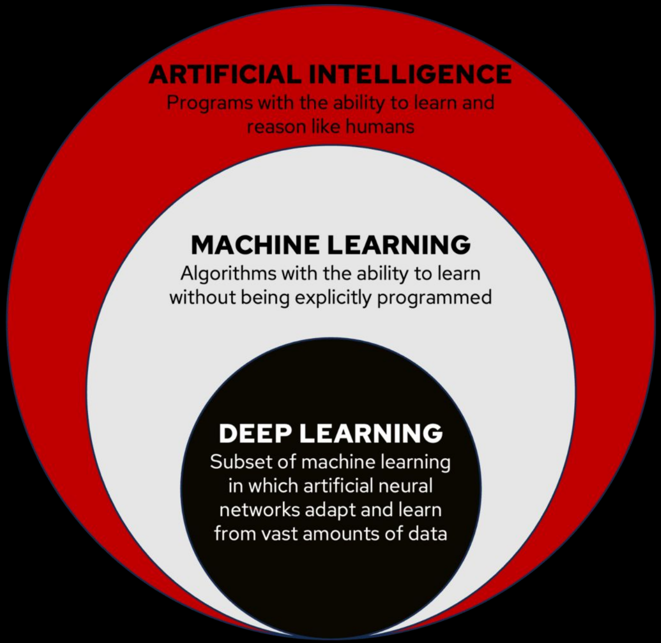
Machine Learning Workshop

9/30/2024

By Grant Congdon

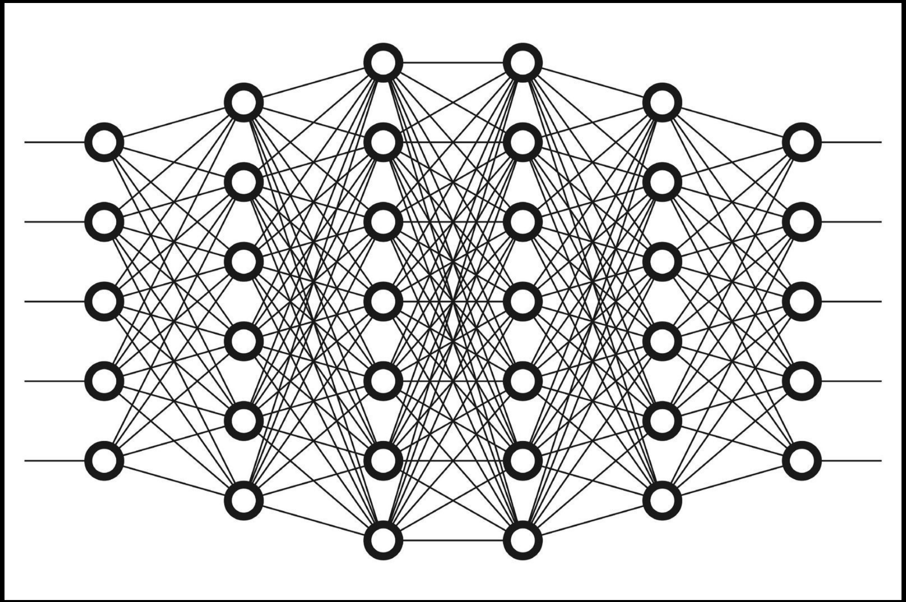
Subdivisions of Artificial Intelligence

- Artificial intelligence - the ability of software to mimic human intelligence
- Most research dedicated to deep learning
  - Models how biological intelligence is structured
  - Removes humans from the learning process (after labeling)



MIND

Deep Learning



Deep Learning

- Models biological intelligence
  - Neurons in organic brains are highly interconnected
  - Action potentials travel through networks of neurons producing thoughts
  - Deep learning mimics this behavior with math
- Neural networks draw connections between data points to complete tasks
- Providing a diverse, high quality dataset is important to creating a robust machine learning model

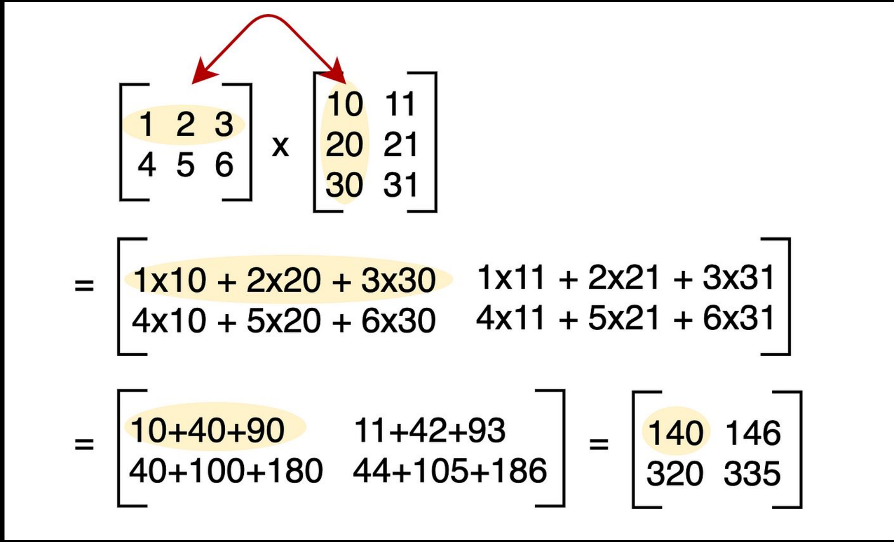
General Neural Network (NN)

- Foundation of neural networks is linear algebra and activation functions
- Incoming data is converted into a tensor (an n-dimensional matrix)
- Data is then normalized, typically between 0 and 1
- The normalized data tensor is multiplied by weights (tuned matrices)
- Final layer maps the data into the number of classification classes
- Nonlinear activation functions add nonlinearity between linear layers
- Different NNs have slight variations in these steps

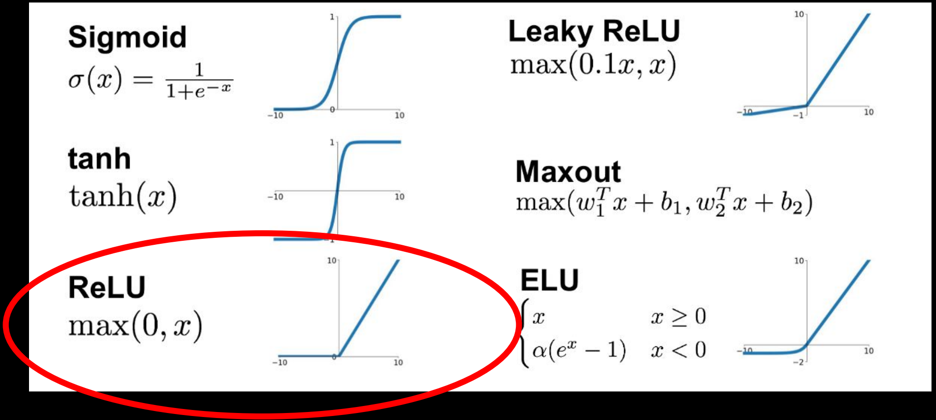
Evaluating Neural Networks

- The NN model needs to know how it's doing as its predicting classes
  - This can't use the data the machine learning model is using for training
- Model validity needs to be tested with an unseen dataset after training
- Therefore, input dataset is typically split into 3 categories

Background Concepts



Matrix Multiplication



Activation Functions

Goals

- Given the vitals of a patient, predict whether they will experience a heart attack
- Compare accuracy of Pearson's R coefficient to deep learning model
- Understand building blocks of more complex deep learning models

Learning as a Machine Step 1: Normalize Data

Raw Data

Age	Sex	Max heart rate	Cholesterol
63	1	145	233
37	1	130	250
56	0	140	294

Normalized Data

Age	Sex	Max heart rate	Cholesterol
1	1	1	0
0	1	0	0.28
0.73	0	0.67	1

Normalization equation: $$x_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

Learning as a Machine Step 2: Split Data

Normalized Data

Age	Sex	Max heart rate	Cholesterol
1	1	1	0
0	1	0	0.28
0.73	0	0.67	1

Train Data

Age	Sex	Max heart rate	Cholesterol
1	1	1	0
0	1	0	0.28

Learning as a Machine Step 3: Draw Connections

Train Data

Age	Sex	Max heart rate	Cholesterol
1	1	1	0
0	1	0	0.28

X

Black Box

Random numbers $\xrightarrow{\text{ReLU}}$ Intermediate Form

Learning as a Machine Step 4: Still Drawing

Intermediate Form → Black Box → Black Box → Final Form

The size and number of 'black boxes' can be tuned to optimize the models performance

MIND

Learning as a Machine Step 5: Classify

Final Form
(0-1)

0

1

The predicted class is whichever number the final form is closest to

Learning as a Machine Step 6: Check Your Work

- The model now compares its answer with the actual classification
- The difference is calculated using a loss function
    - For two classes, binary cross entropy loss is a popular choice
- Using the loss, the model then updates the ‘black boxes’ to make them more accurate
- At the end of each training cycle, between epochs, the validation set is used to indicate how well training is going

Learning as a Machine Step 7: Run it back

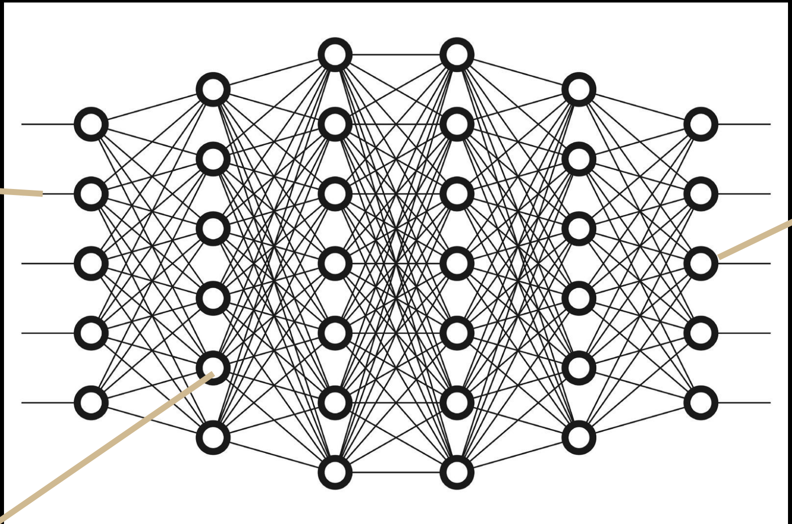
- The training/validation cycle repeats until the model converges, like reaching the minimum value on a parabola
    - Adjusting the weights/'black boxes' after convergence will only make the loss higher
- After the model has run through all epochs, the test set is used to measure how well the model works on unseen data
- The accuracy of the model at this point is a good indicator of the model's performance

MIND

Implementation

Implementation: Designing the NN

Input Nodes:
1 node for each variable connects to each node of the hidden layer (14 x 64)



Output Nodes:
1 output node indicating whether heart attack risk or not (64 x 1)

Hidden Layers:
Perform linear algebra on inputted data using ReLU function (64 x 64)

Implementation: Training the Model

- **Criterion** - Calculates the difference between actual and expected classes
- **Optimizer** - Adjusts parameters for model according to criterion
- **Learning Rate** - How quickly the model's parameters are adjusted
- **Epochs** - Number of training repetitions