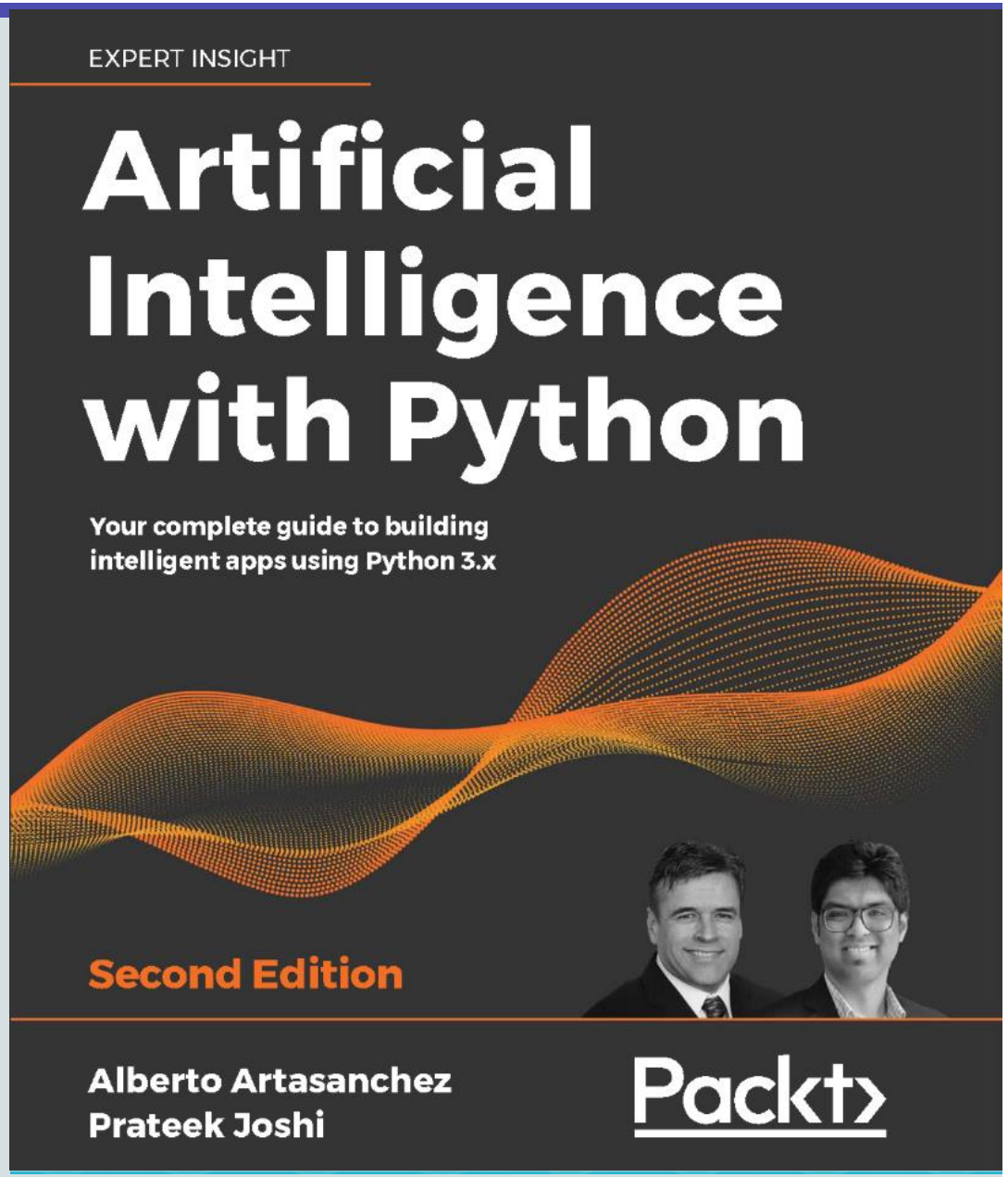
University of South-Eastern Norway
School of Business

Lecture 9: Deep Learning & Reinforcement Learning

Note: This pdf will be updated after lecture 9

Sinuo Wu.

Course: AI for Business Applications (AI3000)



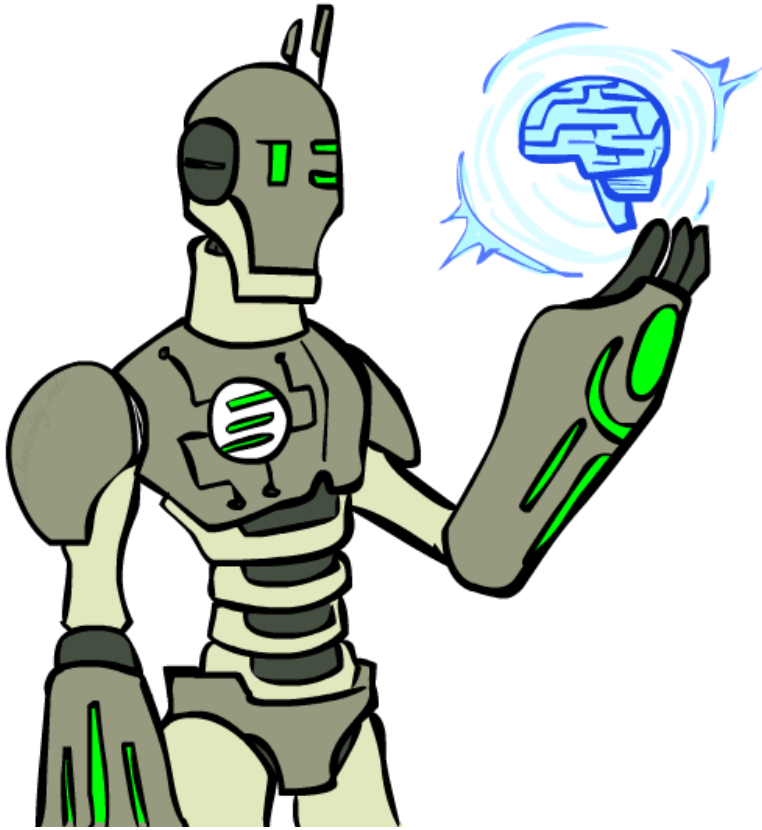
University of South-Eastern Norway
School of Business

Quiz

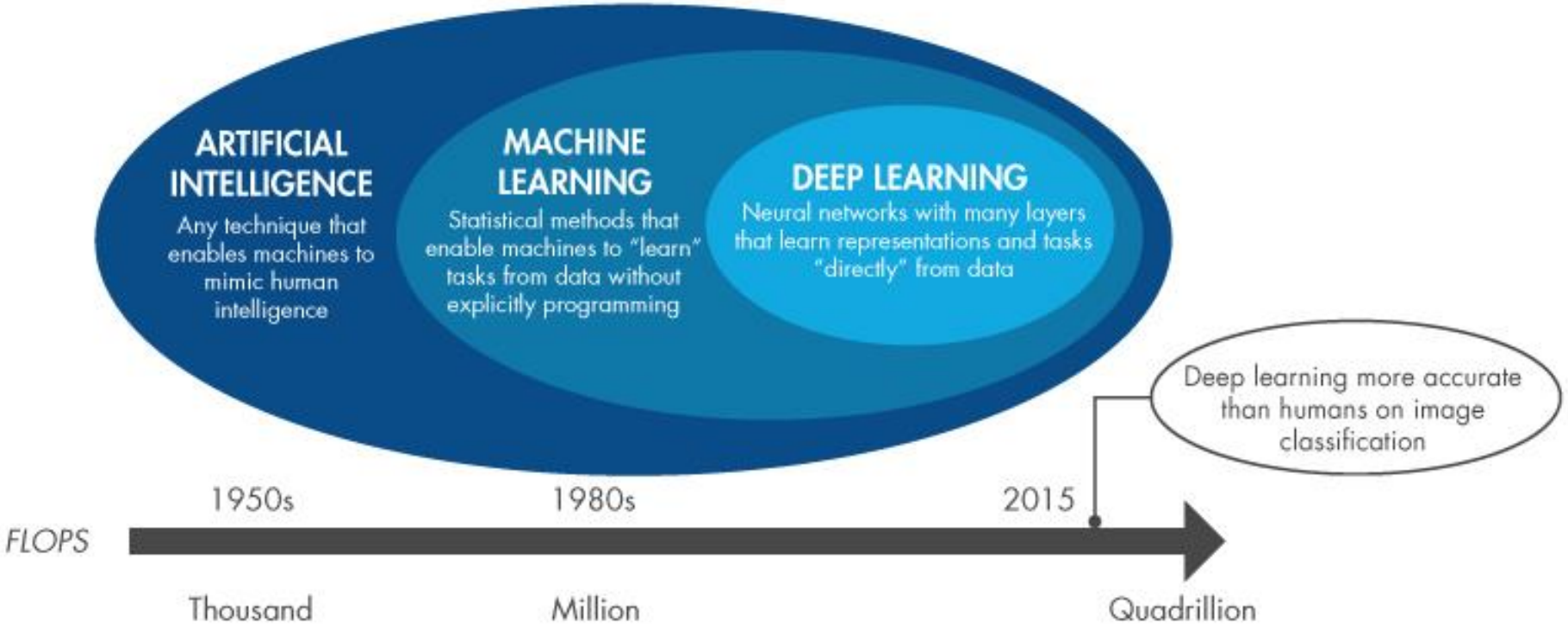
- Enter room number or Scan the QR code

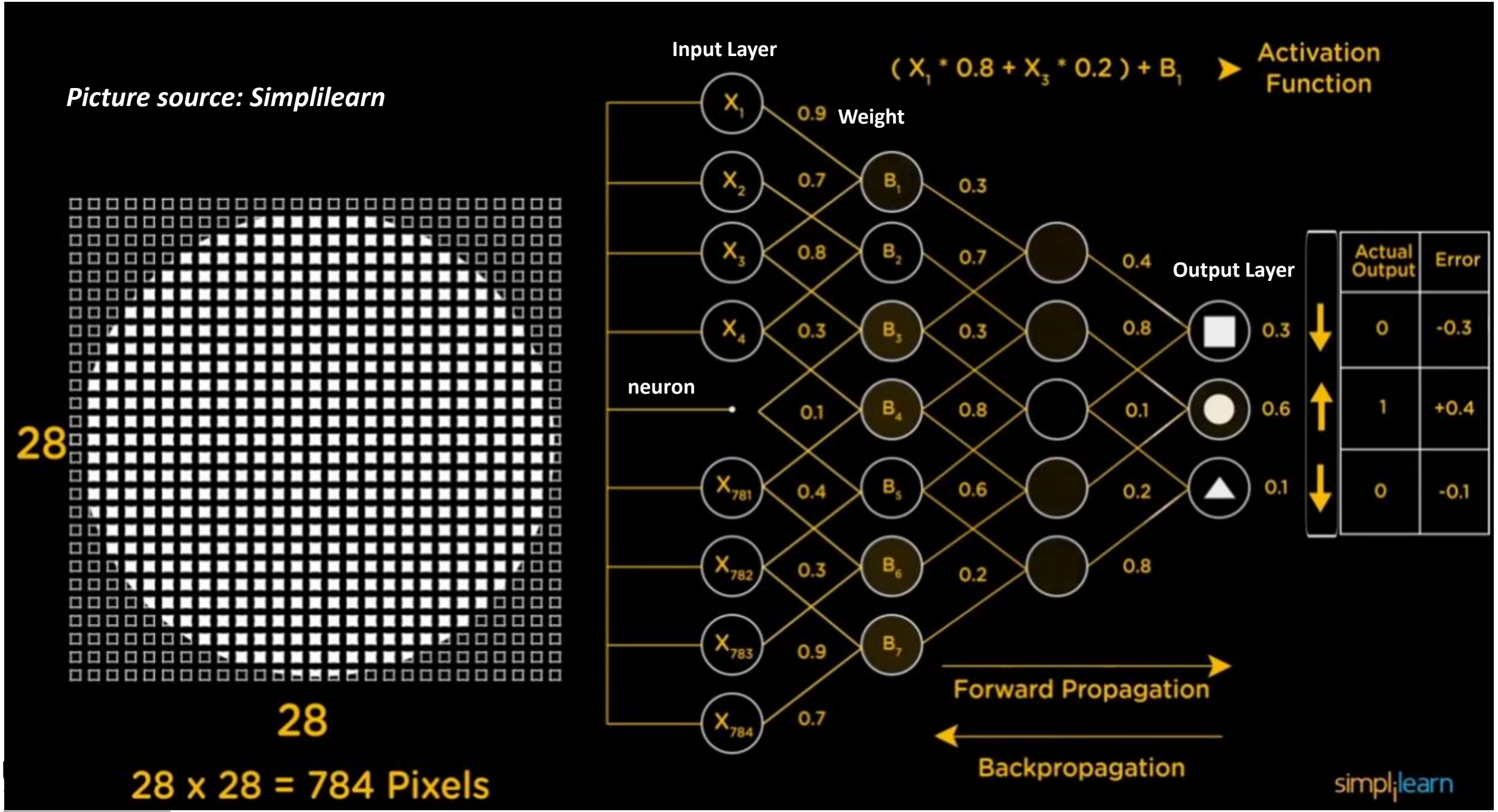
In this lecture

- Deep Learning with Convolutional Networks
  - The basics of CNNs
  - The architecture of CNNs
  - The types of layers in a CNN
- Reinforcement Learning
  - Reinforcement learning versus supervised learning
  - Real-world examples of RL

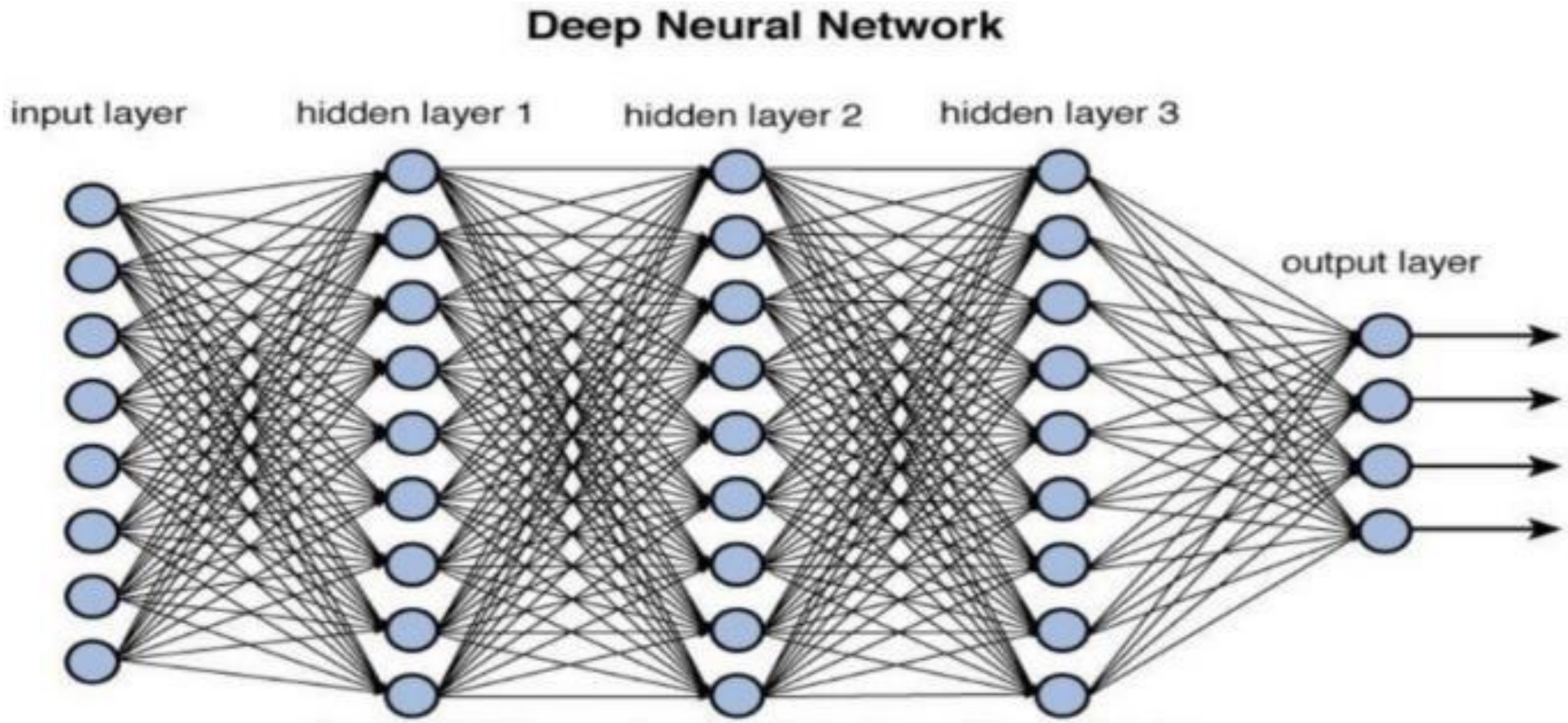


AI- ML- DL


ARTIFICIAL INTELLIGENCE
Any technique that enables machines to mimic human intelligence
MACHINE LEARNING
Statistical methods that enable machines to "learn" tasks from data without explicitly programming
DEEP LEARNING
Neural networks with many layers that learn representations and tasks "directly" from data
Deep learning more accurate than humans on image classification
1950s
1980s
2015
FLOPS
Thousand
Million
Quadrillion


Picture source: Simplilearn
28
28
28 x 28 = 784 Pixels
Input Layer
(X1 * 0.8 + X3 * 0.2) + B1
Activation Function
Weight
neuron
Output Layer
Actual Output
Error
0
-0.3
1
+0.4
0
-0.1
Forward Propagation
Backpropagation
simplilearn

Deep Neural Network



Deep Learning

- Deep learning is a subfield of machine learning that focuses on the development of **artificial neural networks**, inspired by the structure and function of the human brain, to solve complex tasks.
- It is a type of machine learning where algorithms attempt to learn and make decisions directly from data**, without relying on explicit programming.**
- Deep learning models are typically built using artificial neural networks, which consist of **interconnected layers** of artificial neurons or "nodes."

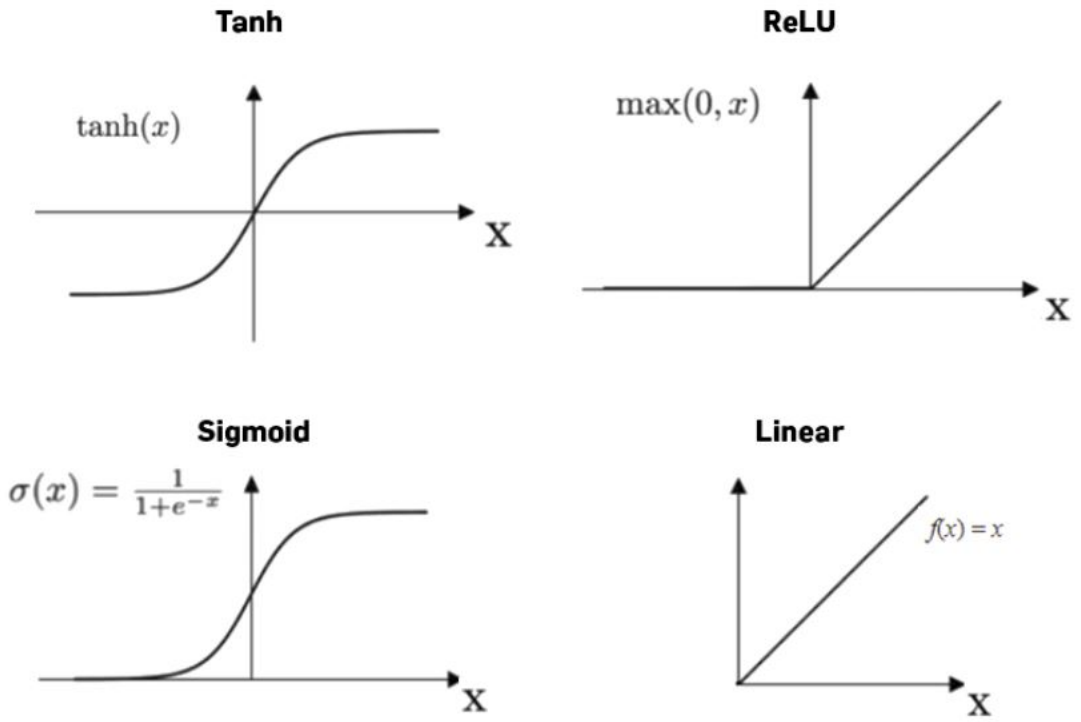
Deep Learning

- Features picked up by NN
- Most efficient way to deal with unstructured data
  - Massive volume of data to train
- Computational Power (GPU)
  - Expensive!
- Time
  - Hours, months!

Convolutional NN

- **Structure:** Conventional neural networks consist of interconnected layers of artificial neurons, including input, hidden, and output layers.
- **Training:** They are usually trained through supervised learning, adjusting weights to minimize error using techniques like backpropagation.
- **Activation Functions:** Neurons use activation functions like ReLU, sigmoid, or tanh to process input.
- **Applications:** CNNs assume that the inputs are images, which allows them to extract properties specific to images.

Activation Functions

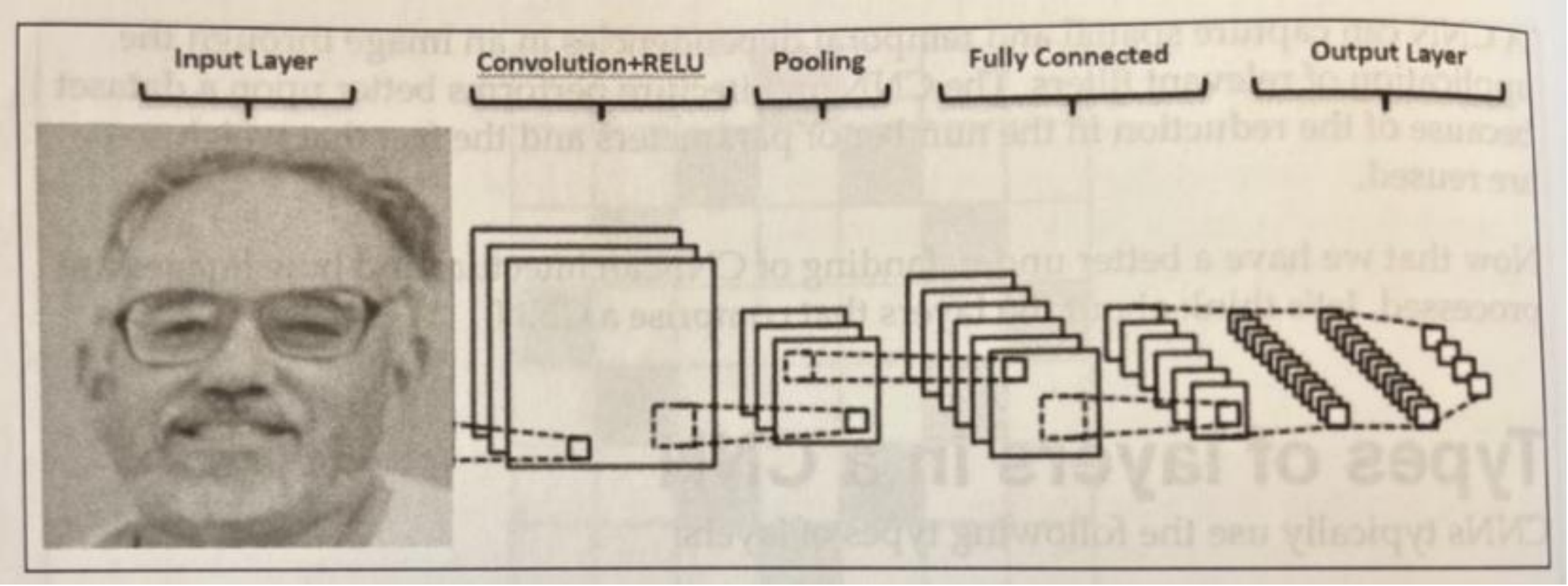


https://www.debadityachakravorty.com/ai-ml/activationfunc/

Convolutional NN Layers

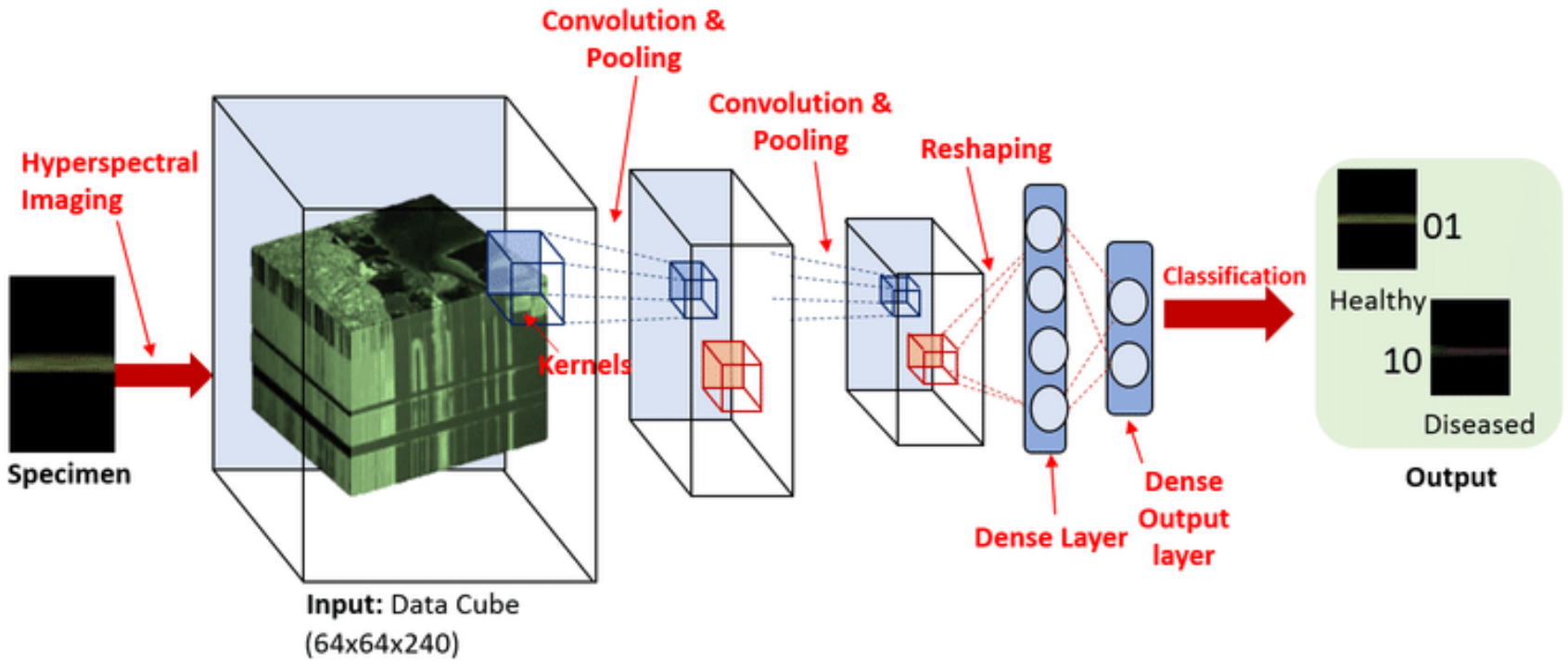
- **Input layer:** This layer takes the raw image data as it is.
- **Convolutional layer:** This layer computes the convolutions between the neurons and the various patches in the input.
- **Rectified Linear Unit layer:** This layer applies an activation function to the output of the previous layer. This function is usually something like max(0, x).
- **Pooling layer:** This layer samples the output of the previous layer resulting in a structure with smaller dimensions. Pooling helps us to keep only the prominent parts as we progress in the network.
- **Fully Connected layer:** This layer computes the output scores in the last layer.
- **Output layer:** This layer shows the probability of the predicted results.

Convolutional NN Example



(Artasanchez & Joshi, 2020)

Convolutional NN Example



Explaining hyperspectral imaging-based plant disease identification: 3D CNN and saliency maps (Nagasubramanian & Jones)

Generative Neural Networks

- https://thispersondoesnotexist.com/
- https://www.artbreeder.com/

- Given a dataset, GAN learns to create new data examples similar to the training set.
  - Created for unsupervised learning

- **Specialized CNN type**.
- **Deep-fake**
- **Data manipulation**
- **Privacy (one-time throwaway keys)**

Recurrent NN

- **Sequential Data:** RNNs are for processing sequences like text and time series.
- **LSTM and GRU:** Specialized RNN types handle long-term dependencies.
- **Applications:** Used in NLP, speech recognition, time series, and more.
- **Challenges:** Can be computationally intensive and suffer from overfitting.
- **Unrolling:** Visualizes RNN as a chain of network copies over time.
- **Real-time Processing:** Specialized hardware for real-time tasks.

Transformer

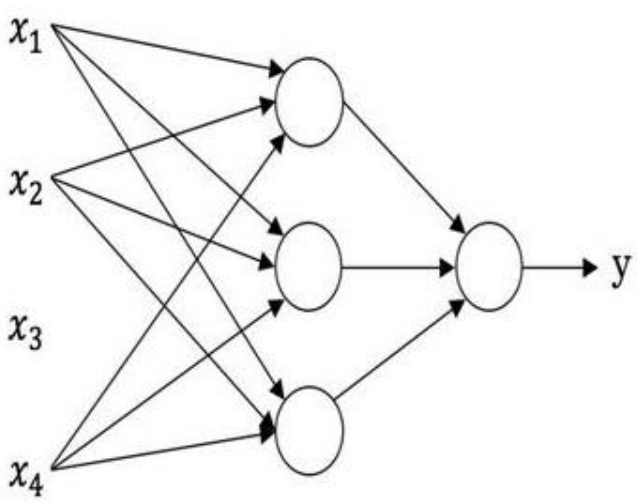
https://www.youtube.com/watch?v=zxQyTK8quyY

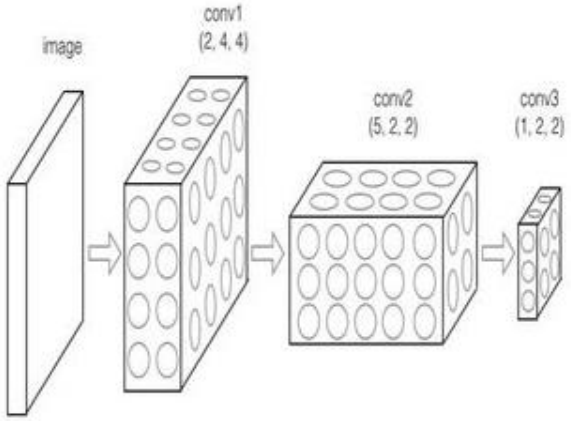
- The Transformer is a deep learning architecture introduced in the paper "Attention is All You Need" by Vaswani et al. in 2017. It has since become a pivotal innovation in the field of natural language processing and machine learning.

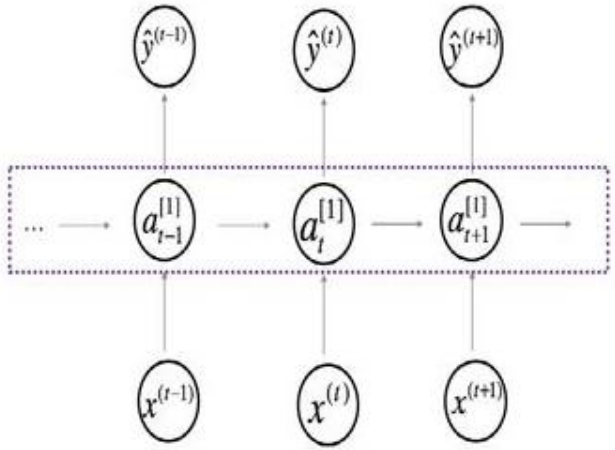
- At its core, the Transformer uses self-attention to weigh and focus on different parts of the input sequence. This allows it to capture relationships and dependencies between elements in the sequence.

- For ChatGPT, which is a variant of the GPT (Generative Pre-trained Transformer) model, the working mechanism is rooted in the Transformer architecture but tailored for generating human-like text responses in a conversational context.

Main types







Standard NN **Convolutional NN** **Recurrent NN**

Supervised Learning

Structured data

staffNo	fName	lName	salary
SL21	John	White	30000
SG37	Ann	Beech	12000
SG14	David	Ford	18000
SA9	Mary	Howe	9000
SG5	Susan	Brand	24000
SL41	Julie	Lee	9000

Unstructured data

In a peaceful village, a shepherd named Alex used supervised learning to predict the coat color of newborn lambs based on their parents' traits. Over time, he refined his model and achieved remarkable accuracy, setting an example for others to use machine learning in their own endeavors.

Supervised Learning

Input (x)	Output (y)	Applications	
Home features	Price	Real Estate	Standard NN
Ad, user information	Click on ad ? (0/1)	Online Advertising	
Image	Object (pixel values)	Photo tagging	CNN
Audio	Text transcript	Speech Recognition	RNN
English	Norwegian	Machine translation	
Image, Rader infor	Position of other cars	Autonomous driving	Customized NN

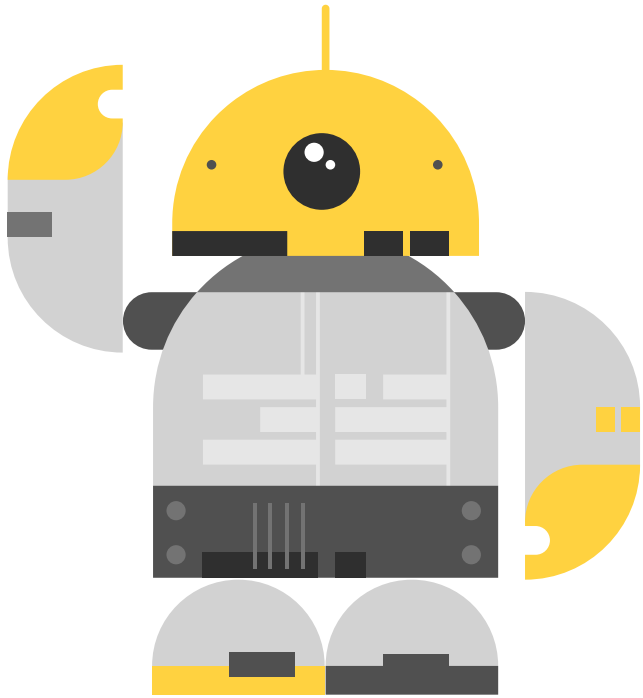
Have a Break!

Reinforcement Learning

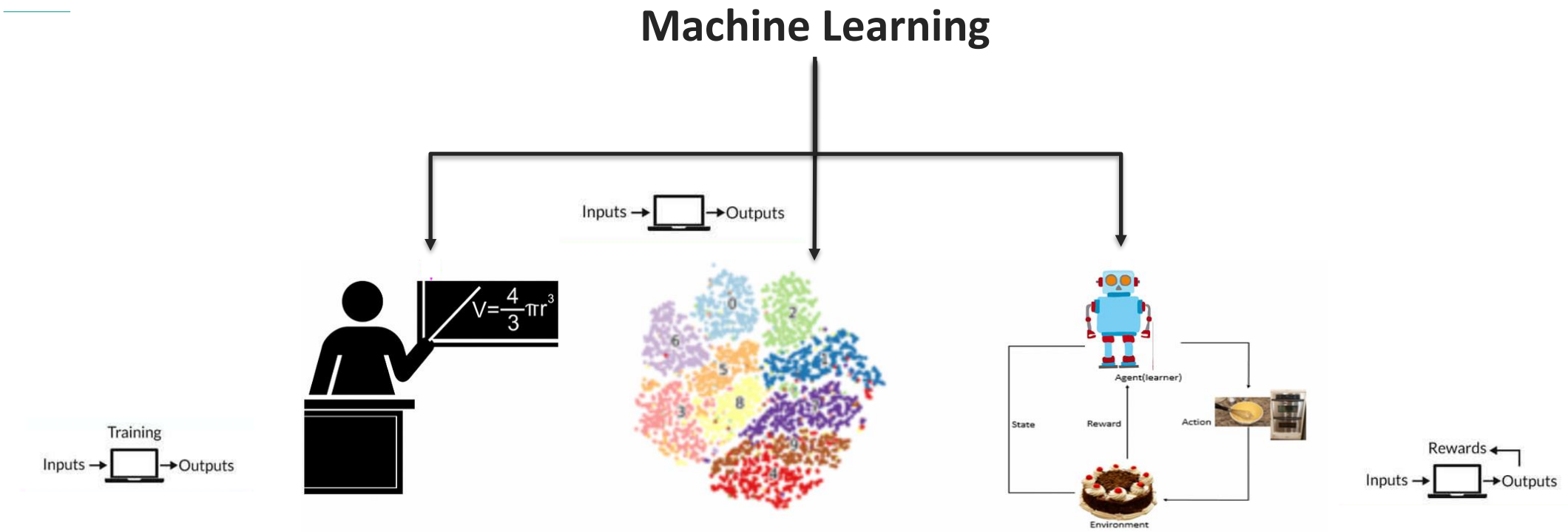
RL

- Process of learning what to do and mapping situations to certain actions to maximize the reward.

- Machine learning paradigm where an agent learns to make sequential decisions in an environment to maximize a cumulative reward.



ML



Task Driven
(Predict/Classify)

Data Driven
(Identify Clusters)

Learn from Mistakes
(Calculate and try)

RL vs Supervised Learning

Objective

- **Supervised Learning**: In supervised learning, the algorithm learns to *map input data to a known target or output*. The goal is to generalize from a labeled training dataset to make predictions on new, unseen data. It is typically used for tasks like classification and regression.
- **Reinforcement Learning**: In RL, the agent's objective is to learn a policy that *maximizes cumulative rewards over time* by interacting with an environment. The agent doesn't have access to a labeled dataset; it must explore the environment and learn from the consequences of its actions.

RL vs Supervised Learning

Learning Signal

- **Supervised Learning:** In supervised learning, the learning signal comes in the form of labeled examples, where the correct answers are provided for the training data. The algorithm's goal is to minimize the difference between its predictions and the actual target values.
- **Reinforcement Learning:** In RL, the learning signal is the reward signal provided by the environment. The agent receives a numerical reward (or penalty) after each action and uses this signal to adjust its policy and improve its decision-making.

RL vs Supervised Learning

Training Data

- **Supervised Learning:** Supervised learning relies on a labeled dataset containing input-output pairs. The model is trained to make predictions based on this historical data.
- **Reinforcement Learning:** RL does not have access to a labeled dataset. Instead, the agent interacts with an environment, collects experiences, and learns from the consequences of its actions.

Type of Problems

- **Supervised Learning:** Supervised learning is suitable for tasks where you have a labeled dataset and aim to make predictions or classifications based on that data.
- **Reinforcement Learning:** RL is used for sequential decision-making tasks, where the agent must interact with an environment and learn a policy for a sequence of actions over time.

RL Applications

Game Playing:

- **Video Games**: RL has been used to achieve superhuman performance in a variety of video games, such as chess, Dota 2.
- **Board Games**: RL agents have been trained to play board games like chess, shogi, and even games like poker.

Robotics:

- **Robotic Control**: RL is used to train robots to perform tasks like pick-and-place, object manipulation, and locomotion.
- **Autonomous Vehicles**: RL helps in training self-driving cars to make real-time decisions in complex traffic environments.

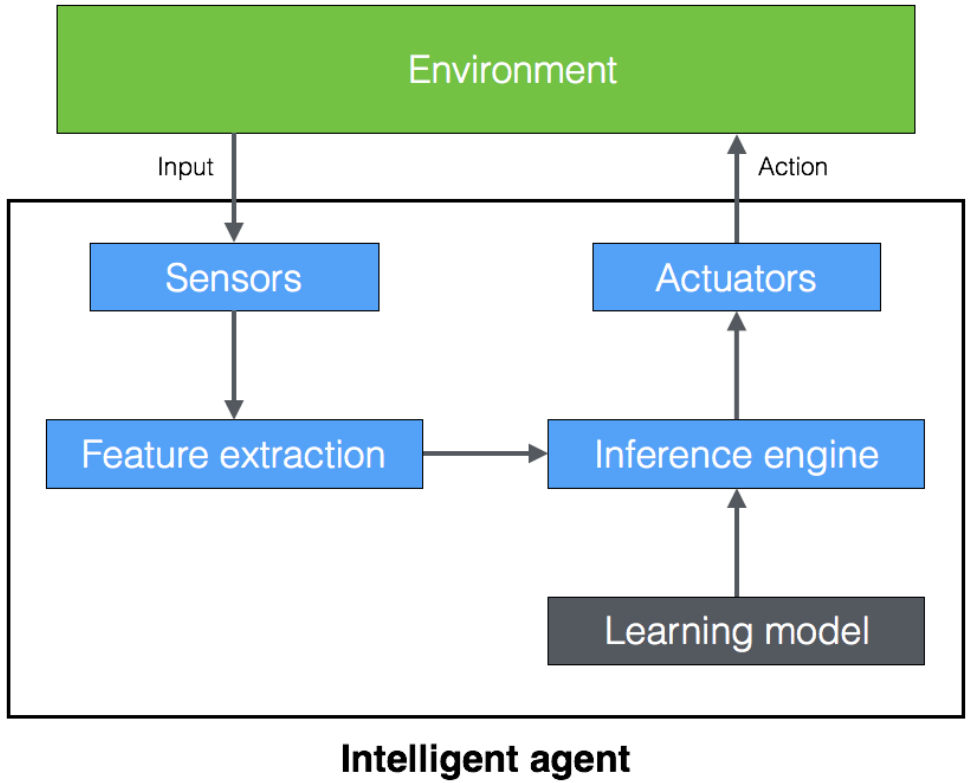
Natural Language Processing (NLP):

- **Chatbots and Virtual Assistants**: Train chatbots and virtual assistants to engage in NLP and provide relevant responses.
- **Language Translation**: It is applied in machine translation models to improve translation quality.

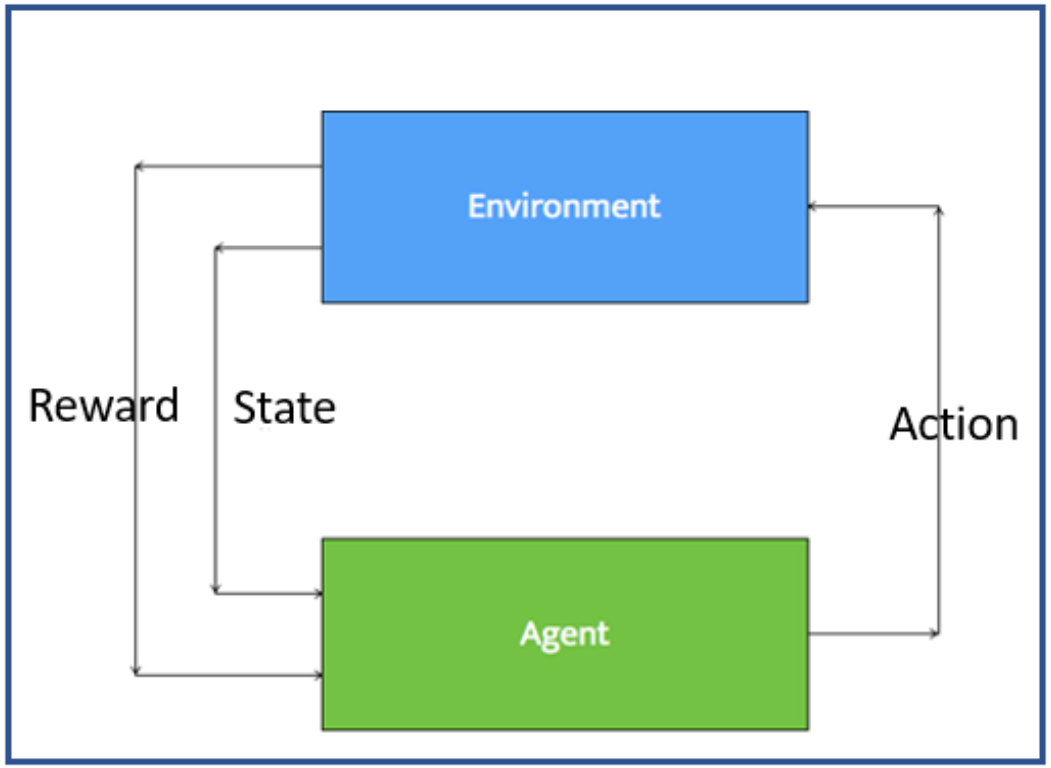
Recommendation Systems:

- **Content Recommendations**: RL algorithms can enhance content recommendations for platforms like Netflix and YouTube.
- **Adaptive Advertising**: It's used to optimize ad recommendations and placement.

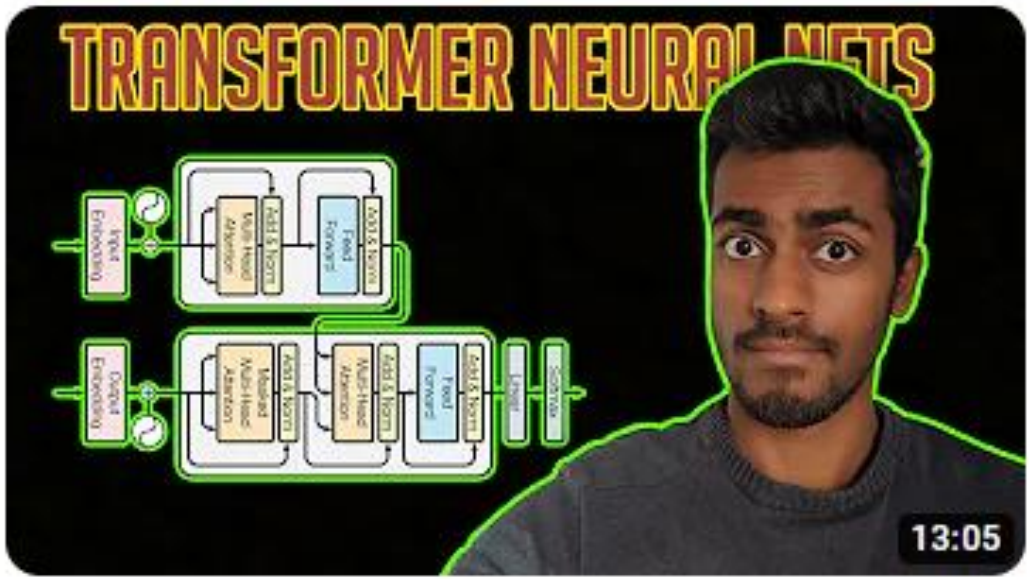
SL agent vs RL agent



Intelligent agent



Inspiration



https://www.youtube.com/watch?v=TQQlZhbC5ps

How ChatGPT works

1. **Data Collection and Preprocessing**: Large amounts of text data are collected and prepared for training.
2. **Transformer Architecture**: ChatGPT uses a transformer architecture, ideal for understanding and generating natural language.
3. **Supervised Learning**: It's trained to predict the next word in a sequence using a supervised learning approach. During training, the model is presented with input text (e.g., a series of user messages in a conversation) and learns to predict the next word or token in the sequence. It's trained to minimize the difference between its predictions and the actual text from the training data. The training process involves backpropagation and gradient descent to adjust the model's parameters.
4. **Large-Scale Training**: The model is trained on a massive dataset with billions of parameters.
5. **Fine-Tuning**: Fine-tuning customizes the model for specific tasks and domains.
6. **Inference and Deployment**: Deployed in applications like chatbots and virtual assistants to understand and generate text.
7. **Continuous Learning**: The model can be updated and improved with new data.

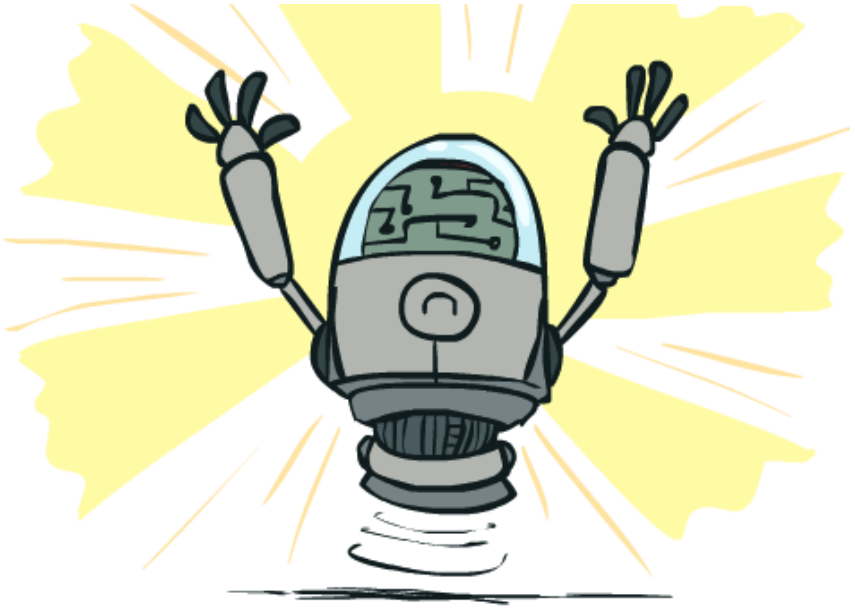
Keras in TensorFlow

Keras is an integral part of TensorFlow, and it is often referred to as "tf.keras" when used within the TensorFlow framework. Keras is an open-source deep learning framework that provides a high-level and user-friendly API for **building, training, and deploying neural networks**. Here's an explanation of Keras in TensorFlow:

- High-level API for building neural networks.
- Modular and flexible, allowing you to customize models.
- Integrated with TensorFlow for performance and scalability.
- Supports automatic differentiation for training.
- Takes advantage of GPUs and TPUs for speed.
- Widely used in research and industry.
- User-friendly with an active community and extensive resources.

Practice – Cat or Dog

Goal: 1. Successfully understand and run the coding
2. classify a random picture with established model



https://www.analyticsvidhya.com/blog/2021/06/beginner-friendly-project-cat-and-dog-classification-using-cnn/

Coding

Load the data

```
import os
import random
import numpy as np
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Dense, Flatten
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Constants
TRAIN_DIR = '/Users/ws/Desktop/train'
TRAIN_FRACTION = 0.8
RANDOM_SEED = 2018

# Function to split data into train and test sets
def make_train_and_test_sets():
    train_examples, test_examples = [], []
    shuffler = random.Random(RANDOM_SEED)

    for class_label, class_name in enumerate(['cat', 'dog']):
        class_path = os.path.join(TRAIN_DIR, class_name)
        if os.path.isdir(class_path):
            filenames = os.listdir(class_path)
            shuffler.shuffle(filenames)
            num_train = int(len(filenames) * TRAIN_FRACTION)
            full_filenames = [os.path.join(class_path, f) for f in filenames]
            examples = list(zip(full_filenames, [class_label] * len(filenames)))
            train_examples.extend(examples[:num_train])
            test_examples.extend(examples[num_train:])

    shuffler.shuffle(train_examples)
    shuffler.shuffle(test_examples)

    classes = {class_label: class_name for class_label, class_name in enumerate(['cat', 'dog'])}

    return train_examples, test_examples, classes
```

Split data into training and testing set

```
# Split the data into train and test sets and get the label classes
TRAIN_EXAMPLES, TEST_EXAMPLES, CLASSES = make_train_and_test_sets()
NUM_CLASSES = len(CLASSES)

print('\nThe dataset has %d label classes: %s' % (NUM_CLASSES, CLASSES.values()))
print('There are %d training images' % len(TRAIN_EXAMPLES))
print('There are %d test images' % len(TEST_EXAMPLES))

# Data Augmentation
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)
```

Image Preprocessing

Coding

Train the CNN model using algorithms

```
# Initializing the CNN
classifier = Sequential()

# Step 1 - Convolution
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))

# Step 2 - Max Pooling
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Step 3 - Flattening
classifier.add(Flatten())

# Step 4 - Full Connection
classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=1, activation='sigmoid'))

# Compiling the CNN
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Load your training and testing data using flow_from_directory
from keras.preprocessing.image import ImageDataGenerator

train_set = train_datagen.flow_from_directory(TRAIN_DIR, target_size=(64, 64), batch_size=32, class_mode='binary')

# Train the CNN
classifier.fit_generator(train_set, steps_per_epoch=len(train_set), epochs=25)
```

Coding

```
# Define the show_confusion_matrix function
def show_confusion_matrix(test_labels, predictions):
    """Compute confusion matrix and normalize."""
    confusion = sk_metrics.confusion_matrix(
        np.argmax(test_labels, axis=1), predictions)
    confusion_normalized = confusion.astype("float") / confusion.sum(axis=1)[:,
    axis_labels = list(CLASSES.values())
    ax = sns.heatmap(
        confusion_normalized, xticklabels=axis_labels, yticklabels=axis_labels,
        cmap='Blues', annot=True, fmt='.2f', square=True)
    plt.title("Confusion matrix")
    plt.ylabel("True label")
    plt.xlabel("Predicted label")
plt.show()
```

Evaluation

Application

```
# To predict new images (Update the image path to your image location)
imagepath = '/Users/ws/Desktop/Test.jpg'

def predict_image(imagepath, classifier):
    # Load and preprocess the image
    predict = image.load_img(imagepath, target_size=(64, 64))
    predict_modified = image.img_to_array(predict)
    predict_modified = predict_modified / 255
    predict_modified = np.expand_dims(predict_modified, axis=0)

    # Make the prediction
    result = classifier.predict(predict_modified)

    # Display the prediction and probability
    if result[0][0] >= 0.5:
        prediction = 'dog'
        probability = result[0][0]
        print("Probability = " + str(probability))
    else:
        prediction = 'cat'
        probability = 1 - result[0][0]
        print("Probability = " + str(probability))

    print("Prediction = " + prediction)

# Call the predict_image function with the image on your desktop
predict_image(imagepath, classifier)
```

Have a Break!

Case for Assignment

Introduction

Our Role: We are a software development company.

Goal: Our primary objective is to provide machine learning models designed for predicting cancer to our valued clientele.

Customers: We serve hospitals and healthcare institutions.

This business plan outlines our strategy as a software development company to **sell a machine learning model** designed for **cancer prediction**. Our primary target customers are hospitals. Accurate cancer diagnosis is pivotal for patient care, and our model provides a solution that enhances diagnostic precision and assists medical professionals in making informed decisions.

Introduction

- **Problem Definition**

Cancer is a pervasive and life-threatening disease, and early, accurate diagnosis is crucial for successful treatment. Hospitals and medical professionals often face challenges in ensuring precise and consistent diagnostic outcomes. The problem lies in the subjectivity and potential human error in the diagnostic process. Our machine learning model addresses this issue by providing an objective, data-driven approach to cancer prediction.

- **Goal Statement**

Our primary goal is to deliver a reliable, accurate, and efficient tool for cancer prediction to hospitals. We aim to establish trust as a valued partner for healthcare institutions, elevating diagnostic accuracy and patient outcomes.

- **Market Opportunity**

The global cancer diagnostics market is experiencing rapid growth, driven by the increasing incidence of cancer cases and the demand for advanced diagnostic solutions. Hospitals are a key part of this market and represent a substantial customer base.

Introduction

- **Competitive Landscape**

The cancer diagnostics market is highly competitive, featuring established players offering various diagnostic tools and services.

Our machine learning model, developed by a software development company, differentiates itself by:

1. Delivering superior accuracy and objectivity.
2. Continuous model improvement with the latest research and data.
3. Offering customization and integration services tailored to hospital needs.
4. Ensuring data security and compliance with healthcare regulations.

Introduction

- **Benefits**

1. Enhanced Accuracy: Ensures superior diagnostic accuracy compared to traditional methods, reducing misdiagnoses.
2. Efficiency: Speeds up the diagnostic process, enabling faster treatment decisions and reducing healthcare costs.
3. Objective Diagnosis: Mitigates subjectivity and human error, providing consistent and data-driven predictions.
4. Customization: Allows hospitals to adapt the model to their unique needs and workflows.
5. Continuous Improvement: Stays current with the latest research and data, maintaining high accuracy.

Introduction

- **Other aspects that you want to address, but keep it simple**

Marketing and sales

Risks

Partners

Team introduction

Etc.

AI Model – Data description

1. Number of Instances: 699 (as of 15 July 1992)

2. Number of Attributes: 10 plus the class attribute

3. Attribute Information: See the table

4. Missing Attribute Values: 16

There are 16 instances in Groups 1 to 6 that contain a single missing (i.e., unavailable) attribute value, now denoted by "?".

5. Target Value:

•Benign: 458 (65.5%)

•Malignant: 241 (34.5%)

This dataset comprises 699 instances, each described by 10 attributes. The class attribute, which indicates whether a tumor is benign (2) or malignant (4), has been moved to the last column for classification purposes. It's important to note that there are 16 instances with missing attribute values, denoted by "?". The class distribution in the dataset shows that 65.5% of the instances are benign, while 34.5% are malignant, which is crucial for understanding the prevalence of cancer types in this dataset.

#	Attribute	Domain
1	Sample code number	ID number
2	Clump Thickness	1 - 10
3	Uniformity of Cell Size	1 - 10
4	Uniformity of Cell Shape	1 - 10
5	Marginal Adhesion	1 - 10
6	Single Epithelial Cell Size	1 - 10
7	Bare Nuclei	1 - 10
8	Bland Chromatin	1 - 10
9	Normal Nucleoli	1 - 10
10	Mitoses	1 - 10
11	Class	(2 for benign, 4 for malignant)

AI Model – Data preprocessing

Selected Features: 'Sample code number', 'Clump Thickness', 'Uniformity of Cell Size', 'Uniformity of Cell Shape', 'Marginal Adhesion', 'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland Chromatin', 'Normal Nucleoli', 'Mitoses'

Target Value: "class", 2: Indicates a benign tumor, suggesting it is not cancerous. 4: Indicates a malignant tumor, suggesting it is cancerous.

Missing Value:

Data Imputation _ Missing values can be imputed using various techniques, such as mean, median, or mode imputation, based on the distribution of the non-missing values in the dataset.

Data Removal _ Alternatively, instances with missing values can be removed from the dataset. However, this approach should be used cautiously, as it may result in the loss of valuable data.

Advanced Imputation _ More advanced imputation methods like predictive modeling can be used to estimate missing values based on the relationships within the dataset.

AI Model – algorithms

Logistic Regression

Logistic Regression is a fundamental statistical and machine learning algorithm used for binary classification tasks, which involve predicting one of two possible outcomes, typically coded as 0 and 1. In our model, it's used to classify tumors as either benign (2) or malignant (4).

Key Characteristics and Components

1.Linear Model: Logistic Regression is a linear model, meaning it models the relationship between the features and the likelihood of a particular outcome through a linear combination of the features. This linear combination is then transformed using a logistic (sigmoid) function to produce a probability value.

2.Sigmoid Function: The logistic (sigmoid) function is a critical component of this model. It takes the linear combination of features as input and maps it to a value between 0 and 1. This value represents the probability of the target being in one of the two classes.

3.Coefficient and Intercept: Logistic Regression involves estimating coefficients (weights) for each feature, which represent the influence of that feature on the prediction. Additionally, there's an intercept term. Together, the coefficients and intercept define the linear boundary that separates the two classes.

AI Model – algorithms

1.Training: During the training phase, the model learns the optimal coefficients and intercept that best fit the training data. It does so by minimizing a loss function (e.g., the log-likelihood loss) that quantifies the difference between predicted probabilities and actual class labels.

2.Prediction: After training, the model can predict the probability of an instance belonging to the positive class (in our case, malignant) based on the values of the input features. If this probability exceeds a certain threshold (usually 0.5), the model classifies the instance as belonging to the positive class; otherwise, it's classified as the negative class.

AI Model – algorithms

Argue why this algorithm

Eg. Logistic Regression is well-suited for cancer prediction tasks like the one we've described, where the goal is to classify tumors as benign or malignant based on a set of features. Its simplicity and interpretability make it a valuable tool in the medical field, where transparency and trust in the model's decisions are of utmost importance. The model estimates the probability of malignancy for a given tumor, which can aid healthcare professionals in making informed decisions about patient care.

State the advantages and limitations of this algorithm

Advantage 1 _ Simplicity: Logistic Regression is easy to understand and interpret, crucial in medical contexts.

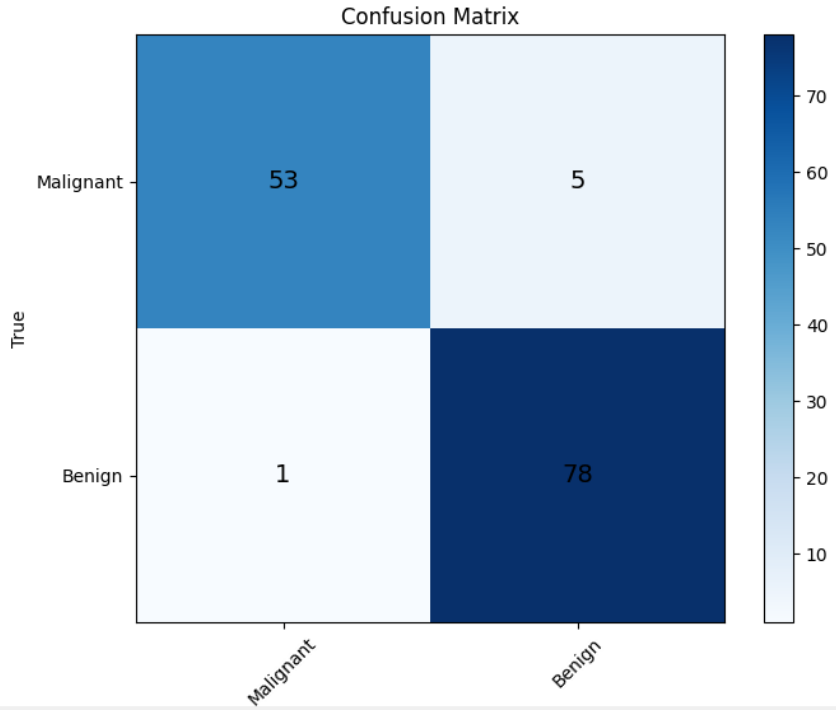
Advantage 2 _ Efficiency: It's computationally efficient, suitable for large feature sets.

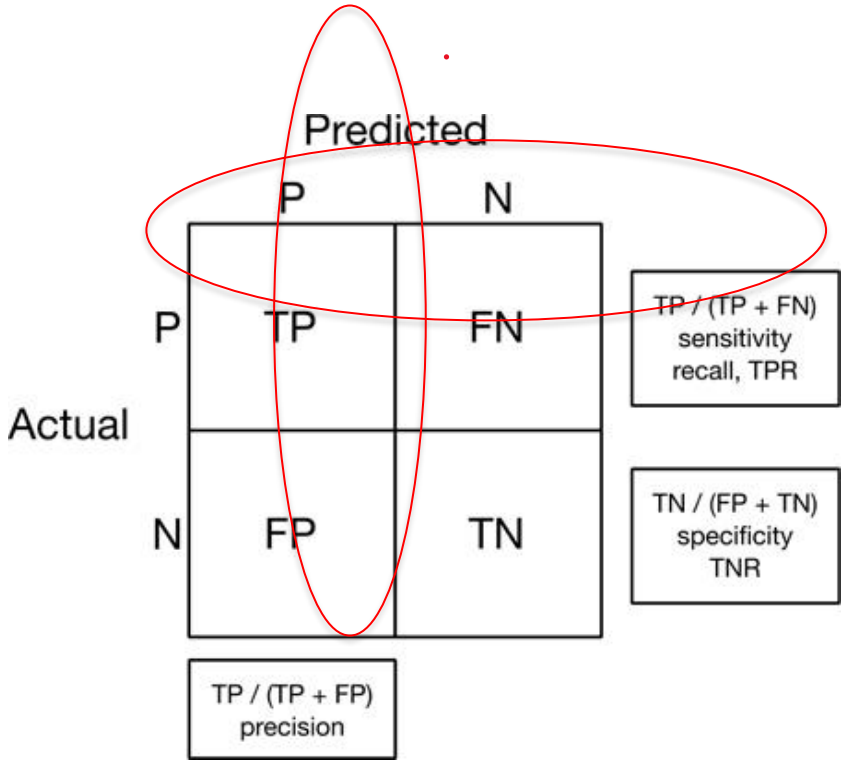
Limitation 1_ Assumes Linearity: It assumes linear relationships, which may limit accuracy in nonlinear scenarios.

Limitation 2_ Binary Classification: Inherently designed for two-class problems, needing extensions for multiclass tasks.

AI Model – Evaluation

Why this model?






```
Recall (True Positive Rate) for Malignant Class: 0.91
```

Discussion

State the limitations of your model

State your experience while developing the model

How the model might be improved (eg. the accuracy)

Etc.

Conclusion

As a pioneering software development company, our unwavering commitment is to provide hospitals with an innovative machine learning model for cancer prediction, underpinned by logistic regression—a reliable and proven algorithm. This model holds the potential to significantly impact patient care and redefine medical practice by improving the efficiency of disease diagnosis.

We must emphasize that while our model serves as a **powerful diagnostic tool**, it is essential for healthcare professionals to recognize that they are the ultimate decision-makers. Doctors, with their expertise and clinical judgment, remain solely responsible for their patients' well-being and the diagnostic process.

By delivering precise and timely information to healthcare professionals, our aim is to empower them with a data-driven, objective tool that can expedite the diagnostic process. We envision this collaboration between technology and healthcare expertise as an approach to enhance patient outcomes.

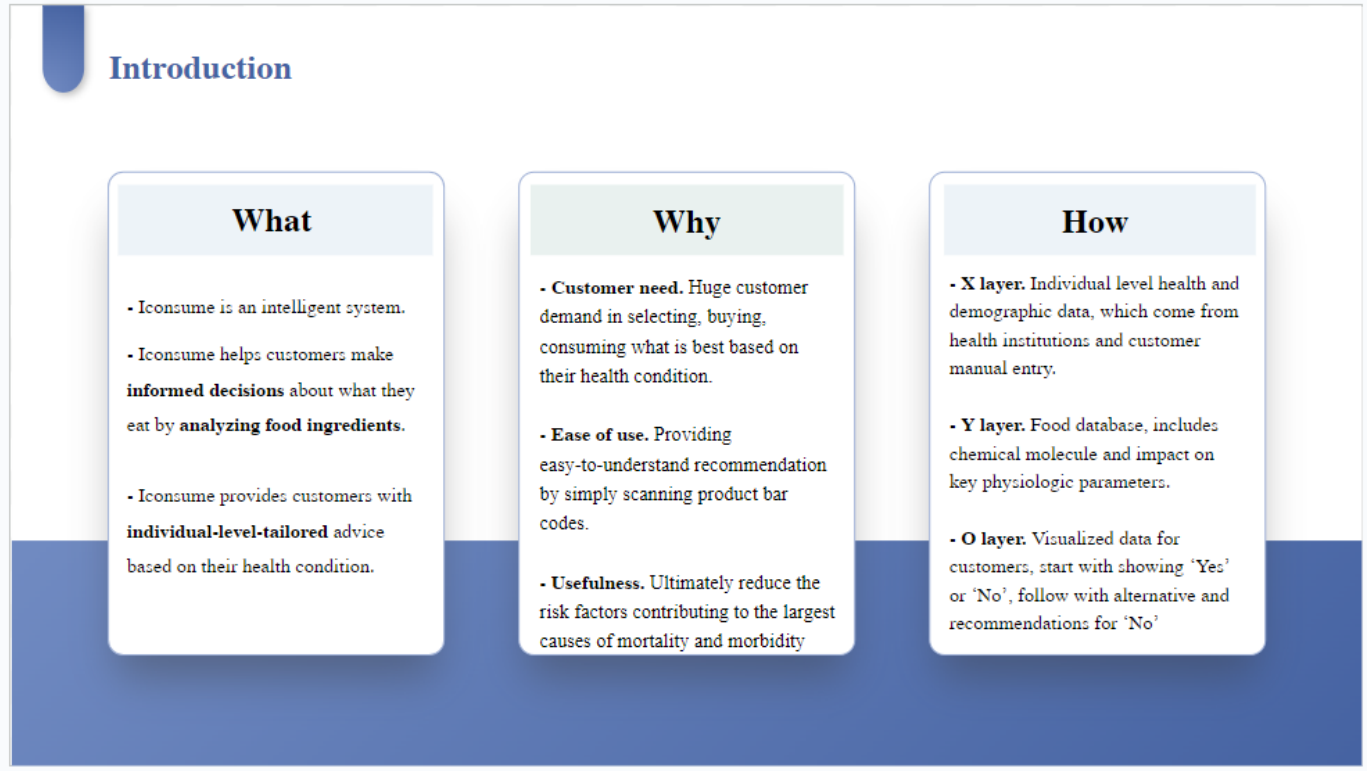
Through rigorous evaluation, including the use of the confusion matrix and other robust assessment methods, we ensure the accuracy, reliability, and effectiveness of our model. It is our belief that this transformative tool will facilitate and support doctors in their vital role, ultimately leading to improved patient outcomes and a brighter future in the fight against cancer.

Appendix – coding & data

```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix

# Load the dataset
column_names = ['Sample code number', 'Clump Thickness', 'Uniformity of Cell Size', 'Uniformity of Cell Shape',
                'Marginal Adhesion', 'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland Chromatin',
                'Normal Nucleoli', 'Mitoses', 'Class']

data = pd.read_csv("breast-cancer-wisconsin.data", names=column_names)

# Data processing
data.replace("?", np.nan, inplace=True)
data.dropna(inplace=True)

# Split data into features (X) and target (y)
X = data.drop('Class', axis=1)
y = data['Class']

# Split data into train and test sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize the data using StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Create and train a logistic regression model
logreg_model = LogisticRegression()
logreg_model.fit(X_train_scaled, y_train)

# Make predictions on the test set
y_pred = logreg_model.predict(X_test_scaled)

# Calculate the confusion matrix
confusion = confusion_matrix(y_test, y_pred)

# Extract values from the confusion matrix
true_negative, false_positive, false_negative, true_positive = confusion.ravel()

# Calculate recall (true positive rate)
recall = true_positive / (true_positive + false_negative)
print("Recall (True Positive Rate):", recall)
```

```
1000025,5,1,1,1,2,1,3,1,1,2
1002945,5,4,4,5,7,10,3,2,1,2
1015425,3,1,1,1,2,2,3,1,1,2
1016277,6,8,8,1,3,4,3,7,1,2
1017023,4,1,1,3,2,1,3,1,1,2
1017122,8,10,10,8,7,10,9,7,1,4
1018099,1,1,1,1,2,10,3,1,1,2
1018561,2,1,2,1,2,1,3,1,1,2
1033078,2,1,1,1,2,1,1,1,5,2
1033078,4,2,1,1,2,1,2,1,1,2
1035283,1,1,1,1,1,1,3,1,1,2
1036172,2,1,1,1,2,1,2,1,1,2
1041801,5,3,3,3,2,3,4,4,1,4
1043999,1,1,1,1,2,3,3,1,1,2
1044572,8,7,5,10,7,9,5,5,4,4
1047630,7,4,6,4,6,1,4,3,1,4
1048672,4,1,1,1,2,1,2,1,1,2
1049815,4,1,1,1,2,1,3,1,1,2
1050670,10,7,7,6,4,10,4,1,2,4
1050718,6,1,1,1,2,1,3,1,1,2
1054590,7,3,2,10,5,10,5,4,4,4
1054593,10,5,5,3,6,7,7,10,1,4
1056784,3,1,1,1,2,1,2,1,1,2
1057013,8,4,5,1,2,?,7,3,1,4
1059552,1,1,1,1,2,1,3,1,1,2
1065726,5,2,3,4,2,7,3,6,1,4
1066373,3,2,1,1,1,1,2,1,1,2
1066979,5,1,1,1,2,1,2,1,1,2
1067444,2,1,1,1,2,1,2,1,1,2
1070935,1,1,3,1,2,1,1,1,1,2
1070935,3,1,1,1,1,1,2,1,1,2
1071760,2,1,1,1,2,1,3,1,1,2
```

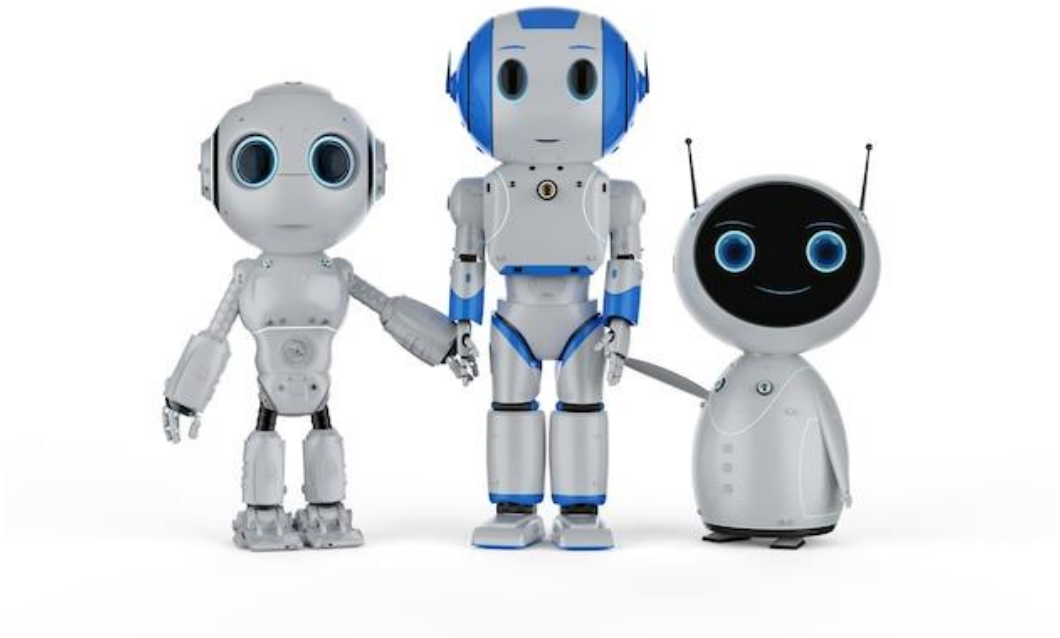
Presentation Example



https://docs.google.com/presentation/d/1apGrruOOSUAUrAjkWuUG4qt59G1_Ywaz/edit#slide=id.p4

Have a Break!

Group Up!



Work on your assignment!

Guidance

https://www.youtube.com/watch?v=dGtDTjYs3xc&list=PLeo1K3hjS3uu7CxAacxVndI4bE_o3BDtO&index=50

Potato Disease _ Deep learning project end to end

THANKS