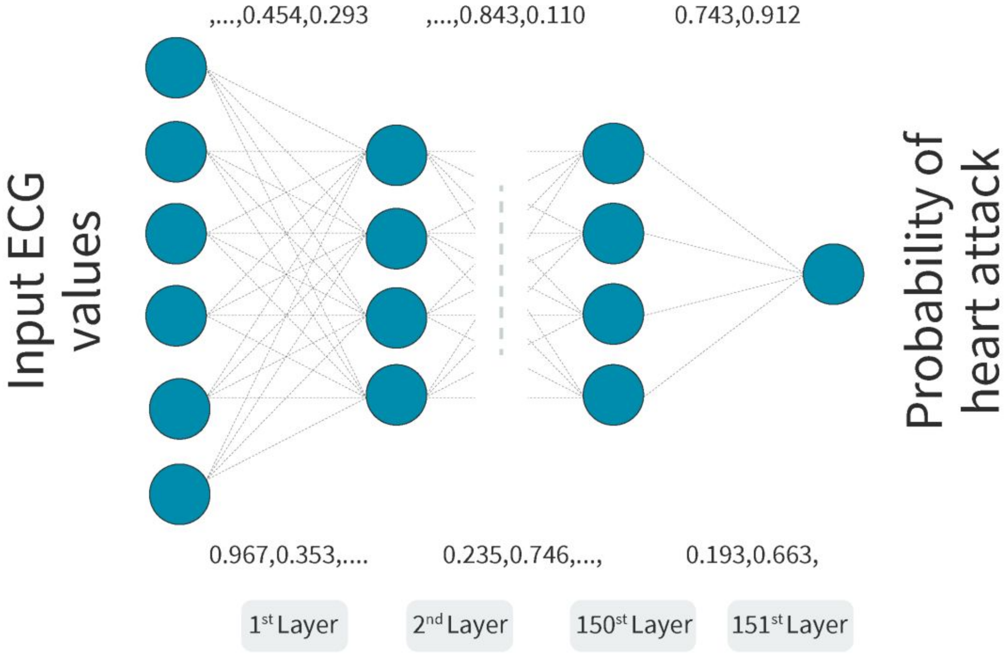
Lecture 4
Fundamentals of deep learning and neural networks

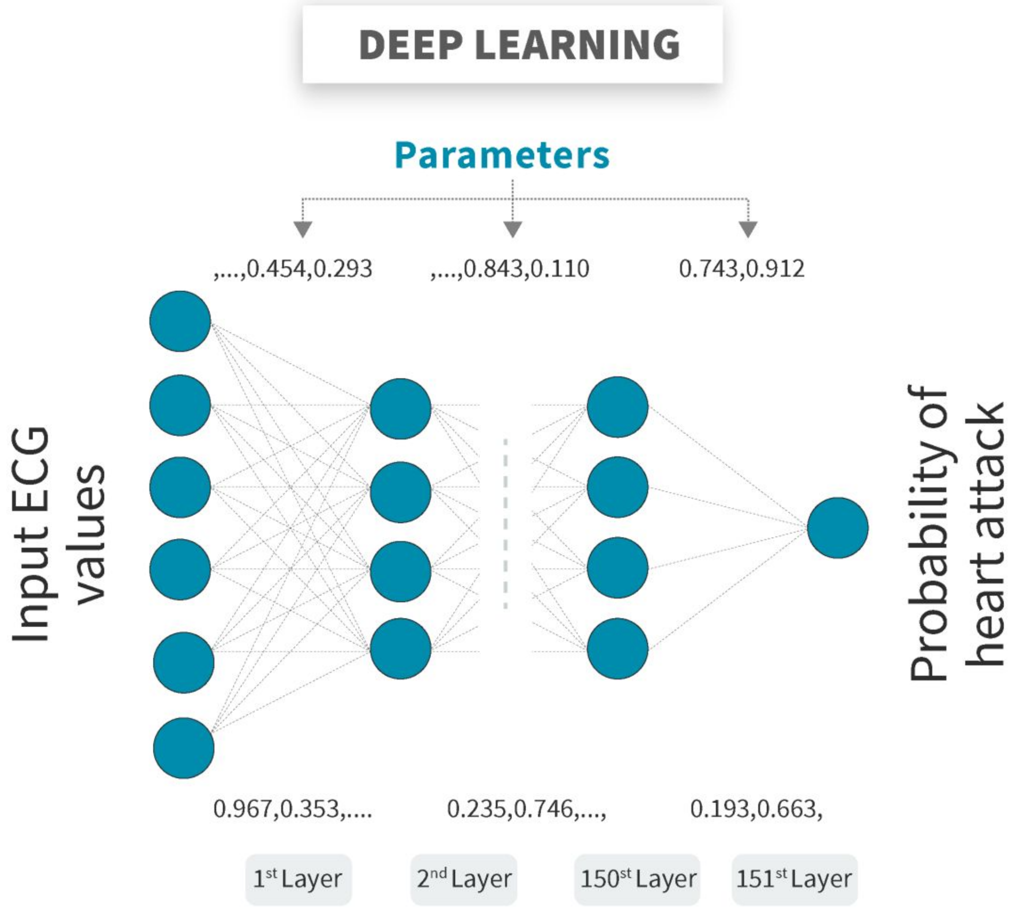
Serena Yeung
BIODS 388

Deep learning: Machine learning models based on "deep" neural networks comprising millions (sometimes billions) of parameters organized into hierarchical layers.

Features are multiplied and added together repeatedly, with the outputs from one layer of parameters being fed into the next layer -- before a prediction is made.

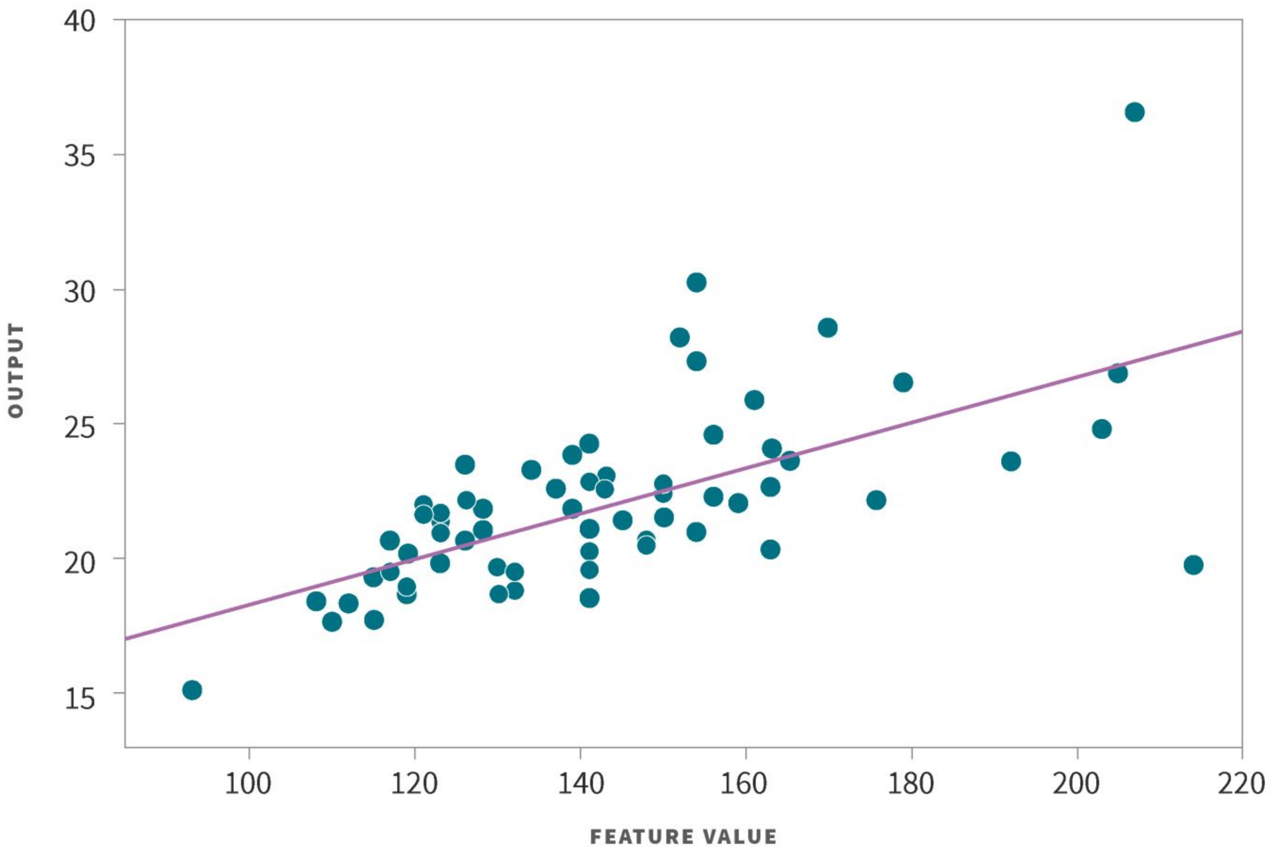
DEEP LEARNING




DEEP LEARNING
Parameters
,...,0.454,0.293
,...,0.843,0.110
0.743,0.912
Input ECG values
Probability of heart attack
0.967,0.353,....
0.235,0.746,...,
0.193,0.663,
1st Layer
2nd Layer
150st Layer
151st Layer

Contrast with linear regression:

$$y = \overbrace{\overset{\text{WEIGHT}}{m}x + \overset{\text{BIAS}}{b}}^{\text{PARAMETERS}}$$



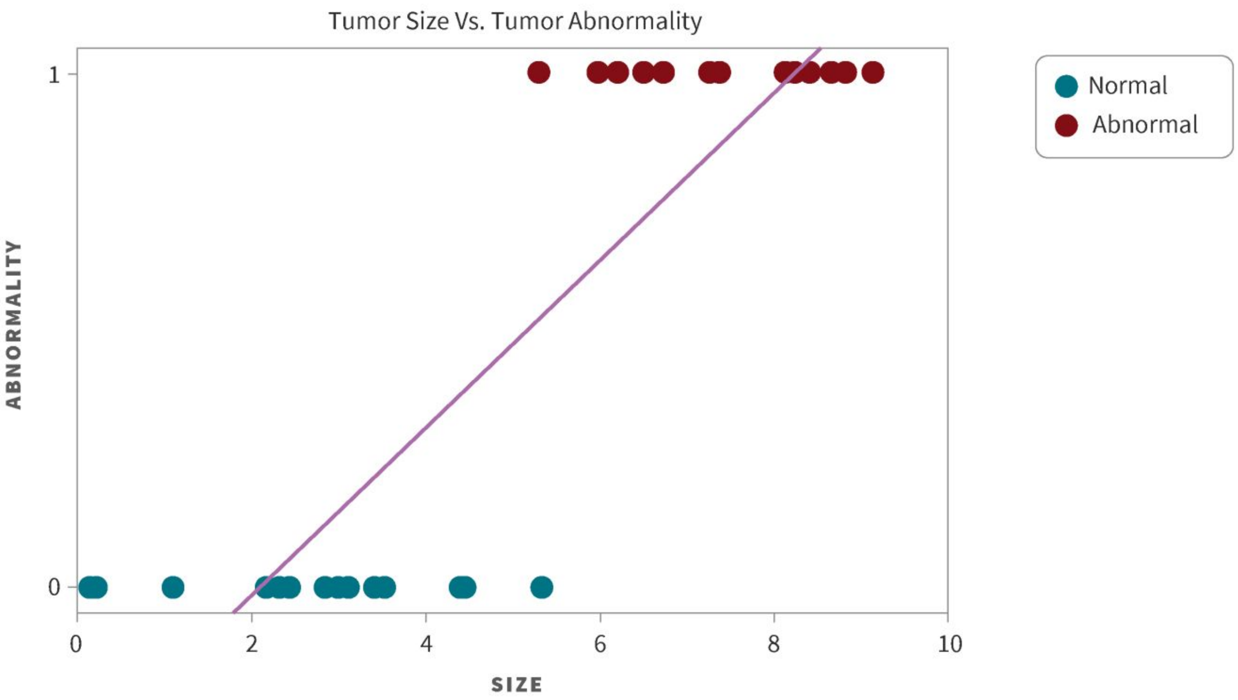
40
35
30
25
20
15
OUTPUT
100
120
140
160
180
200
220
FEATURE VALUE

Agenda for today

- More on the structure of neural network models
- Machine learning training loop and concept of *loss*, in the context of neural networks
- Minimizing the loss for complex neural networks: gradient descent and backpropagation

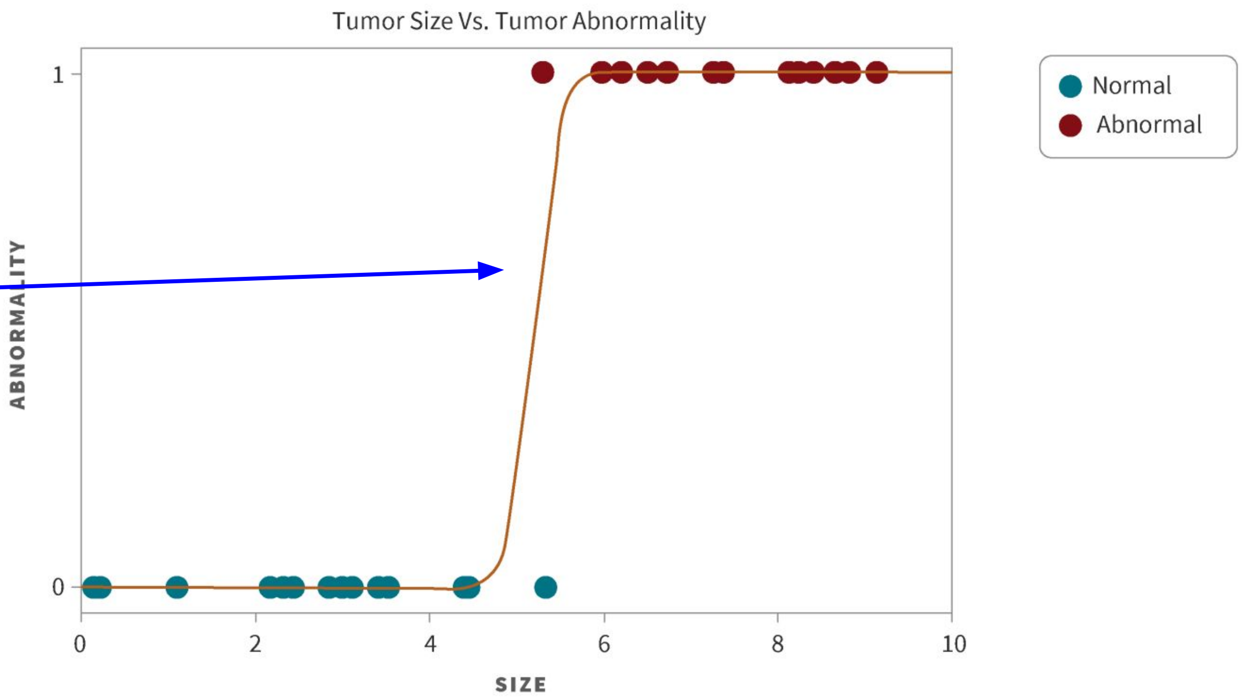
Let's start by considering again *logistic* regression, for binary classification

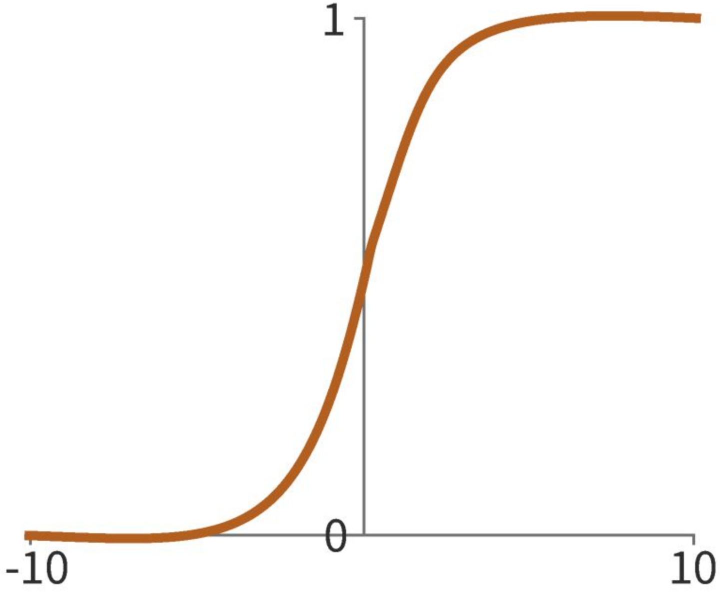
EXAMPLE: *Using tumor size to classify as normal or abnormal*



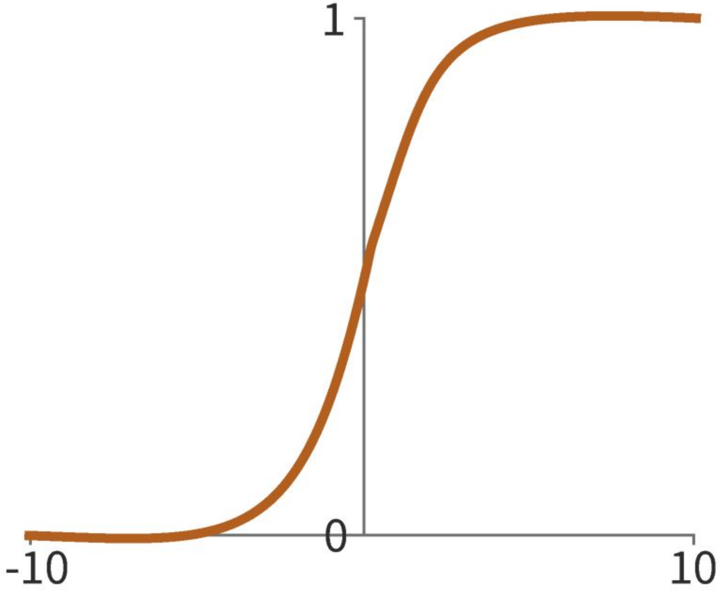
EXAMPLE: *Using tumor size to classify as normal or abnormal*

Nonlinear squashing to (0,1) with sigmoid nonlinearity

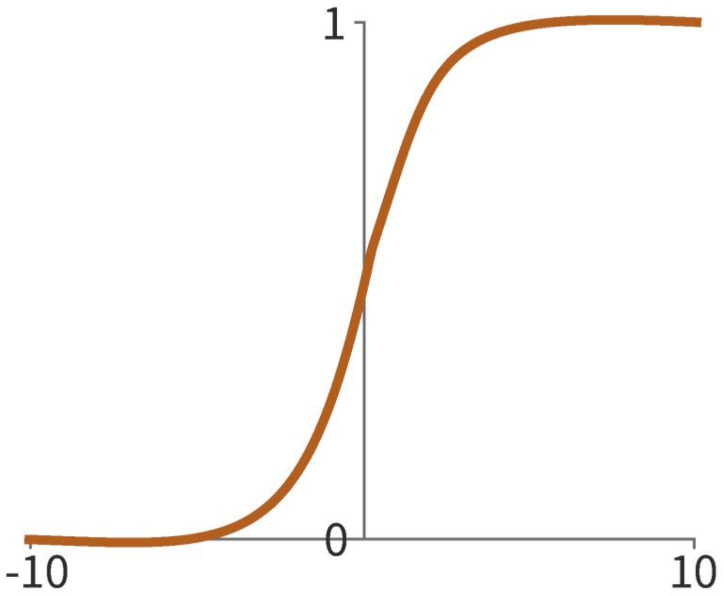




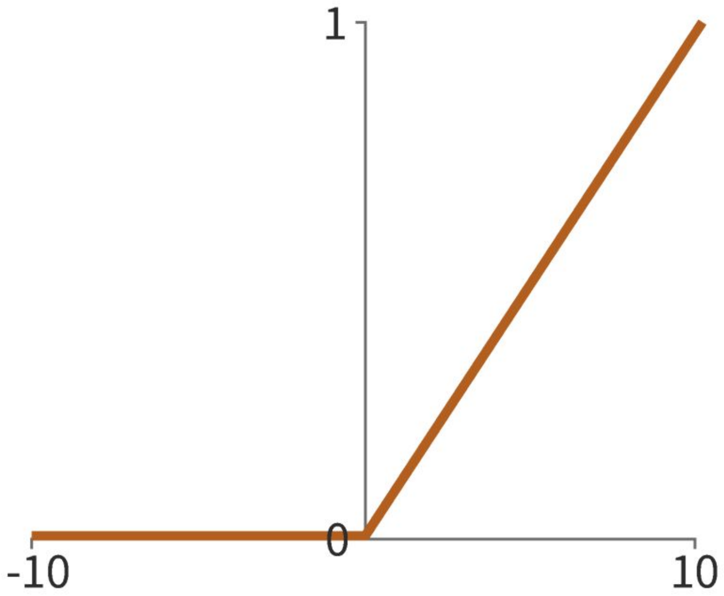
Sigmoid Function



Sigmoid Function
(an activation function)

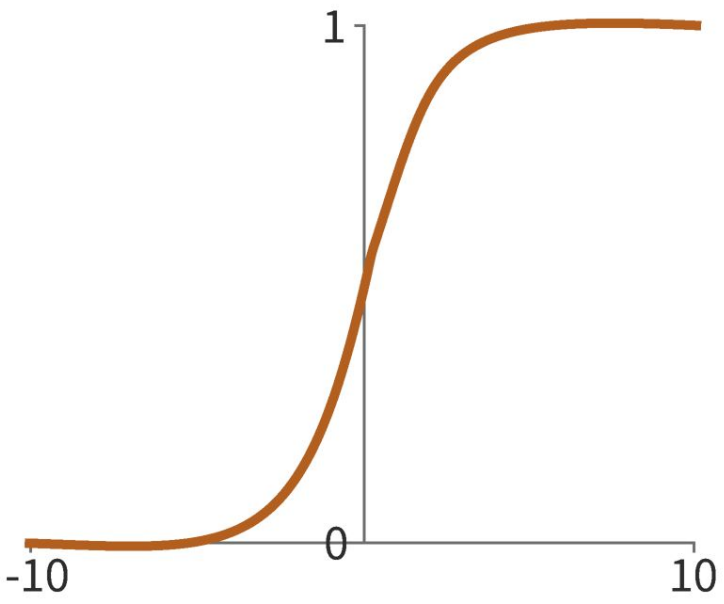


Sigmoid Function
(an activation function)

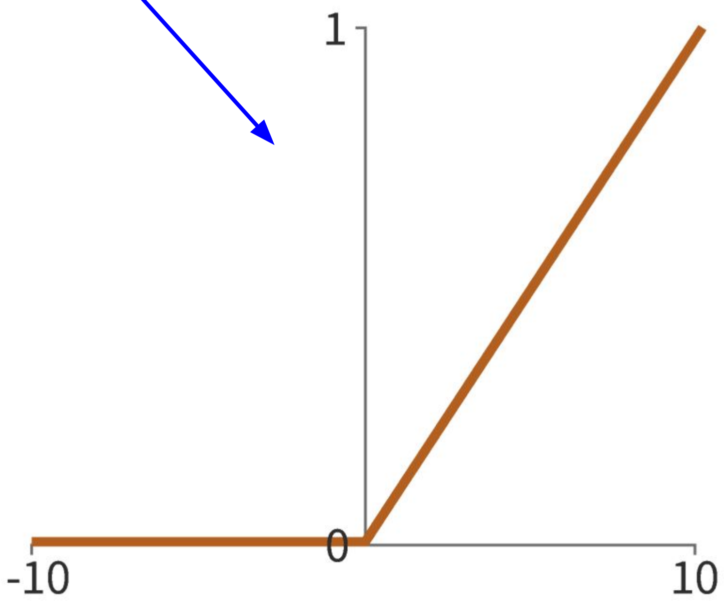


ReLU Function
(another activation function)

Also commonly used in
modern neural networks!

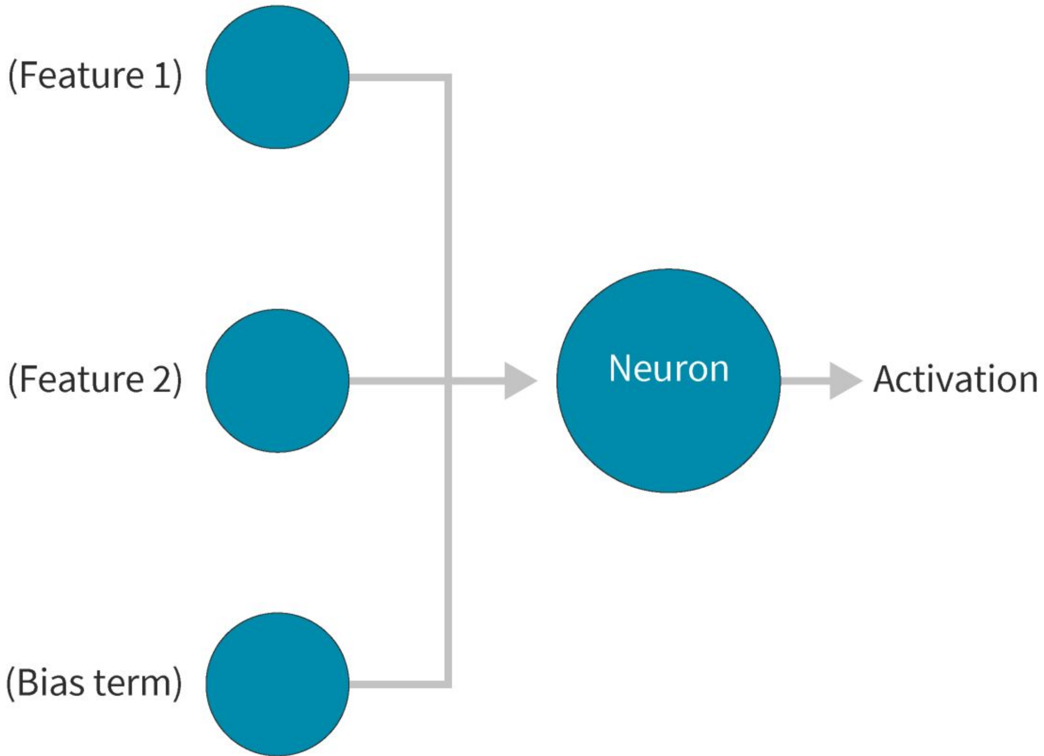


Sigmoid Function
(an activation function)

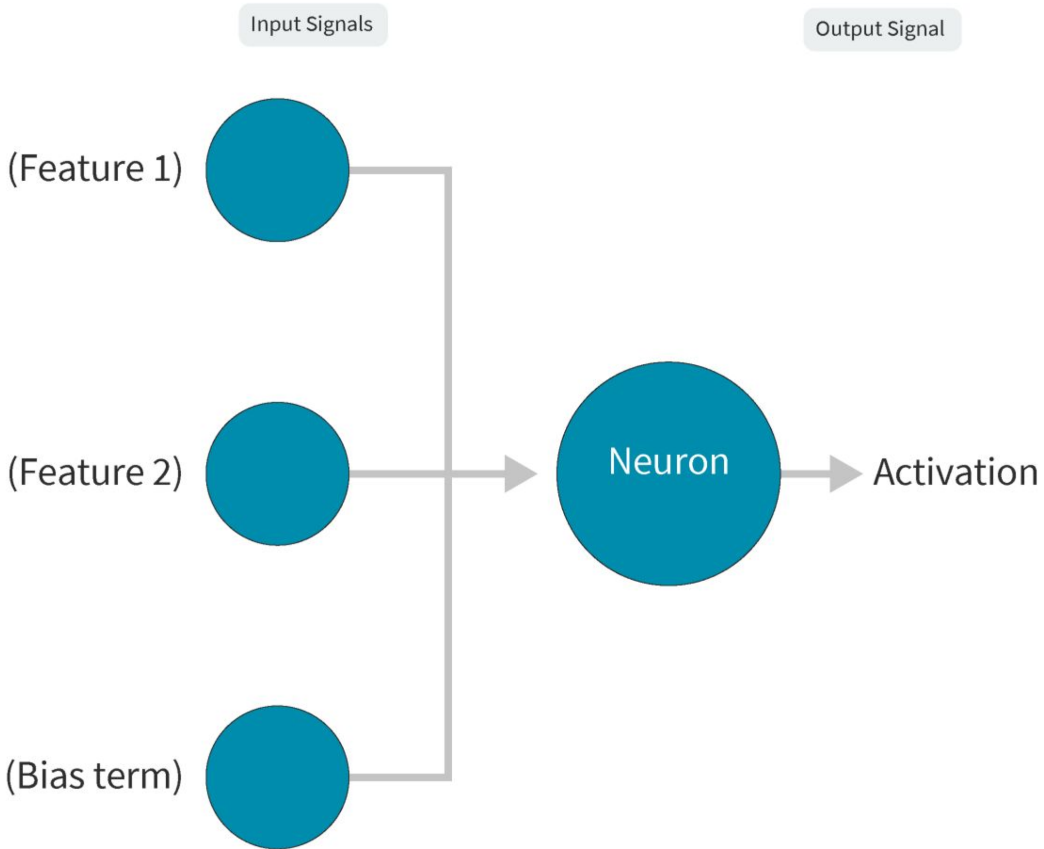


ReLU Function
(another activation function)

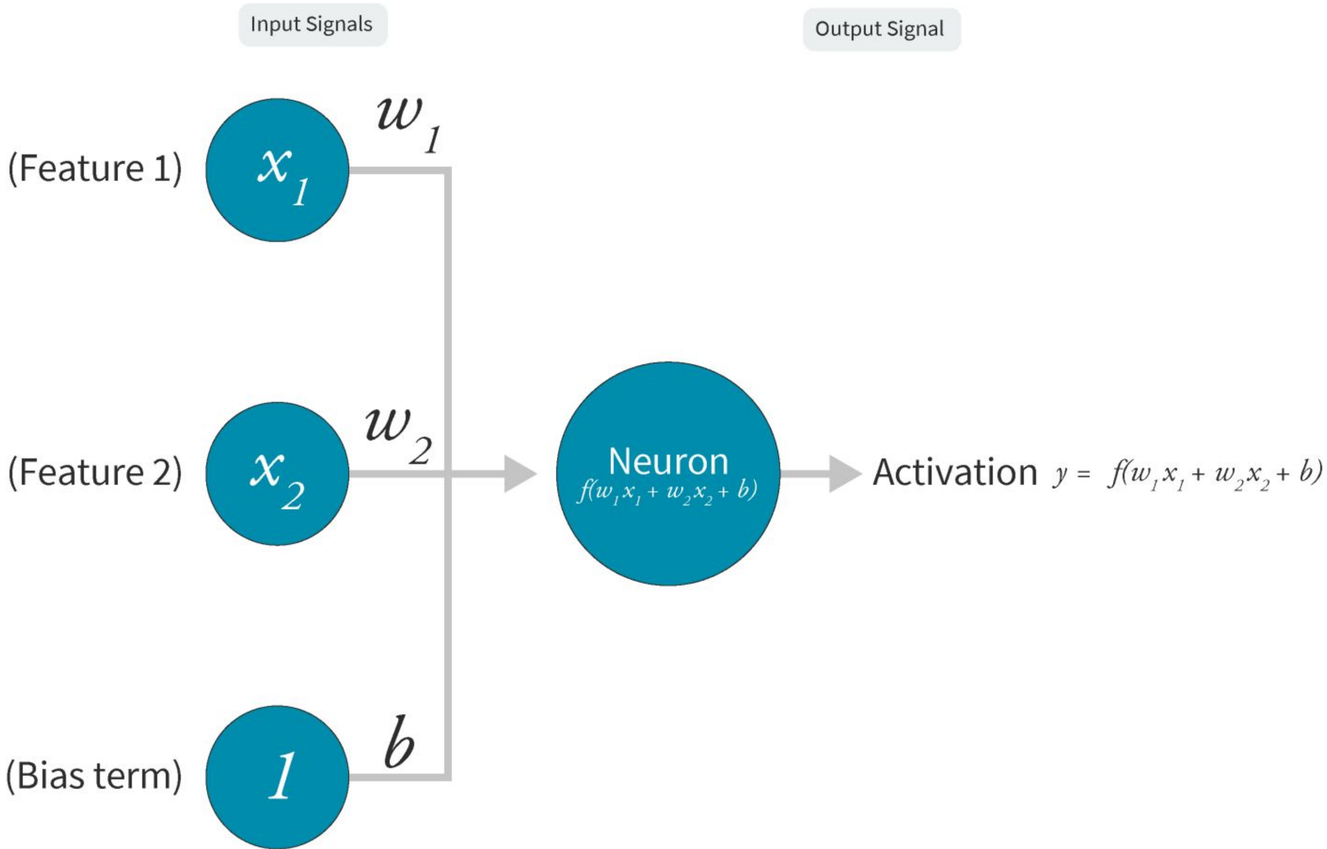
The logistic regression with sigmoid that we just saw can be considered as a single “neuron” model:



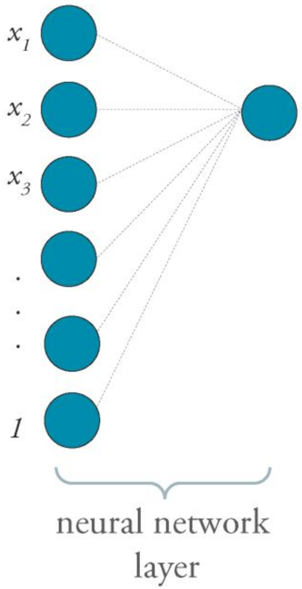
The logistic regression with sigmoid that we just saw can be considered as a single “neuron” model:



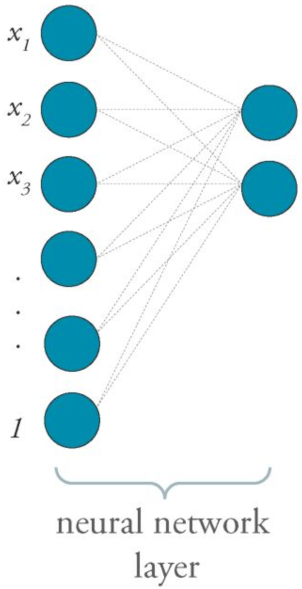
The logistic regression with sigmoid that we just saw can be considered as a single “neuron” model:



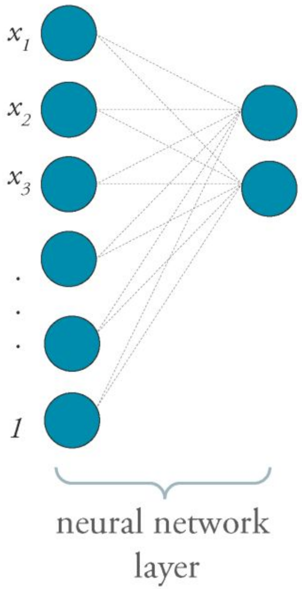
A layer of a neural networks consists of a set of neurons that each take the same input!



A layer of a neural networks consists of a set of neurons that each take the same input!

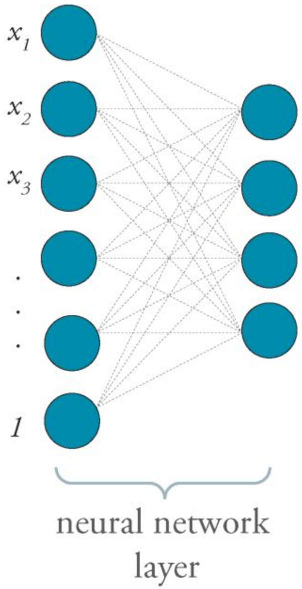


A layer of a neural networks consists of a set of neurons that each take the same input!

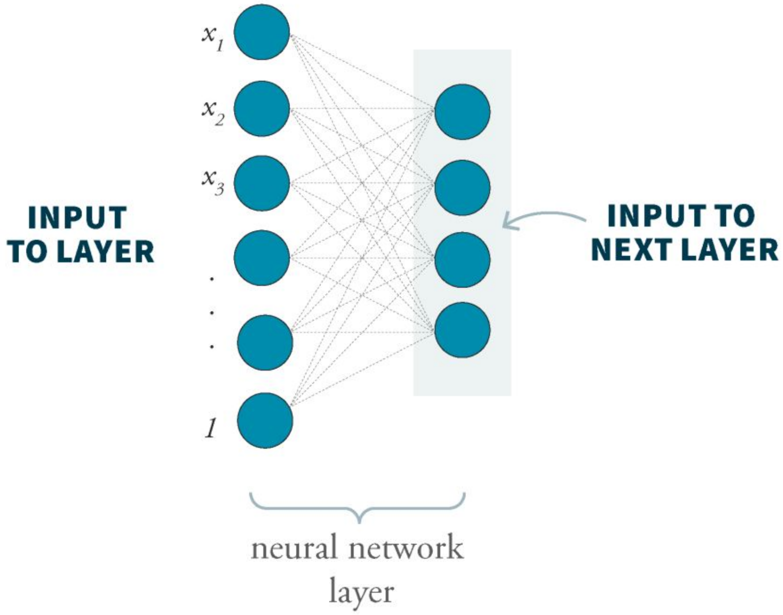


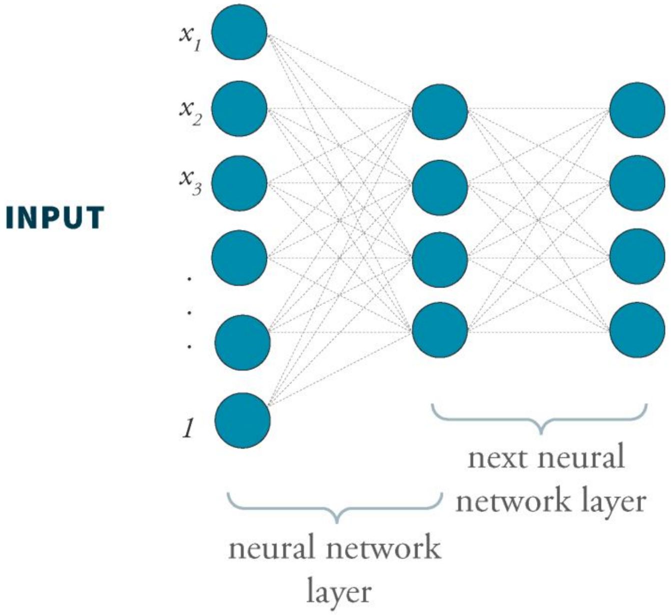
Note: each neuron will have its own set of parameters that it learns, which will produce different outputs

A layer of a neural networks consists of a set of neurons that each take the same input!



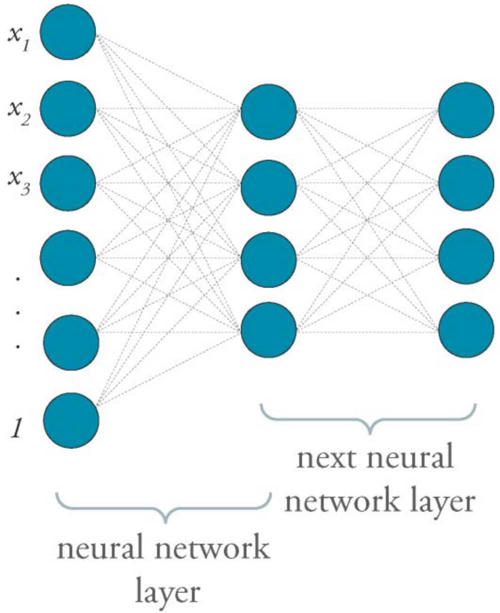
Concatenate the multiple outputs from a layer of a neural network to be the input to the next layer

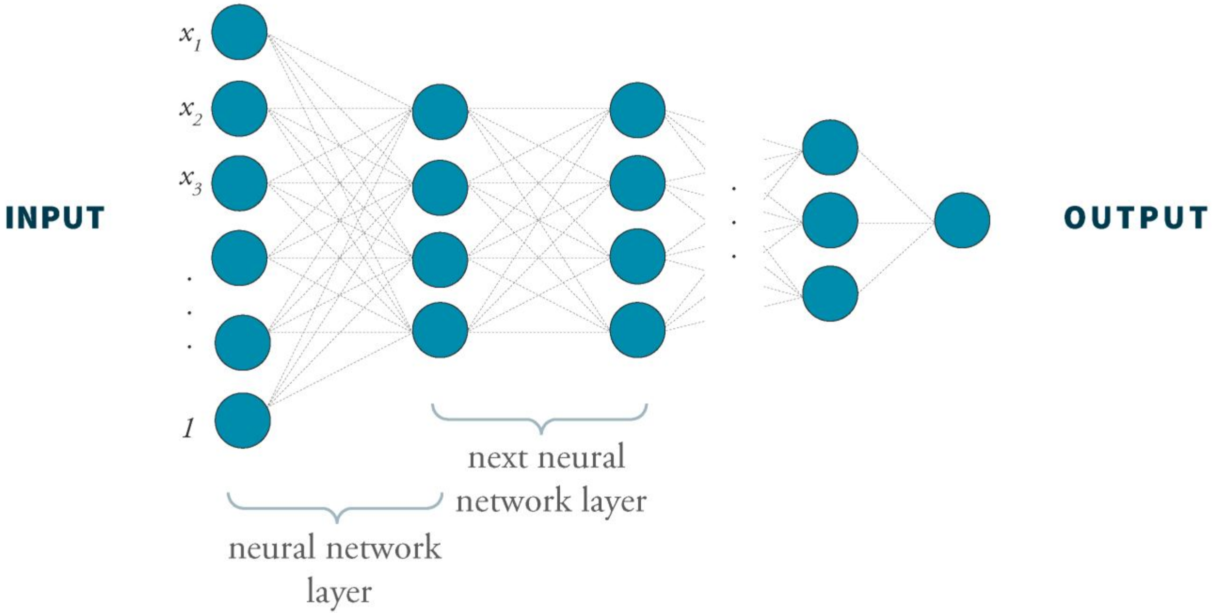


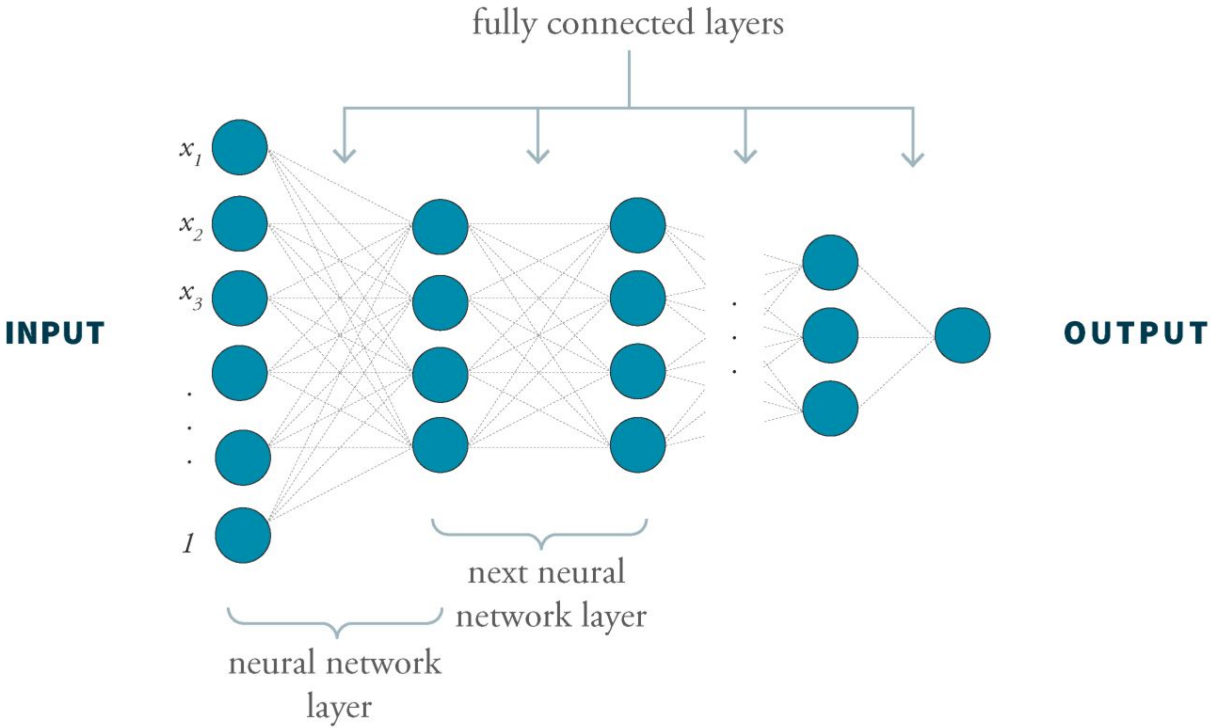


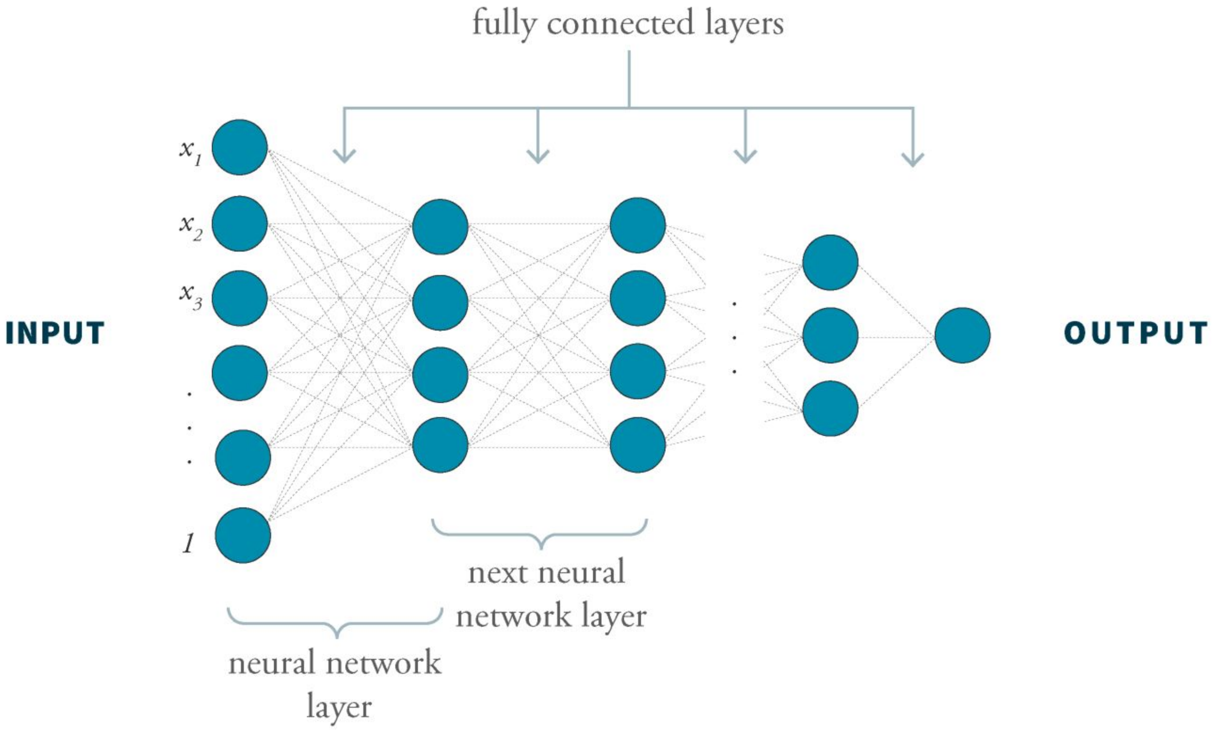
Represents increasingly complex (and hierarchical) function that is being computed!

INPUT



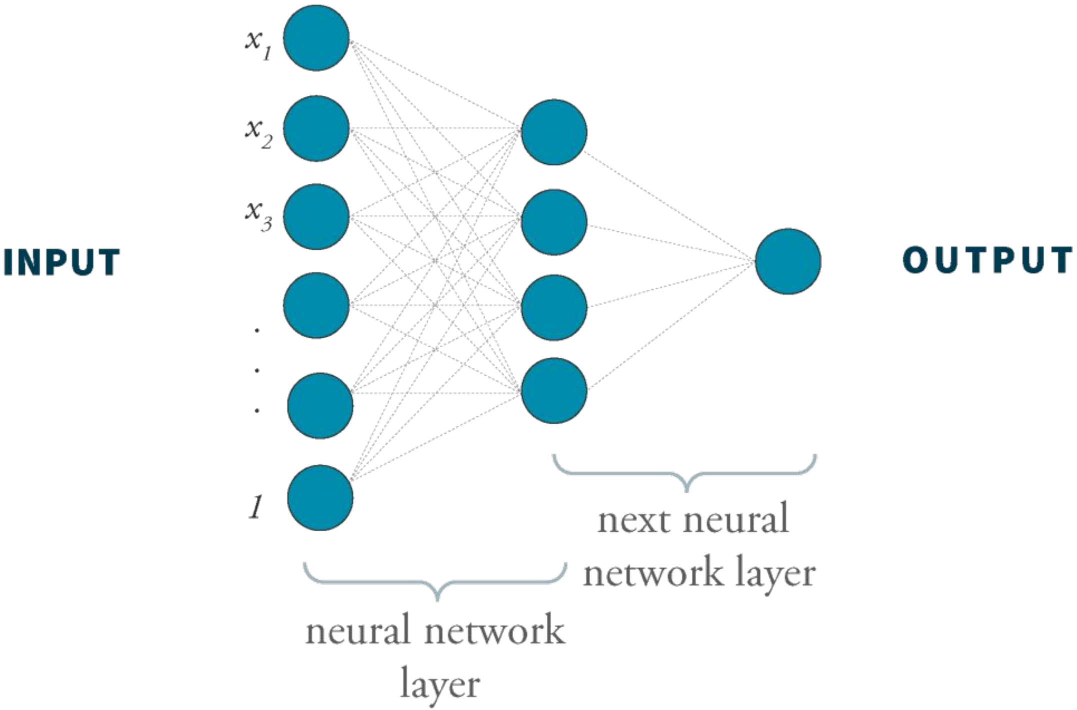

INPUT
x_1
x_2
x_3
1
neural network layer
next neural network layer
OUTPUT


fully connected layers
INPUT
x_1
x_2
x_3
1
OUTPUT
next neural network layer
neural network layer



Fully connected layer: all neurons in the layer takes as input the full input to the layer (also called dense layer or linear layer)

EXAMPLE: TWO-LAYER FULLY CONNECTED NEURAL NETWORK



How do we train neural networks to learn good values of the (many) parameters, to accurately map from inputs to desired outputs?

TRAINING LOOP

1. We start the program.
2. Feed each sample into our model
3. Our model will make a prediction based on the sample's features.
4. Compute the loss between the model's prediction and the sample's label.
5. The model will then update its parameters in a way that will reduce the loss it produces the next time it sees that same sample.
6. We can then evaluate our model on a validation set

TRAINING LOOP

1. We start the program.
2. Feed each sample into our model
3. Our model will make a prediction based on the sample's features.
4. Compute the loss between the model's prediction and the sample's label.
5. The model will then update its parameters in a way that will reduce the loss it produces the next time it sees that same sample.
6. We can then evaluate our model on a validation set

Optimization step

Periodically use validation set to measure how the model will do “in the real world”. Save a version of the model if it gives the best validation performance seen so far.

TRAINING LOOP

1. We start the program.
2. Feed each sample into our model
3. Our model will make a prediction based on the sample’s features.
4. Compute the loss between the model’s prediction and the sample’s label.
5. The model will then update its parameters in a way that will reduce the loss it produces the next time it sees that same sample.
6. We can then evaluate our model on a validation set

Can also run the entire process for different training configurations, or hyperparameters, to choose the best ones. Referred to as "**hyperparameter tuning**".

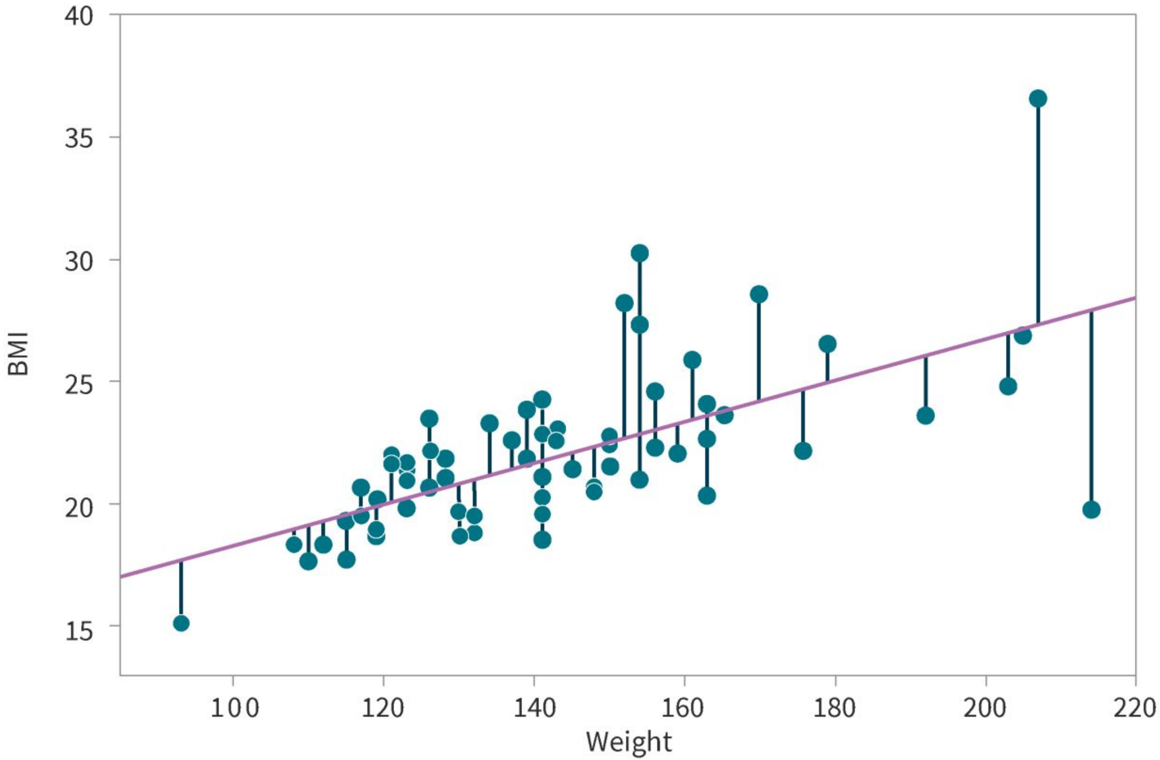
TRAINING LOOP

1. We start the program.
2. Feed each sample into our model
3. Our model will make a prediction based on the sample's features.
4. Compute the loss between the model's prediction and the sample's label.
5. The model will then update its parameters in a way that will reduce the loss it produces the next time it sees that same sample.
6. We can then evaluate our model on a validation set

Agenda

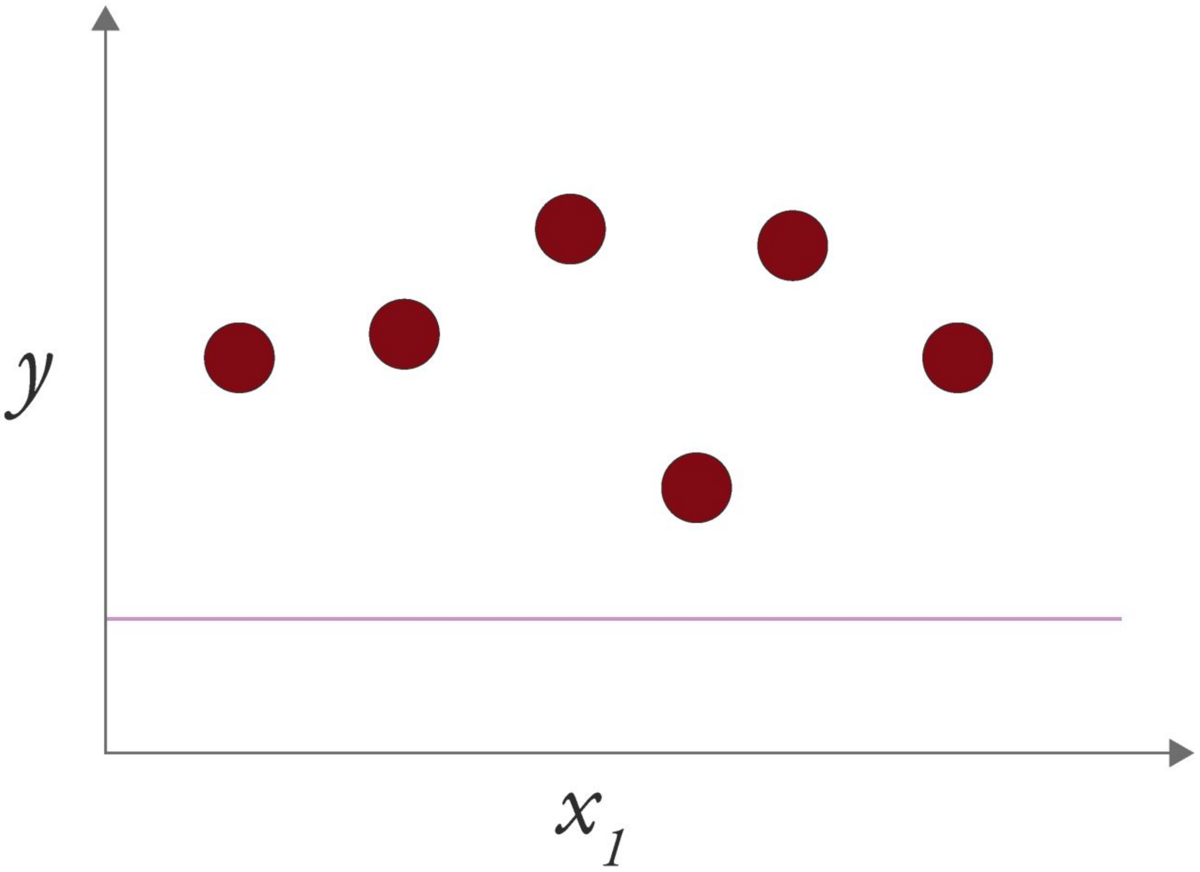
- More on the structure of neural network models
- **Machine learning training loop and concept of *loss*, in the context of neural networks**
- Minimizing the loss for complex neural networks: gradient descent and backpropagation

EXAMPLE: *USING BODY WEIGHT TO PREDICT BODY MASS INDEX (BMI)*

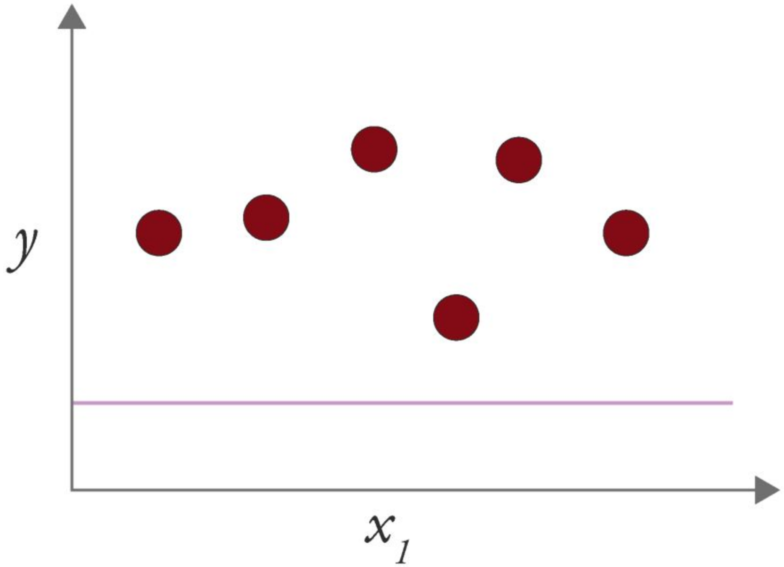


Loss: the difference between the function and the actual y values.

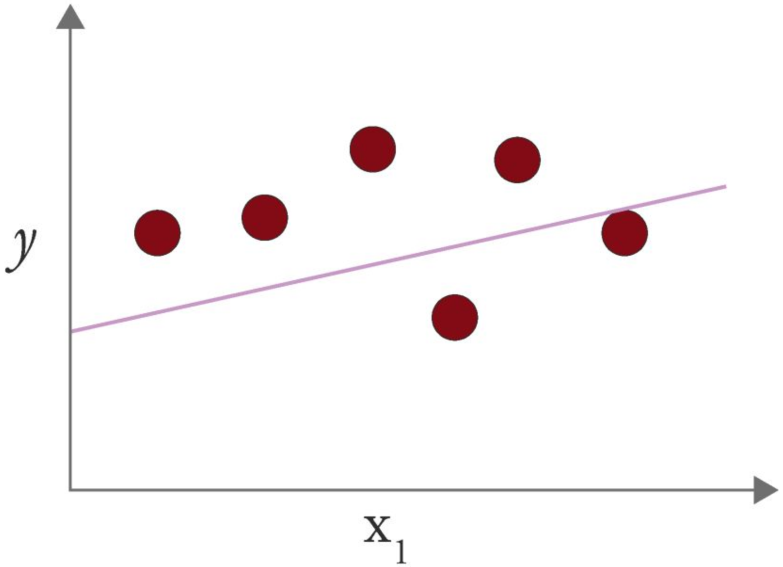
Training a machine learning model means to minimize the loss!



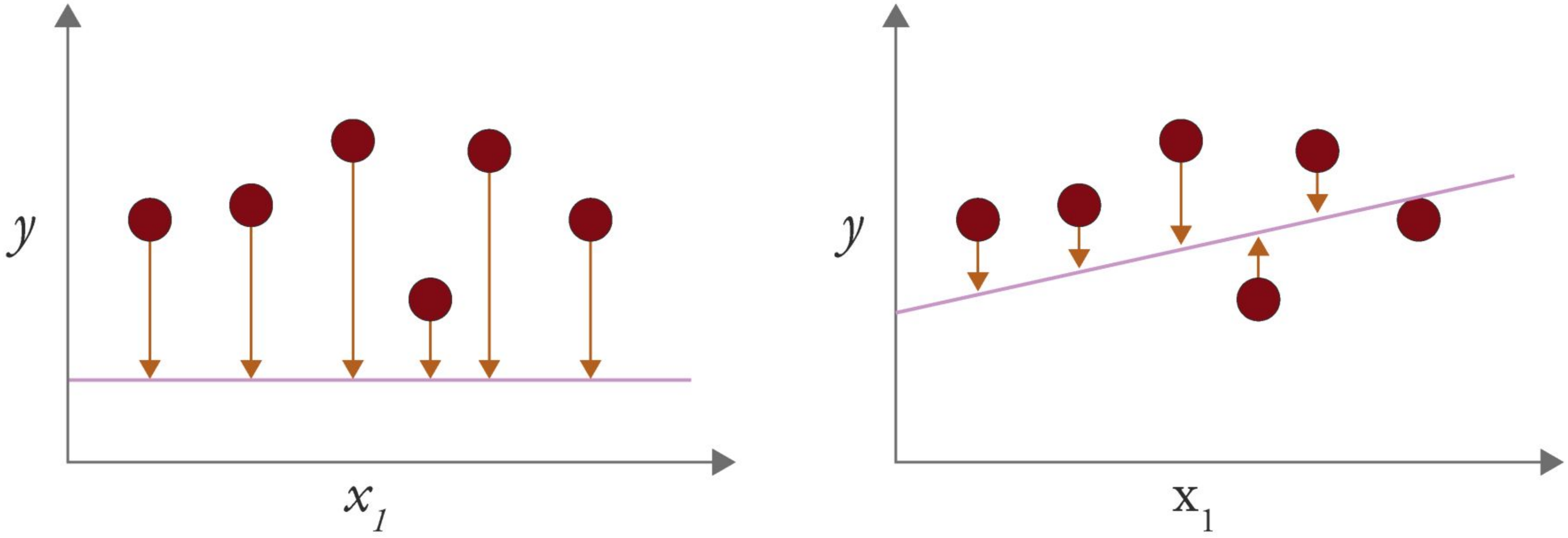
y
x_1



Higher loss



Lower loss


y
x1
y
x1

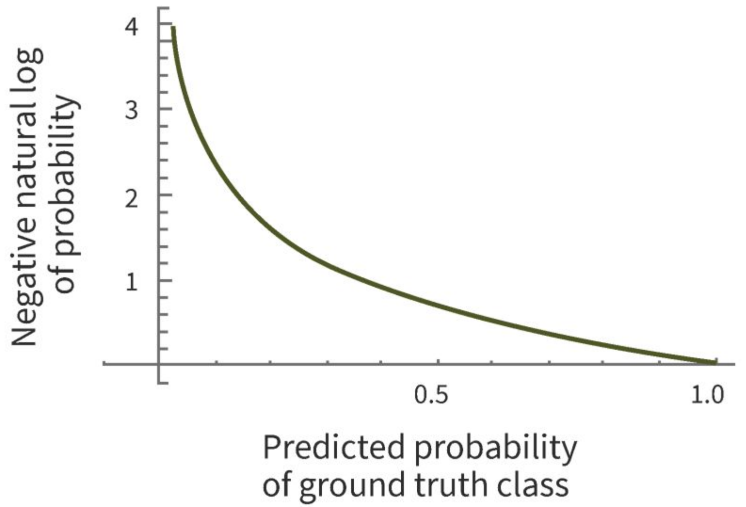
MOST COMMON LOSS FUNCTION FOR CLASSIFICATION

Cross Entropy Loss:

In practice often use ln (natural log)

01 Loss for one example = - ln (probability of ground truth class)

02 Loss over dataset = average of loss over all examples



	PROBABILITY APPENDICITIS	PROBABILITY DIVERTICULOSIS	PROBABILITY CHOLECYSTITIS
True label	0	1	1
Prediction	0.25	**0.60**	0.15

Cross-entropy loss: ~0.51

	PROBABILITY APPENDICITIS	PROBABILITY DIVERTICULOSIS	PROBABILITY CHOLECYSTITIS
True label	0	1	0
Prediction	0.05	**0.90**	0.05

Cross-entropy loss: ~0.15

Agenda

- More on the structure of neural network models
- Machine learning training loop and concept of *loss*, in the context of neural networks
- **Minimizing the loss for complex neural networks: gradient descent and backpropagation**

How can we find “good” values of many parameters?

How can we find “good” values of many parameters?

One option: Try all combination of possible weights and test how good each one is. But this would take forever, since there’s infinite possibilities and there is no indication of how best to adjust.

How can we find “good” values of many parameters?

One option: Try all combination of possible weights and test how good each one is. But this would take forever, since there’s infinite possibilities and there is no indication of how best to adjust.

Instead: the trick is that we need to have some idea of which “direction” to adjust the weights to reduce the loss function.

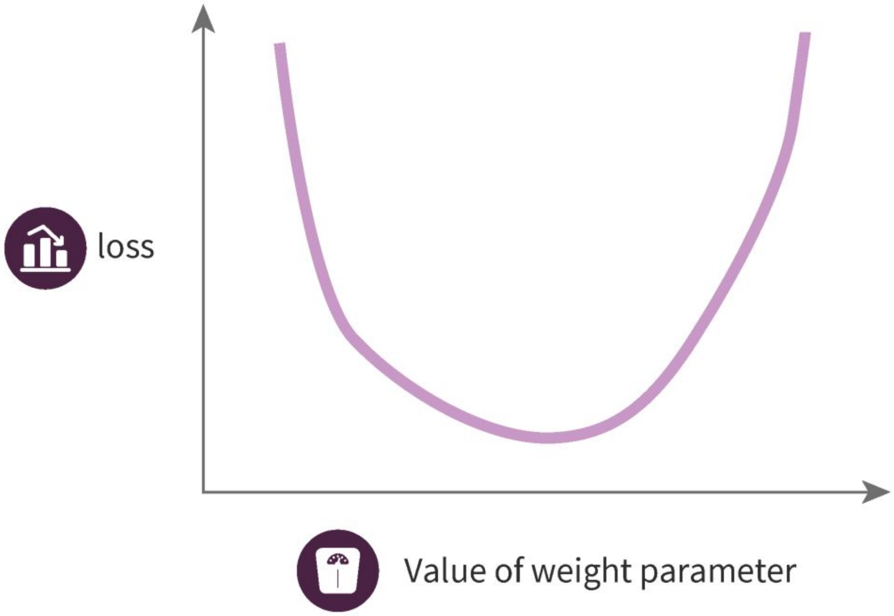
Analogy: the game of Marco Polo!

“Yeah, I pretty much never sit by the pool anymore”

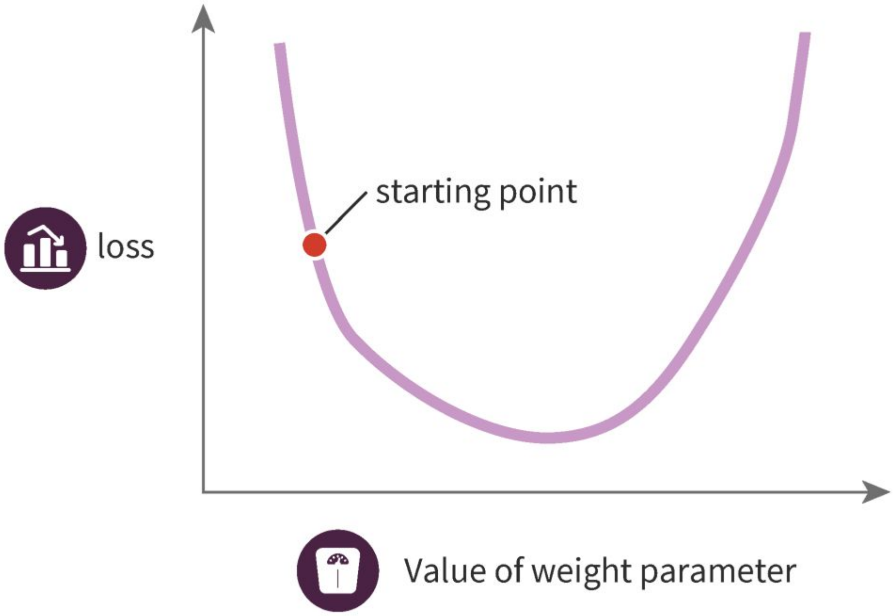
-MARCO POLO

A GRAPHICAL LOOK AT GRADIENT DESCENT:

1D case (single weight parameter)

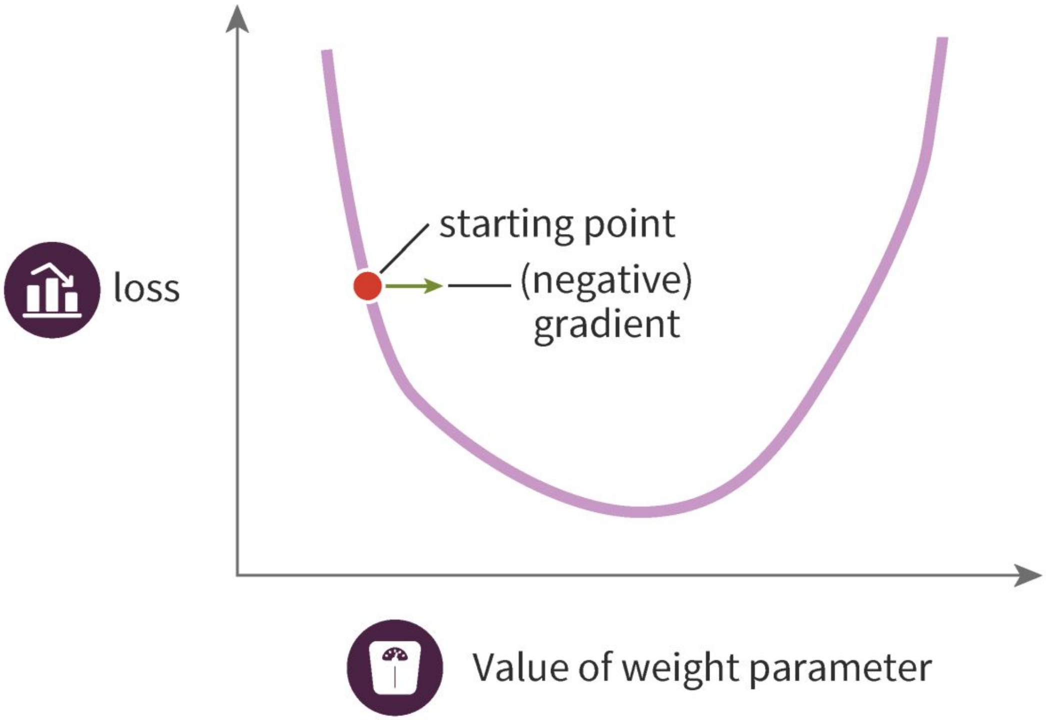


A GRAPHICAL LOOK AT GRADIENT DESCENT:
1D case (single weight parameter)



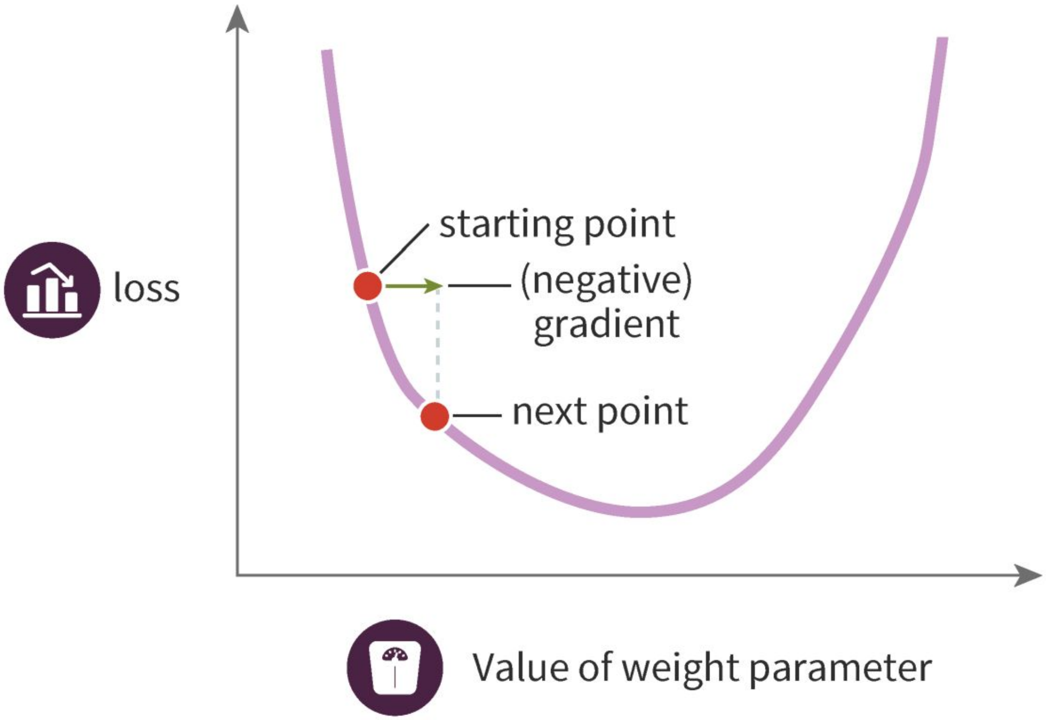
A GRAPHICAL LOOK AT GRADIENT DESCENT:

1D case (single weight parameter)



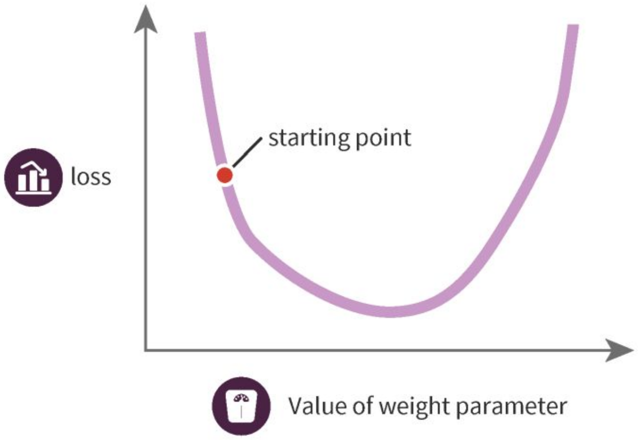
A GRAPHICAL LOOK AT GRADIENT DESCENT:

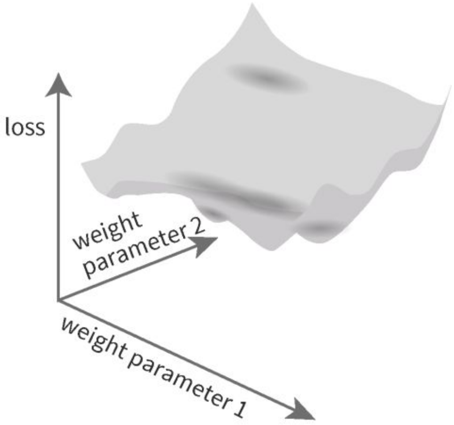
1D case (single weight parameter)



A GRAPHICAL LOOK AT GRADIENT DESCENT:

1D case (single weight parameter)





Too complex to visualize

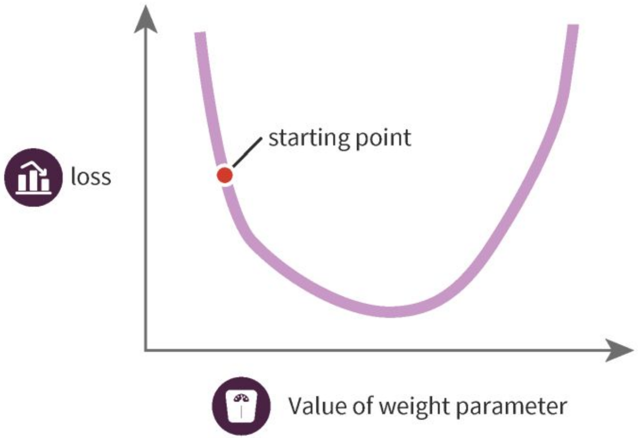
Function with 1 weight parameter

Function with 2 weight parameter

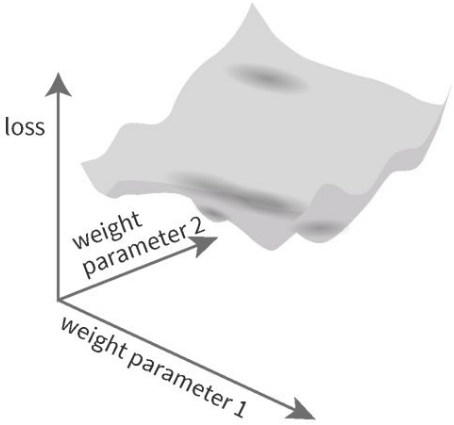
Modern neural network (millions of weight parameters!)

Backpropagation: mathematical technique that breaks down complex gradient computation into local gradient computations that are then combined together. Secret sauce for allowing us to obtain gradient for large neural network models!
(with the help of graphical processing units or GPUs)

A GRAPHICAL LOOK AT GRADIENT DESCENT:
1D case (single weight parameter)



Function with 1
weight parameter



Function with 2
weight parameter

**Too complex
to visualize**

Modern neural network
(millions of weight parameters!)

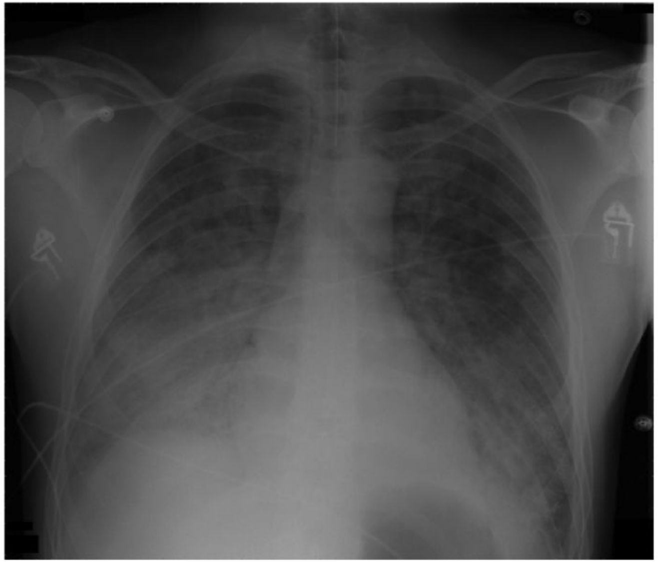
Now that we have a deeper understanding of neural networks, let’s look at how they work for common types of input data...

GRAYSCALE IMAGES

- Pixels → number grid
- Pixel brightness → grid value

```
[[  2  2  1  37   1  10  66  60  77  94  78  69  64  23  12  45  28  45 ]
 [ 58  1  9  13  17  29  56  72  65  64  59  58  39  18  15  12   7   1 ]
 [ 71 49 53  38  30  41  73  73  80  71  69  69  72  45  45  49  36  59 ]
 [ 88 60 73  50  59  59  54  51  71  81  69  50  54  75  56  61  80  67 ]
 [ 94 91 86  59  65  57  57  52  64  88  66  56  55  54  70  64 109 114 ]
 [ 94 95 84  74  70  41  48  55  74  85  84  60  50  46  70  82  92 122 ]
 [ 85 85 95  83  54  37  59  60  84  97  82  50  38  44  56  92 111 112 ]
 [ 81 87 94  92  54  54  56  54  79  96  79  48  36  44  62 103 107 145 ]
 [ 67 83 91  87  60  59  61  71  91 108  86  65  53  40  63 101 110 121 ]
 [ 49 73 88  72  66  73  78  84 107 120 102  71  57  39  56  89 114 103 ]
 [ 31 61 84  65  73  80  92 103 117 128 114  76  66  57  52  89 111  91 ]
 [  6 51 82  84  92  90  92 114 128 135 122 109  73  69  69  84 109  66 ]
 [  2 44 72  87  95 104 113 124 138 141 130 122  96  77  68  76 104  10 ]
 [  0 37 74  84 102 113 115 131 146 146 133 124 113  94  83  96  90   1 ]
 [  0 33 67  90 113 126 130 140 148 147 136 130 117  95  91  81  71   1 ]
 [  0 33 68  98 122 139 141 144 153 149 135 127 122 108  96  76  65   1 ]
 [  0 36 81 105 127 144 151 151 155 149 125 114 113 121 105  76  49   1 ]
 [  0 39 90 114 131 151 155 157 161 153 122  96 102 107 110  66  50   1 ]]
```

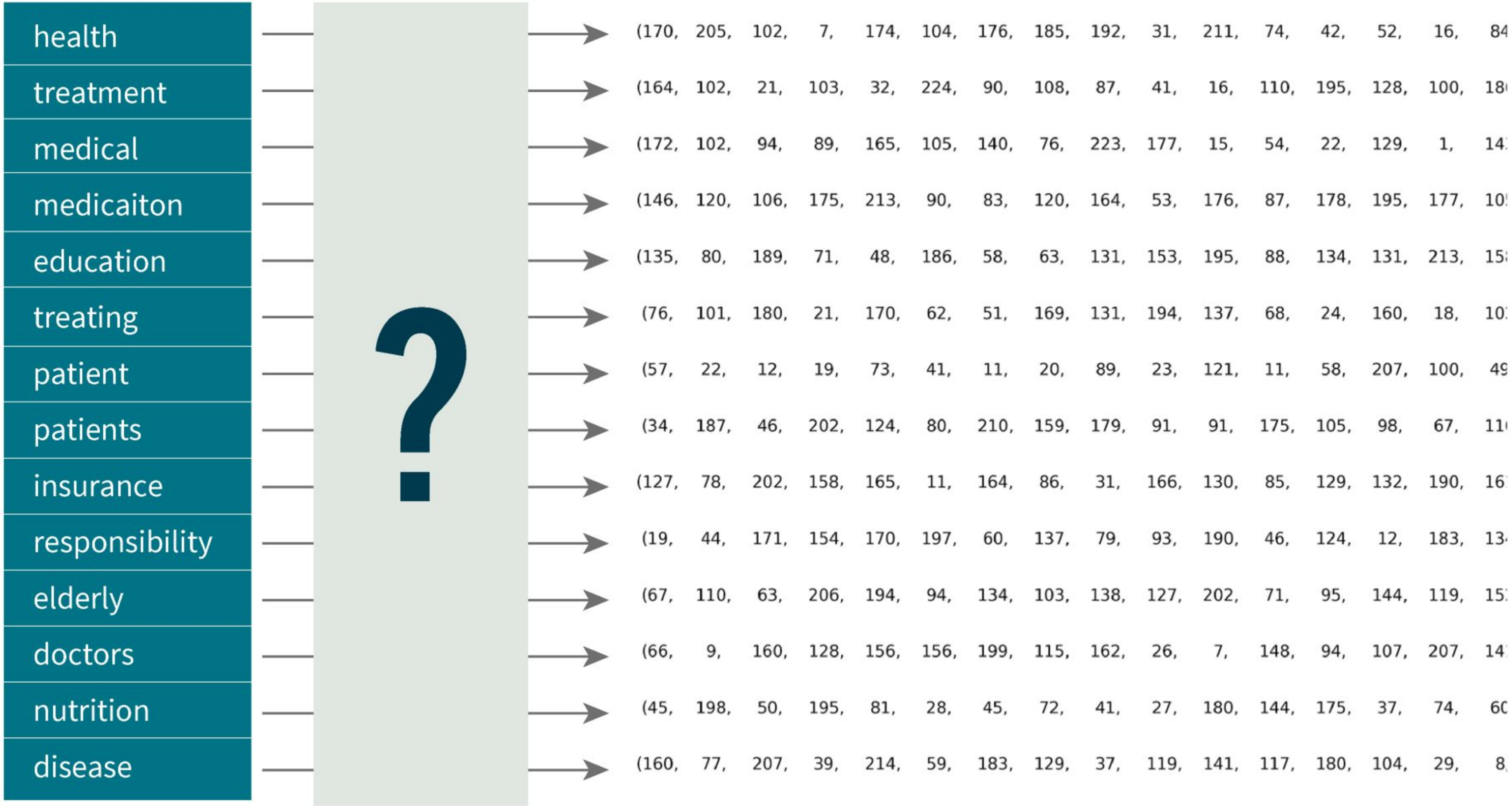
UNSTRUCTURED DATA



Physician Note

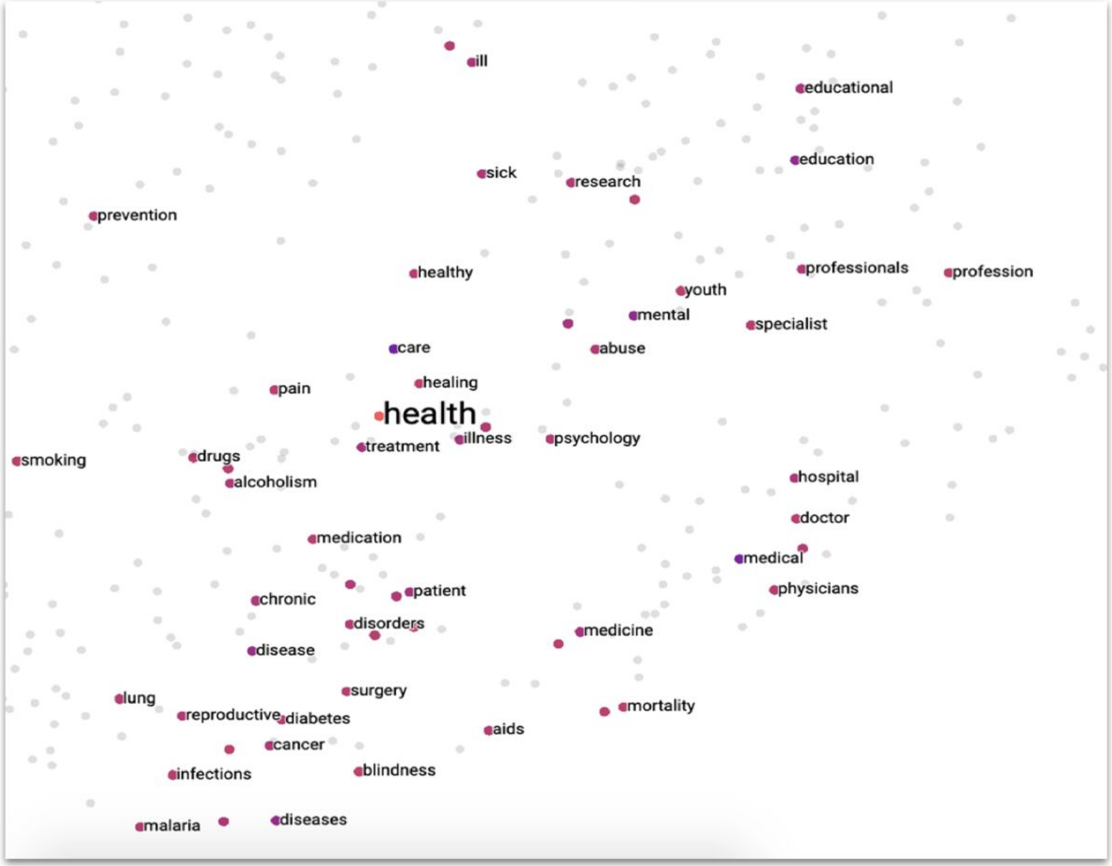
"...PMH of **metastatic breast cancer, R lung malignant** effusion, and **R lung empyema** who presents with increased drainage from **R lung pleurx** tract..."

health	(170,	205,	102,	7,	174,	104,	176,	185,	192,	31,	211,	74,	42,	52,	16,	84,	80,	120,	101,
treatment	(164,	102,	21,	103,	32,	224,	90,	108,	87,	41,	16,	110,	195,	128,	100,	186,	111,	211,	215,
medical	(172,	102,	94,	89,	165,	105,	140,	76,	223,	177,	15,	54,	22,	129,	1,	143,	176,	47,	191,
medicaiton	(146,	120,	106,	175,	213,	90,	83,	120,	164,	53,	176,	87,	178,	195,	177,	105,	56,	74,	203,
education	(135,	80,	189,	71,	48,	186,	58,	63,	131,	153,	195,	88,	134,	131,	213,	158,	101,	171,	163,
treating	(76,	101,	180,	21,	170,	62,	51,	169,	131,	194,	137,	68,	24,	160,	18,	102,	5,	20,	112,
patient	(57,	22,	12,	19,	73,	41,	11,	20,	89,	23,	121,	11,	58,	207,	100,	49,	48,	43,	11,
patients	(34,	187,	46,	202,	124,	80,	210,	159,	179,	91,	91,	175,	105,	98,	67,	110,	28,	195,	220,
insurance	(127,	78,	202,	158,	165,	11,	164,	86,	31,	166,	130,	85,	129,	132,	190,	161,	67,	82,	28,
responsibility	(19,	44,	171,	154,	170,	197,	60,	137,	79,	93,	190,	46,	124,	12,	183,	134,	48,	119,	179,
elderly	(67,	110,	63,	206,	194,	94,	134,	103,	138,	127,	202,	71,	95,	144,	119,	152,	109,	95,	47,
doctors	(66,	9,	160,	128,	156,	156,	199,	115,	162,	26,	7,	148,	94,	107,	207,	141,	37,	174,	81,
nutrition	(45,	198,	50,	195,	81,	28,	45,	72,	41,	27,	180,	144,	175,	37,	74,	60,	208,	197,	109,
disease	(160,	77,	207,	39,	214,	59,	183,	129,	37,	119,	141,	117,	180,	104,	29,	8,	144,	183,	112,



health
treatment
medical
medicaiton
education
treating
patient
patients
insurance
responsibility
elderly
doctors
nutrition
disease
?
(170, 205, 102, 7, 174, 104, 176, 185, 192, 31, 211, 74, 42, 52, 16, 84
(164, 102, 21, 103, 32, 224, 90, 108, 87, 41, 16, 110, 195, 128, 100,
(172, 102, 94, 89, 165, 105, 140, 76, 223, 177, 15, 54, 22, 129, 1,
(146, 120, 106, 175, 213, 90, 83, 120, 164, 53, 176, 87, 178, 195, 177,
(135, 80, 189, 71, 48, 186, 58, 63, 131, 153, 195, 88, 134, 131, 213,
(76, 101, 180, 21, 170, 62, 51, 169, 131, 194, 137, 68, 24, 160, 18,
(57, 22, 12, 19, 73, 41, 11, 20, 89, 23, 121, 11, 58, 207, 100,
(34, 187, 46, 202, 124, 80, 210, 159, 179, 91, 91, 175, 105, 98, 67,
(127, 78, 202, 158, 165, 11, 164, 86, 31, 166, 130, 85, 129, 132, 190,
(19, 44, 171, 154, 170, 197, 60, 137, 79, 93, 190, 46, 124, 12, 183,
(67, 110, 63, 206, 194, 94, 134, 103, 138, 127, 202, 71, 95, 144, 119,
(66, 9, 160, 128, 156, 156, 199, 115, 162, 26, 7, 148, 94, 107, 207,
(45, 198, 50, 195, 81, 28, 45, 72, 41, 27, 180, 144, 175, 37, 74,
(160, 77, 207, 39, 214, 59, 183, 129, 37, 119, 141, 117, 180, 104, 29, 8,

health	(178,168)
treatment	(165, 50)
medical	(209,159)
medicaiton	(208,171)
education	(73, 213)
treating	(207, 68)
patient	(185, 1)
patients	(36, 187)
insurance	(176, 52)
responsibility	(165, 87)
elderly	(94, 17)
doctors	(133,199)
nutrition	(53, 142)
disease	(217,144)



UNSTRUCTURED DATA

Physician Note

"...PMH of **metastatic breast cancer, R lung malignant** effusion, and **R lung empyema** who presents with increased drainage from **R lung pleurx** tract..."

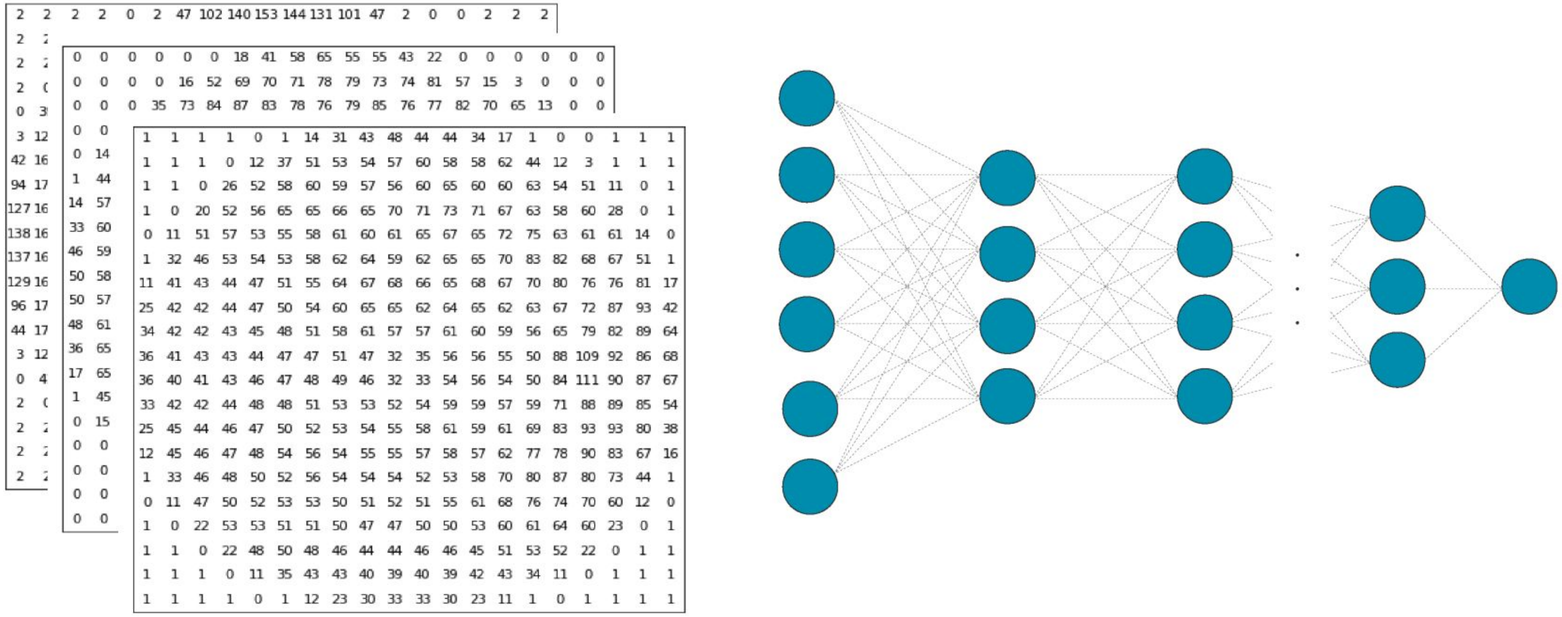
STRUCTURED DATA

GRAYSCALE IMAGES

- Pixels → number grid
- Pixel brightness → grid value

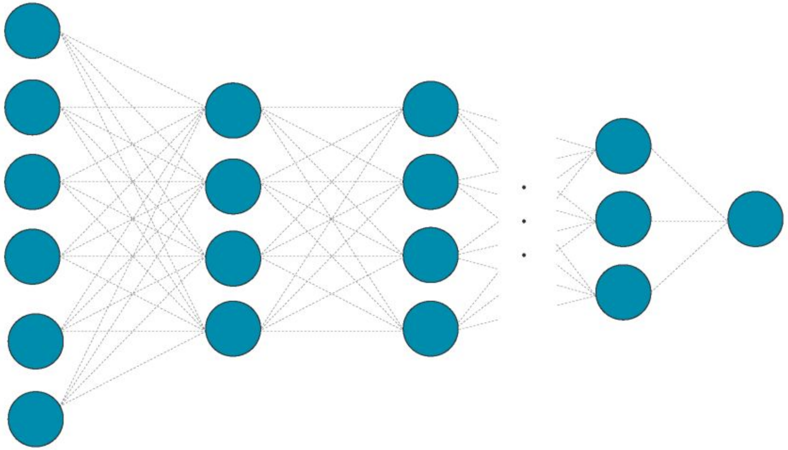
```
[[   2   2   1  37   1  10  66  60  77  94  78  69  64  23  12  45  28  45 ]
 [  58   1   9  13  17  29  56  72  65  64  59  58  39  18  15  12   7   1 ]
 [  71  49  53  38  30  41  73  73  80  71  69  69  72  45  45  49  36  59 ]
 [  88  60  73  50  59  59  54  51  71  81  69  50  54  75  56  61  80  67 ]
 [  94  91  86  59  65  57  57  52  64  88  66  56  55  54  70  64 109 114 ]
 [  94  95  84  74  70  41  48  55  74  85  84  60  50  46  70  82  92 122 ]
 [  85  85  95  83  54  37  59  60  84  97  82  50  38  44  56  92 111 112 ]
 [  81  87  94  92  54  54  56  54  79  96  79  48  36  44  62 103 107 145 ]
 [  67  83  91  87  60  59  61  71  91 108  86  65  53  40  63 101 110 121 ]
 [  49  73  88  72  66  73  78  84 107 120 102  71  57  39  56  89 114 103 ]
 [  31  61  84  65  73  80  92 103 117 128 114  76  66  57  52  89 111  91 ]
 [   6  51  82  84  92  90  92 114 128 135 122 109  73  69  69  84 109  66 ]
 [   2  44  72  87  95 104 113 124 138 141 130 122  96  77  68  76 104  10 ]
 [   0  37  74  84 102 113 115 131 146 146 133 124 113  94  83  96  90   1 ]
 [   0  33  67  90 113 126 130 140 148 147 136 130 117  95  91  81  71   1 ]
 [   0  33  68  98 122 139 141 144 153 149 135 127 122 108  96  76  65   1 ]
 [   0  36  81 105 127 144 151 151 155 149 125 114 113 121 105  76  49   1 ]
 [   0  39  90 114 131 151 155 157 161 153 122  96 102 107 110  66  50   1 ]]
```

224 X 224 COLORED IMAGE



FIRST ATTEMPT: FLATTEN THE IMAGE

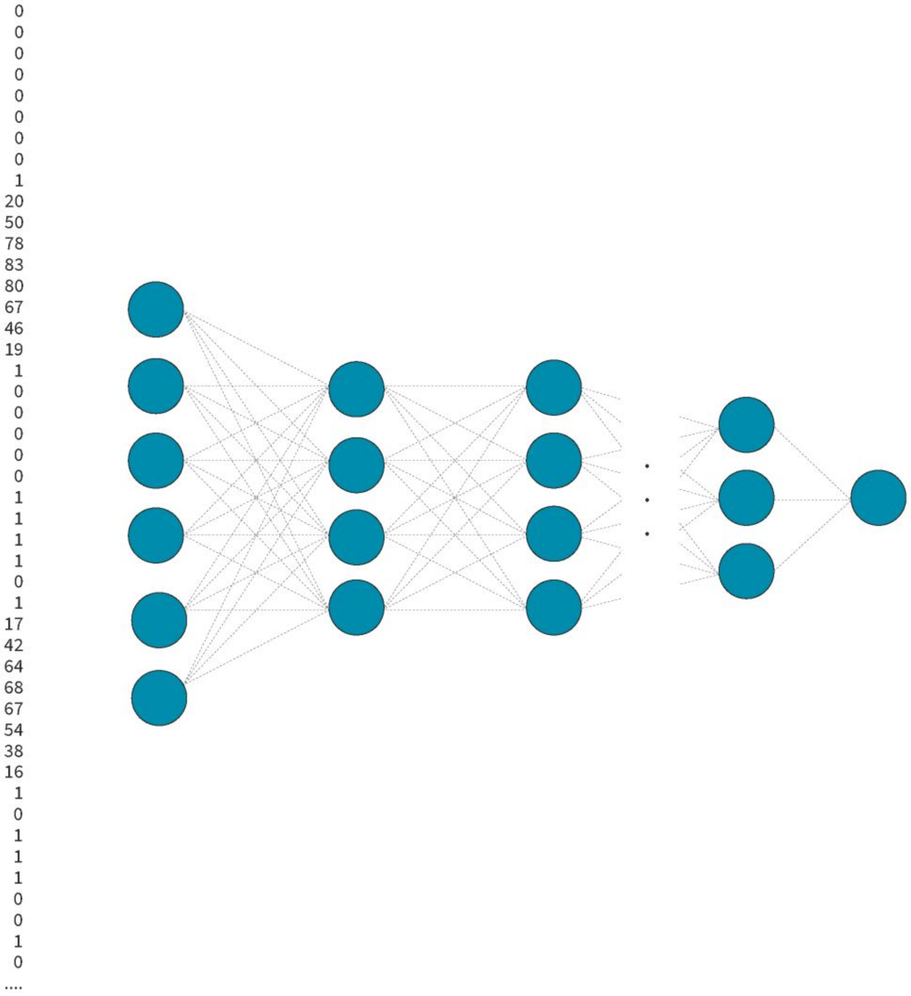
0
0
0
0
0
0
0
0
1
20
50
78
83
80
67
46
19
1
0
0
0
0
0
1
1
1
1
0
1
17
42
64
68
67
54
38
16
1
0
1
1
1
0
0
1
0
....



FIRST ATTEMPT: FLATTEN THE IMAGE

Problems:

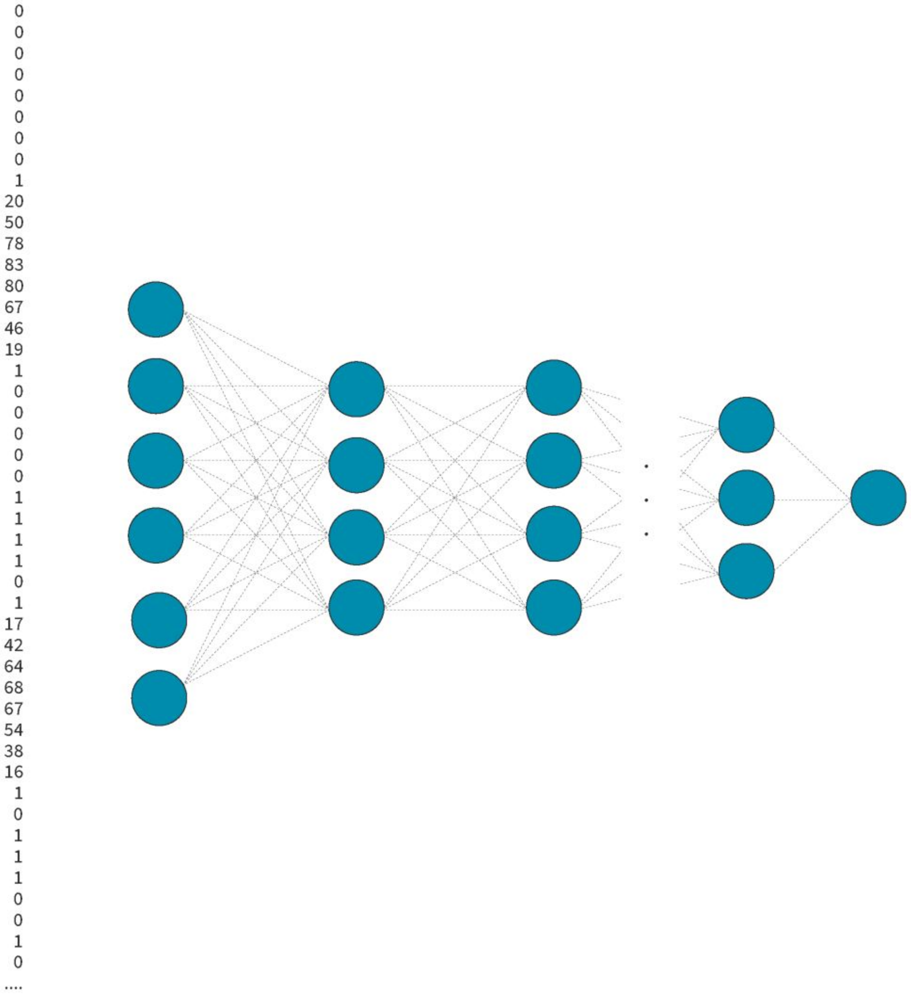
- Requires a HUGE number of parameters



FIRST ATTEMPT: FLATTEN THE IMAGE

Problems:

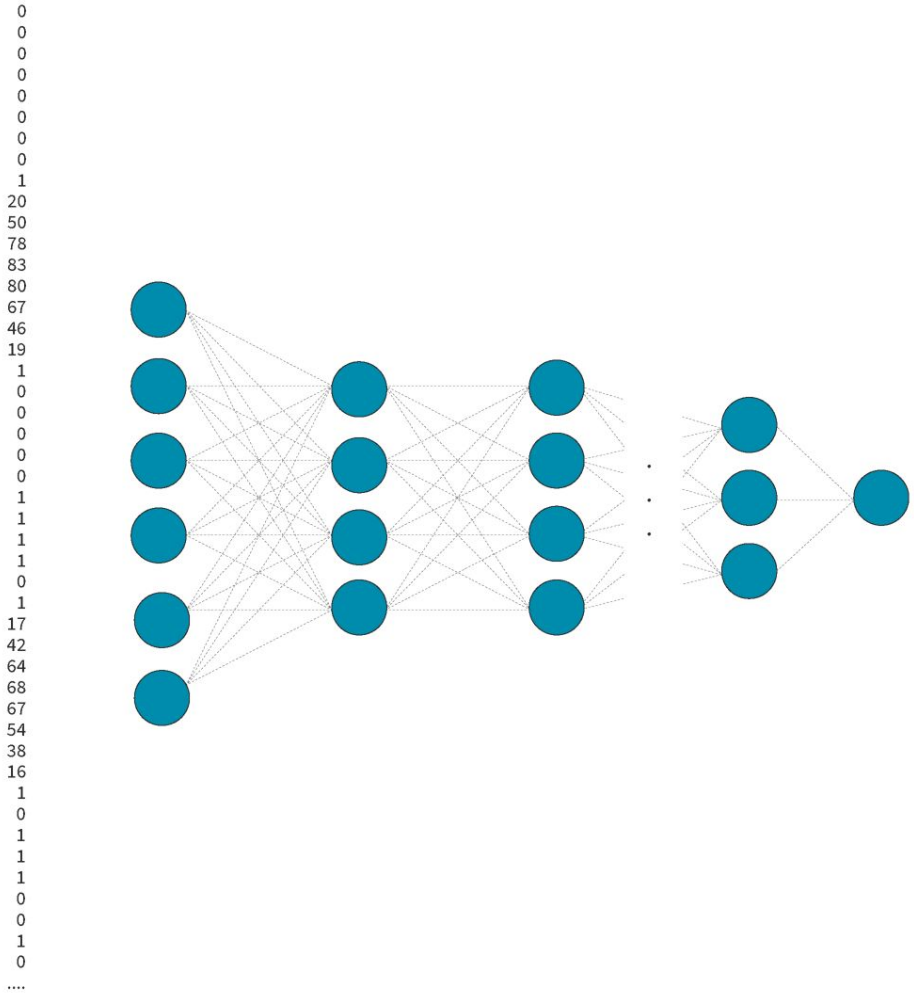
- Requires a HUGE number of parameters
- Highly sensitive to object displacement



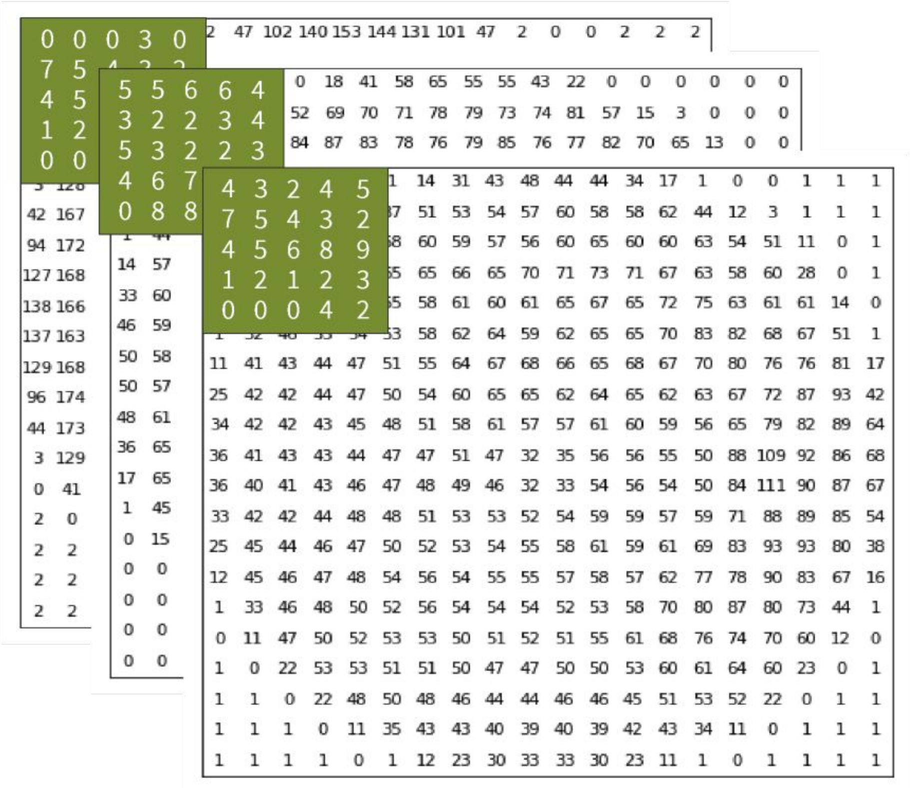
FIRST ATTEMPT: FLATTEN THE IMAGE

Problems:

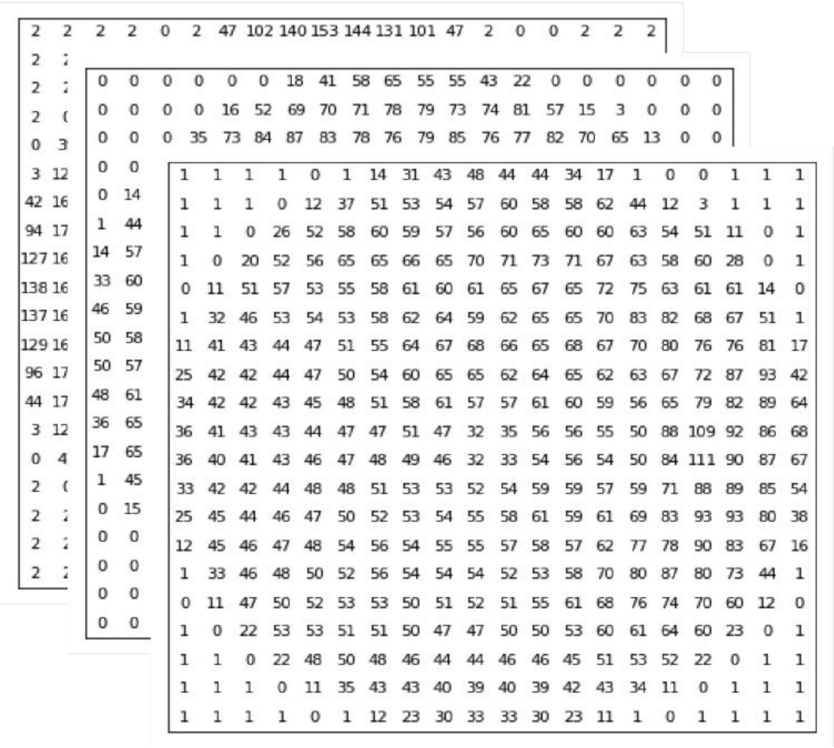
- Requires a HUGE number of parameters
- Highly sensitive to object displacement
- Destroys spatial information



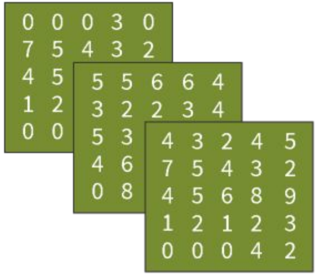
IDEA: ANALYZE THE IMAGE BY REGION



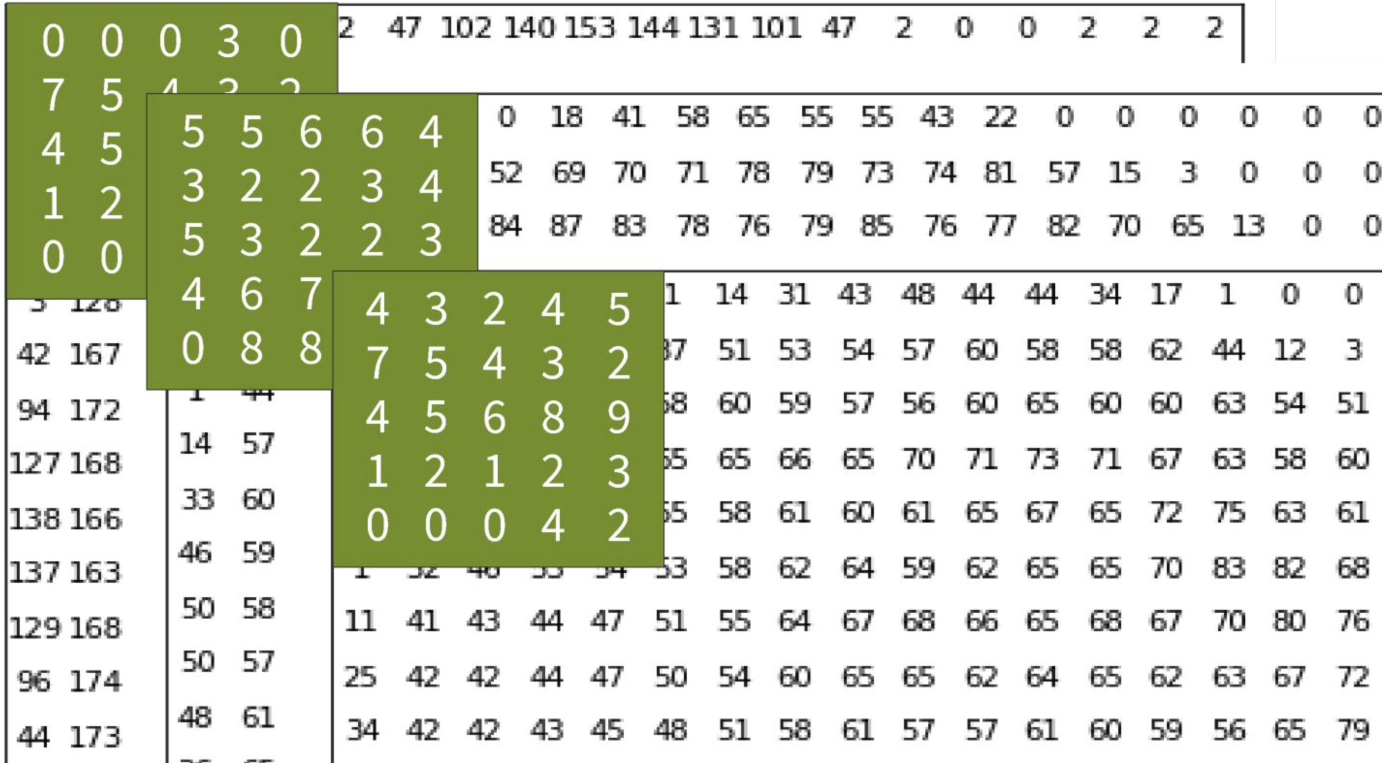
IDEA: ANALYZE THE IMAGE BY REGION



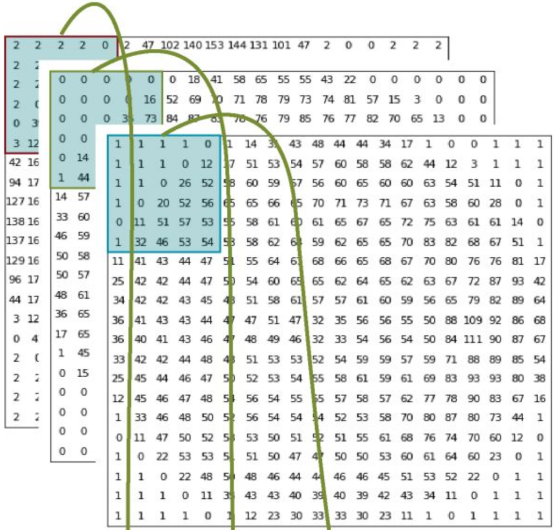
Convolutional filter



IDEA: ANALYZE THE IMAGE BY REGION

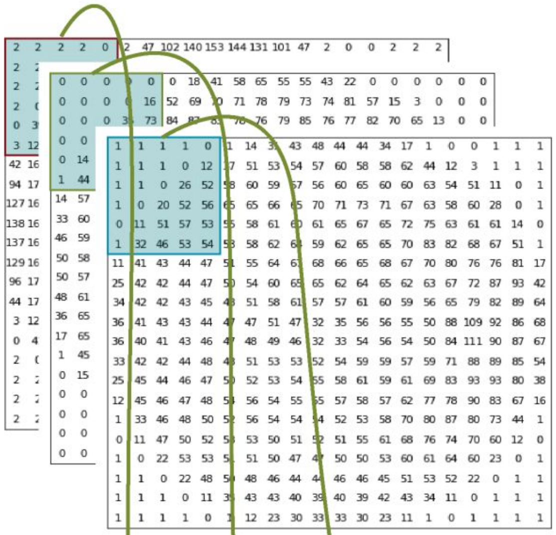


IDEA: ANALYZE THE IMAGE BY REGION



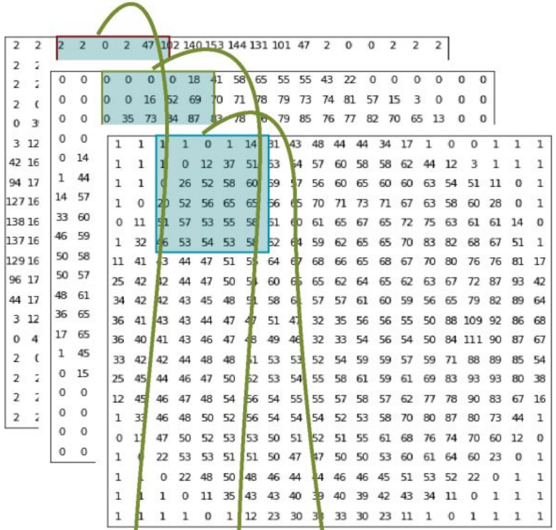
sigmoid ($\boxed{10}$ + $\boxed{10}$ + $\boxed{20}$) = $\boxed{0.9}$

IDEA: ANALYZE THE IMAGE BY REGION

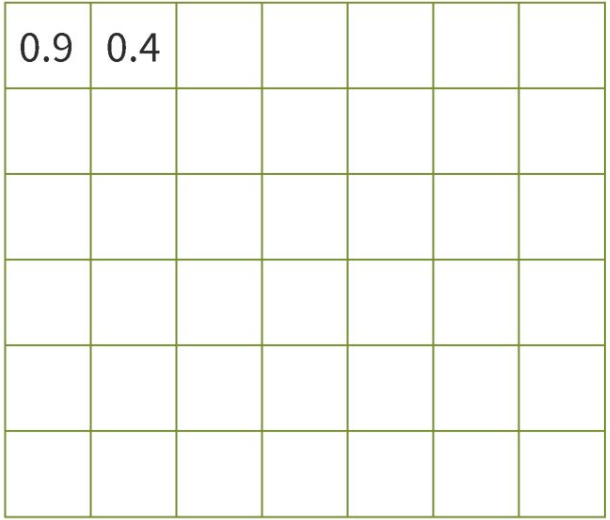


$$\text{sigmoid}\left(\boxed{10}+\boxed{10}+\boxed{20}\right)=\boxed{0.9}$$

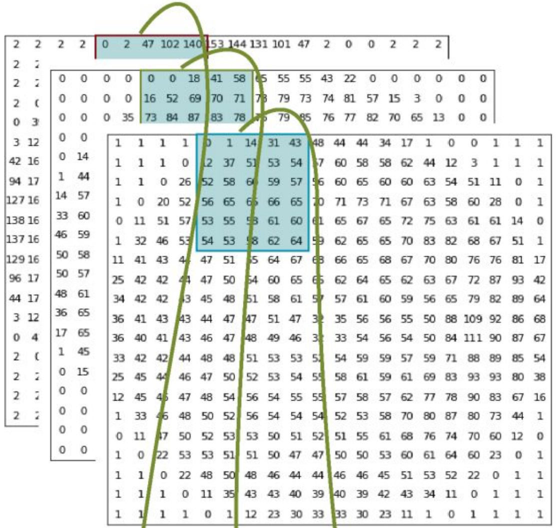
IDEA: ANALYZE THE IMAGE BY REGION



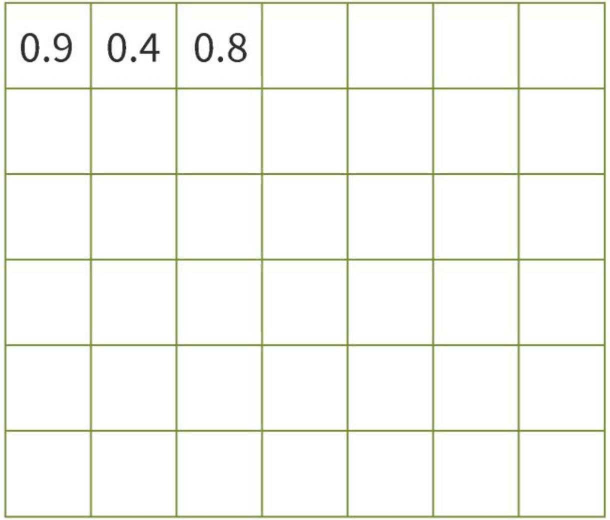
sigmoid ($\boxed{4}$ + $\boxed{2}$ + $\boxed{10}$) = $\boxed{0.4}$



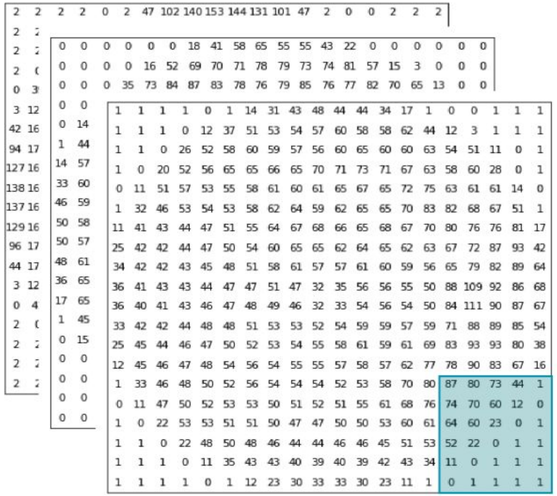
IDEA: ANALYZE THE IMAGE BY REGION



$$\textbf{sigmoid}\left(\boxed{9} + \boxed{12} + \boxed{10}\right) = \boxed{0.8}$$



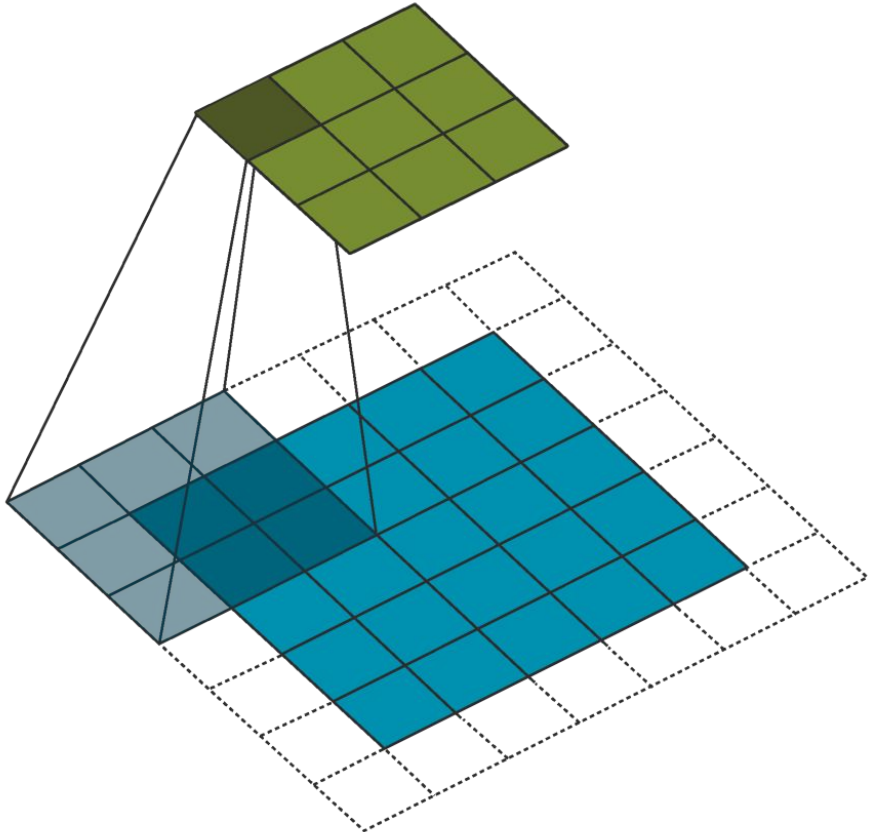
IDEA: ANALYZE THE IMAGE BY REGION

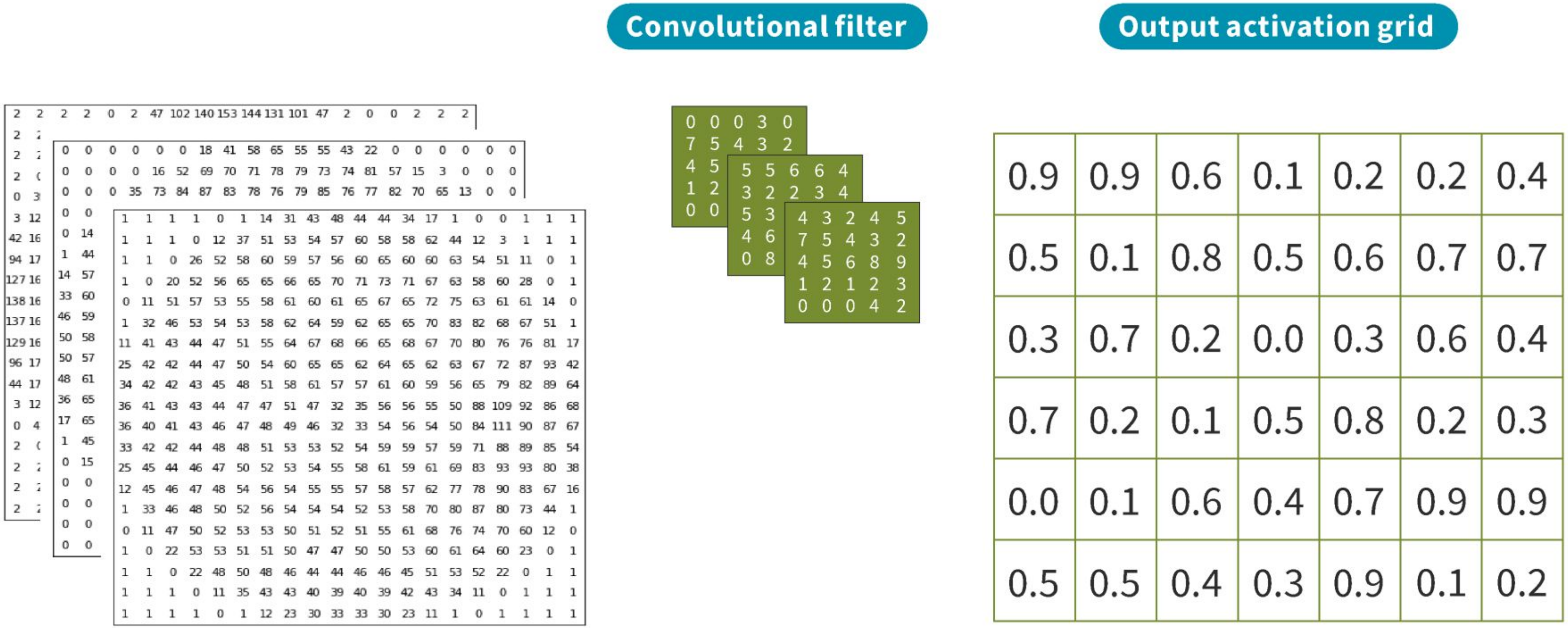


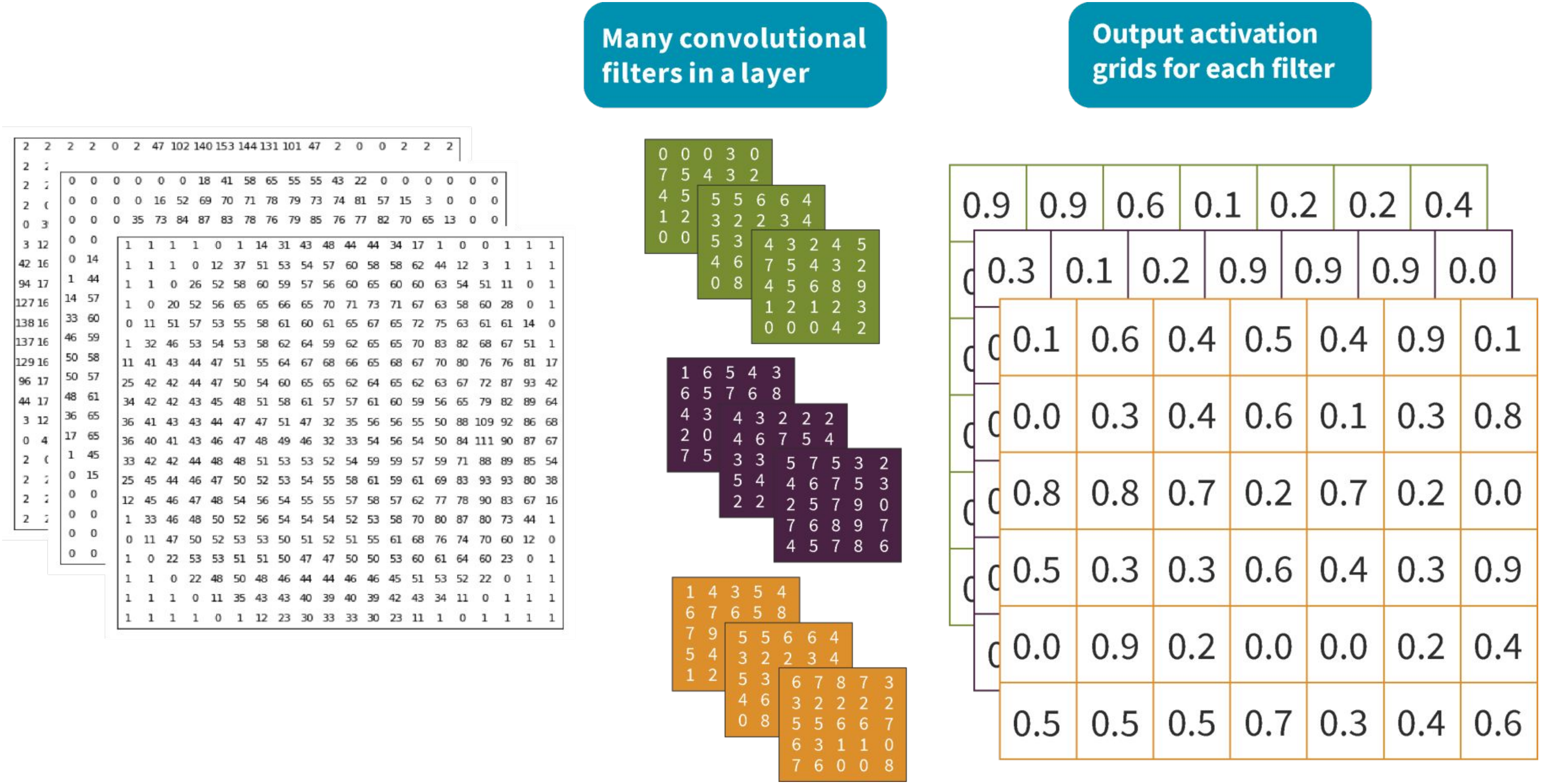
0.9	0.9	0.6	0.1	0.2	0.2	0.4
0.5	0.1	0.8	0.5	0.6	0.7	0.7
0.3	0.7	0.2	0.0	0.3	0.6	0.4
0.7	0.2	0.1	0.5	0.8	0.2	0.3
0.0	0.1	0.6	0.4	0.7	0.9	0.9
0.5	0.5	0.4	0.3	0.9	0.1	0.2

$$\textbf{sigmoid}\left(\boxed{0} + \boxed{6} + \boxed{3}\right) = \boxed{0.1}$$

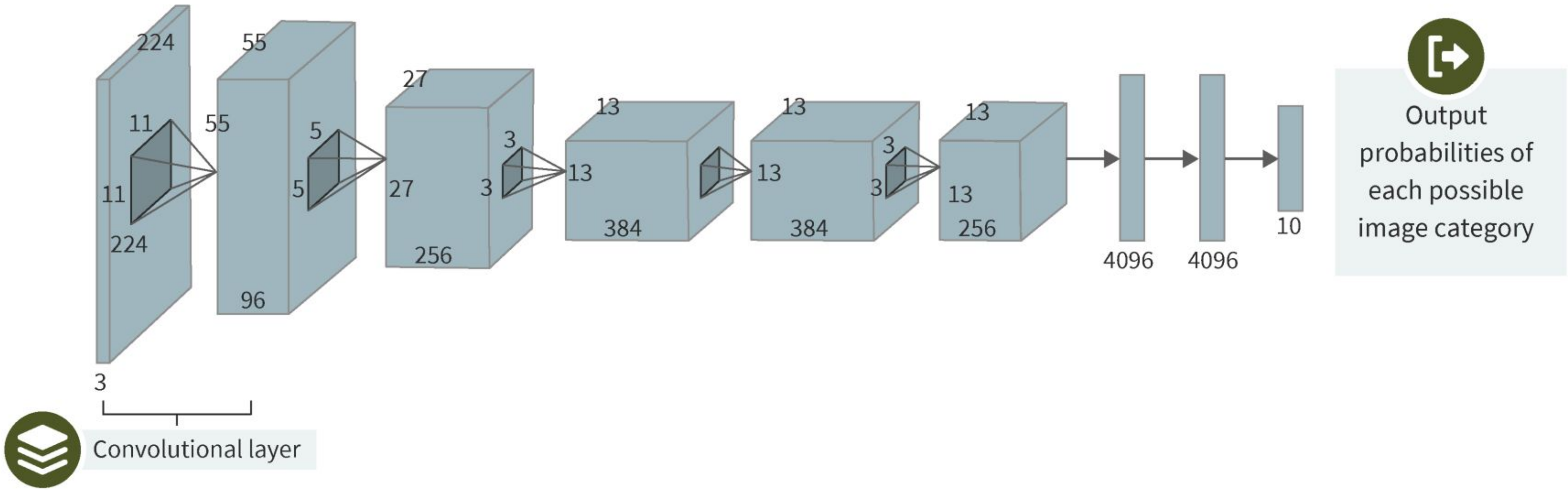
ANOTHER VISUALIZATION OF A CONVOLUTIONAL FILTER



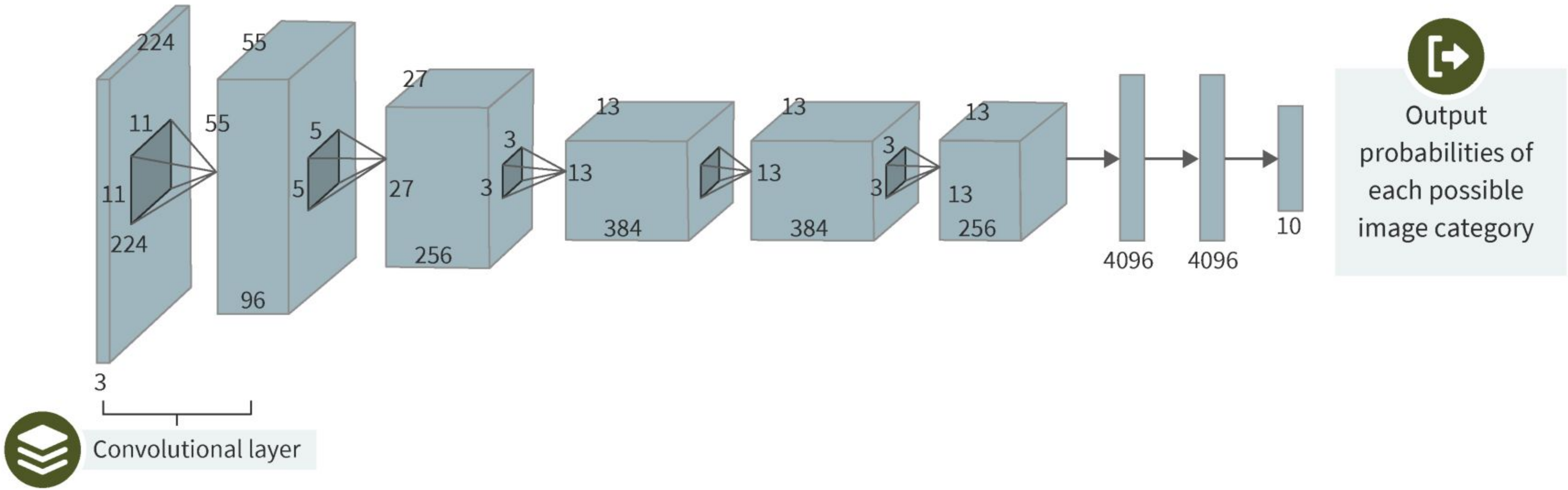

Convolutional filter
Output activation grid


Many convolutional filters in a layer
Output activation grids for each filter

CONVOLUTIONAL NEURAL NETWORKS (CNNs)



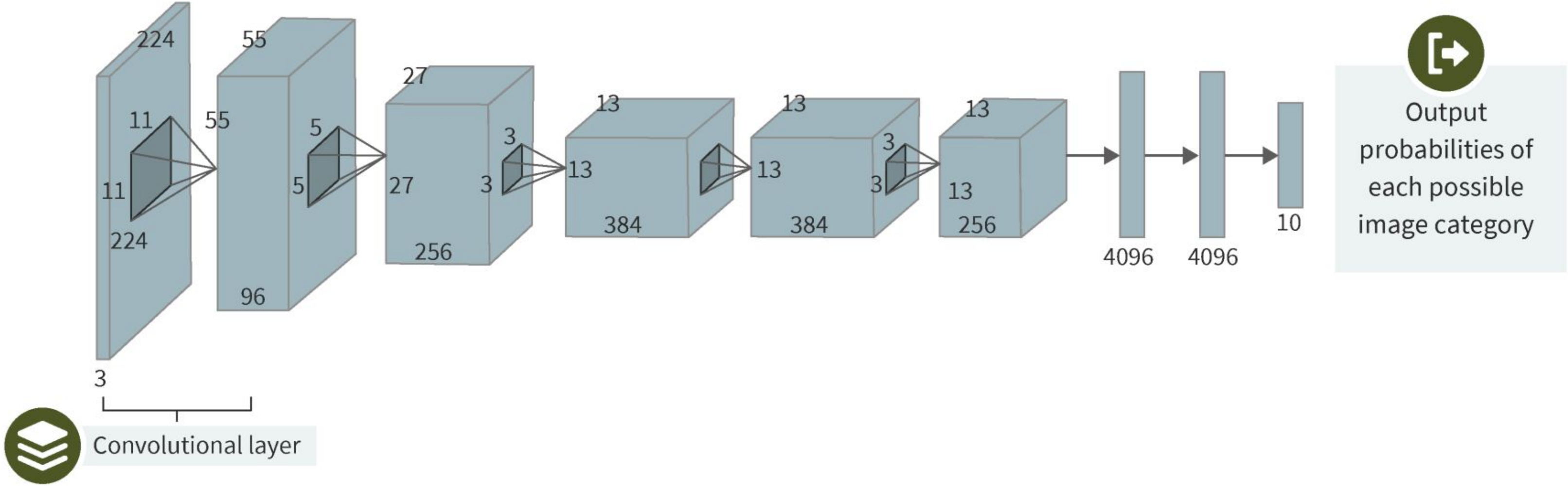
CONVOLUTIONAL NEURAL NETWORKS (CNNs)



Major advantages:

- Parameter efficient (filters “scan” the entire image for objects)

CONVOLUTIONAL NEURAL NETWORKS (CNNs)



Major advantages:

- Parameter efficient (filters “scan” the entire image for objects)
- Preserves spatial information (activations are the result of regional information)

Some case studies of convolutional neural networks...

Gulshan et al. 2016

- **Task**: Binary classification of referable diabetic retinopathy from **retinal fundus photographs**
- **Input**: Retinal fundus photographs
- **Output**: Binary classification of referable diabetic retinopathy (y in {0,1})
    - Defined as moderate and worse diabetic retinopathy, referable diabetic macular edema, or both

Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.

Gulshan et al. 2016

- **Dataset**:
  - 128,175 images, each graded by 3-7 ophthalmologists.
  - 54 total graders, each paid to grade between 20 to 62508 images.
- **Data preprocessing**:
  - Circular mask of each image was detected and rescaled to be 299 pixels wide
- **Model**:
  - Inception-v3 CNN, with ImageNet pre-training
  - Multiple binary cross-entropy losses corresponding to different binary prediction problems, which were then used for final determination of referable diabetic retinopathy

Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.

Gulshan et al. 2016

Pre-training means training first on a different (usually larger) dataset first to learn generally useful visual features as a starting point

- **Dataset**:
  - 128,175 images, each graded by 3-7 ophthalmologists.
  - 54 total graders, each paid to grade between 20 to 62508 images.
- **Data preprocessing**:
  - Circular mask of each image was detected and rescaled to be 299 pixels wide
- **Model**:
  - Inception-v3 CNN, with ImageNet pre-training
  - Multiple binary cross-entropy losses corresponding to different binary prediction problems, which were then used for final determination of referable diabetic retinopathy

Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.


IM GENET

Gulshan et al. 2016

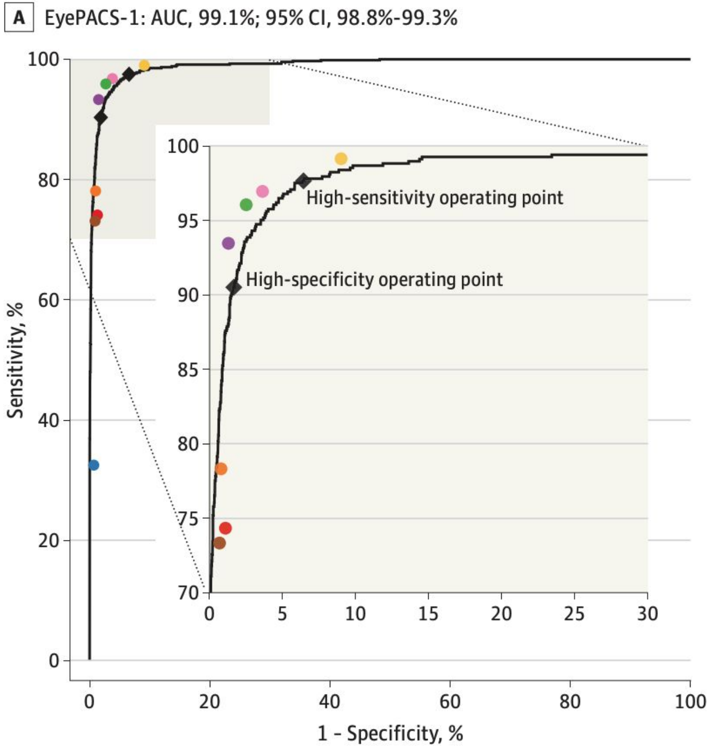
Graders provided finer-grained labels which were then consolidated into (easier) binary prediction problems

- **Dataset**:
  - 128,175 images, each graded by 3-7 ophthalmologists.
  - 54 total graders, each paid to grade between 20 to 62508 images.
- **Data preprocessing**:
  - Circular mask of each image was detected and rescaled to be 299 pixels wide
- **Model**:
  - Inception-v3 CNN, with ImageNet pre-training
  - Multiple binary cross-entropy losses corresponding to different binary prediction problems, which were then used for final determination of referable diabetic retinopathy

Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.

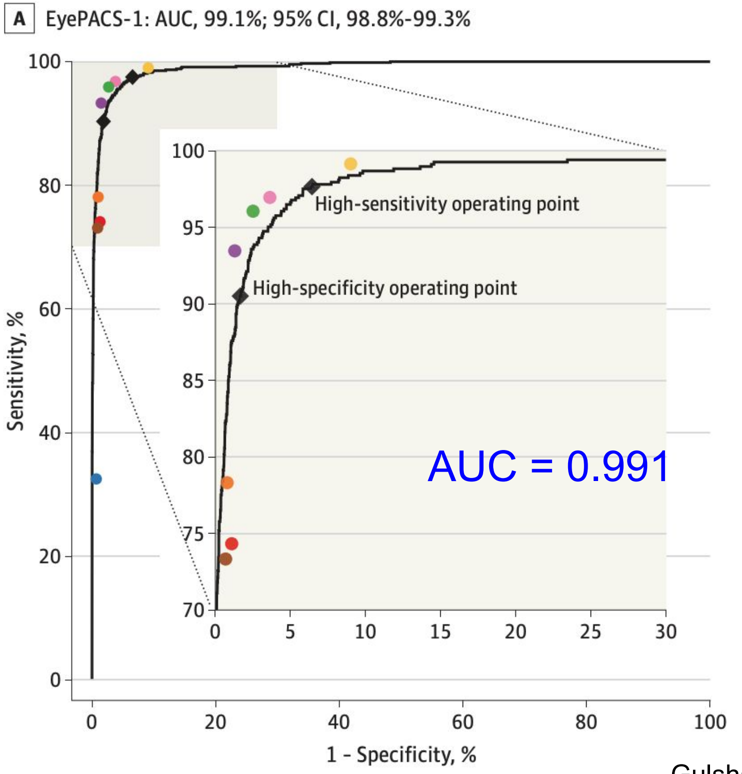
Gulshan et al. 2016

- **Results**:
    - Evaluated using ROC curves, AUC, sensitivity and specificity analysis



Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.

Gulshan et al. 2016

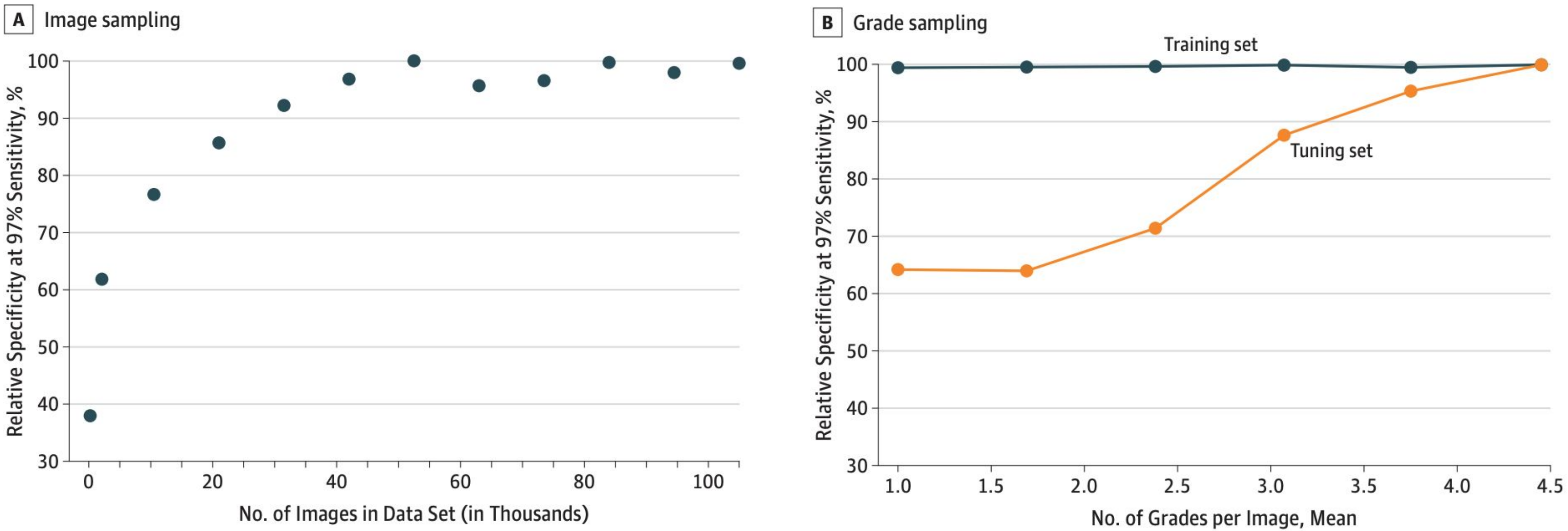


Looked at different operating points

- High-specificity point approximated ophthalmologist specificity for comparison. Should also use high-specificity to make decisions about high-risk actions.
- High-sensitivity point should be used for screening applications.

Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.

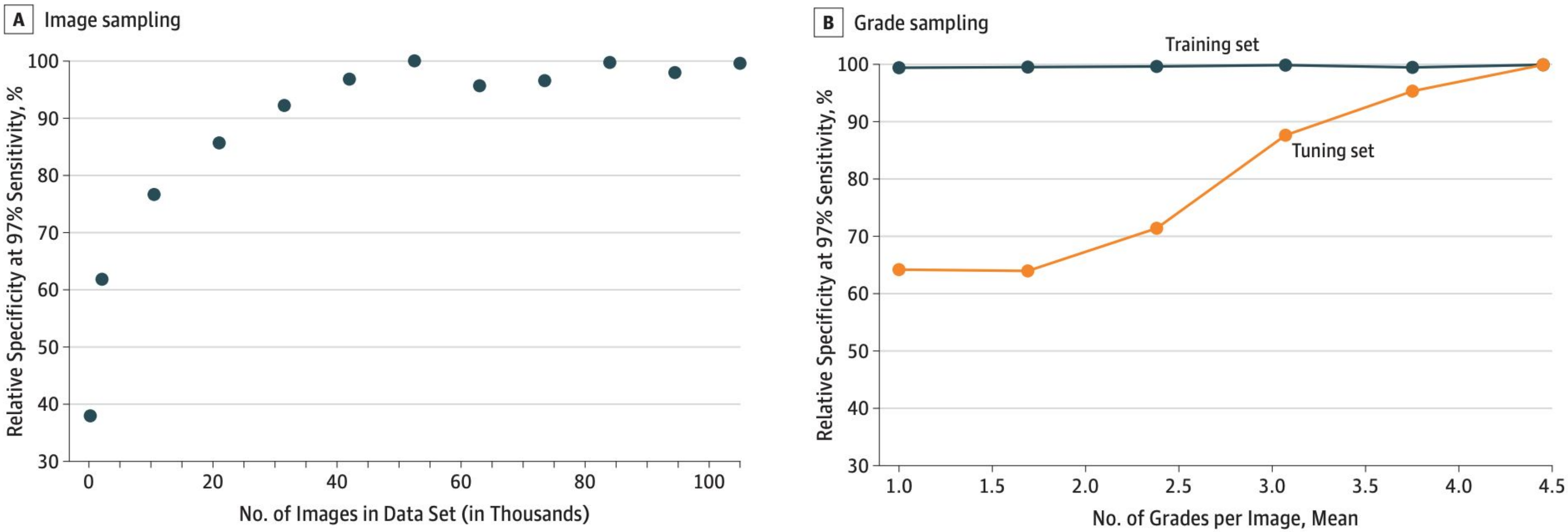
Gulshan et al. 2016



Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.

Gulshan et al. 2016

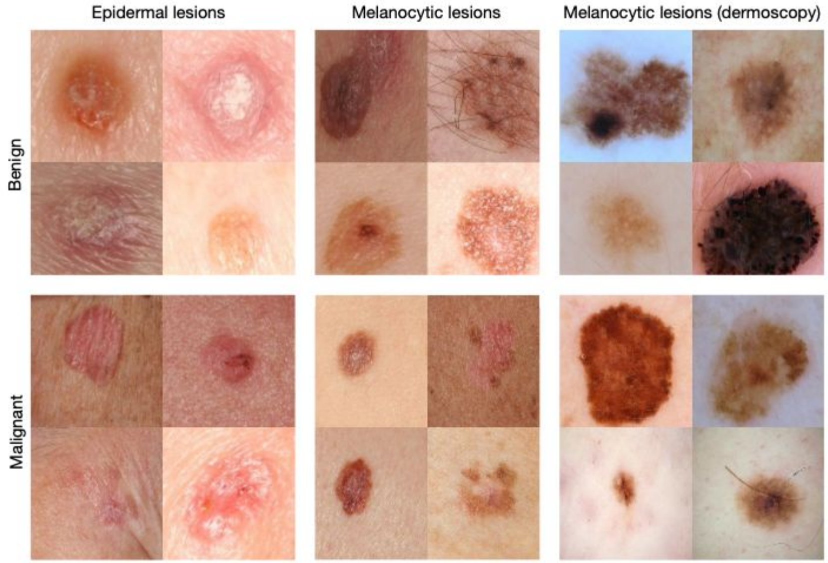
Q: What could explain the difference in trends for reducing # grades / image on training set vs. tuning set, on tuning set performance?



Gulshan, et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA, 2016.

Esteva et al. 2017

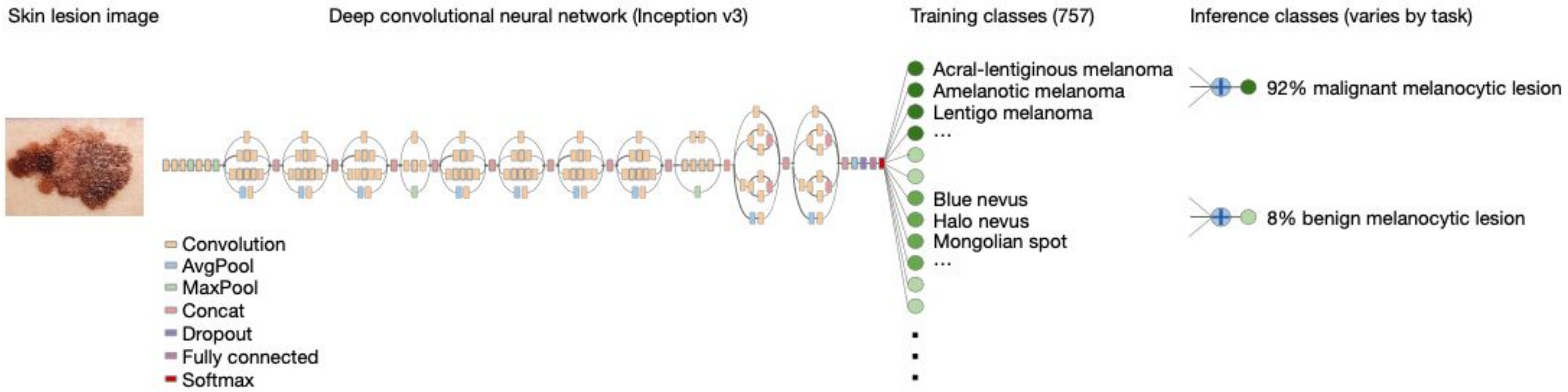
- Two binary classification tasks on **dermatology images**: malignant vs. benign lesions of epidermal or melanocytic origin
- Inception-v3 (GoogLeNet) CNN with ImageNet pre-training
- Fine-tuned on dataset of 129,450 lesions (from several sources) comprising 2,032 diseases
- Evaluated model vs. 21 or more dermatologists in various settings



Esteva*, Kuprel*, et al. Dermatologist-level classification of skin cancer with deep neural networks. Nature, 2017.

Esteva et al. 2017

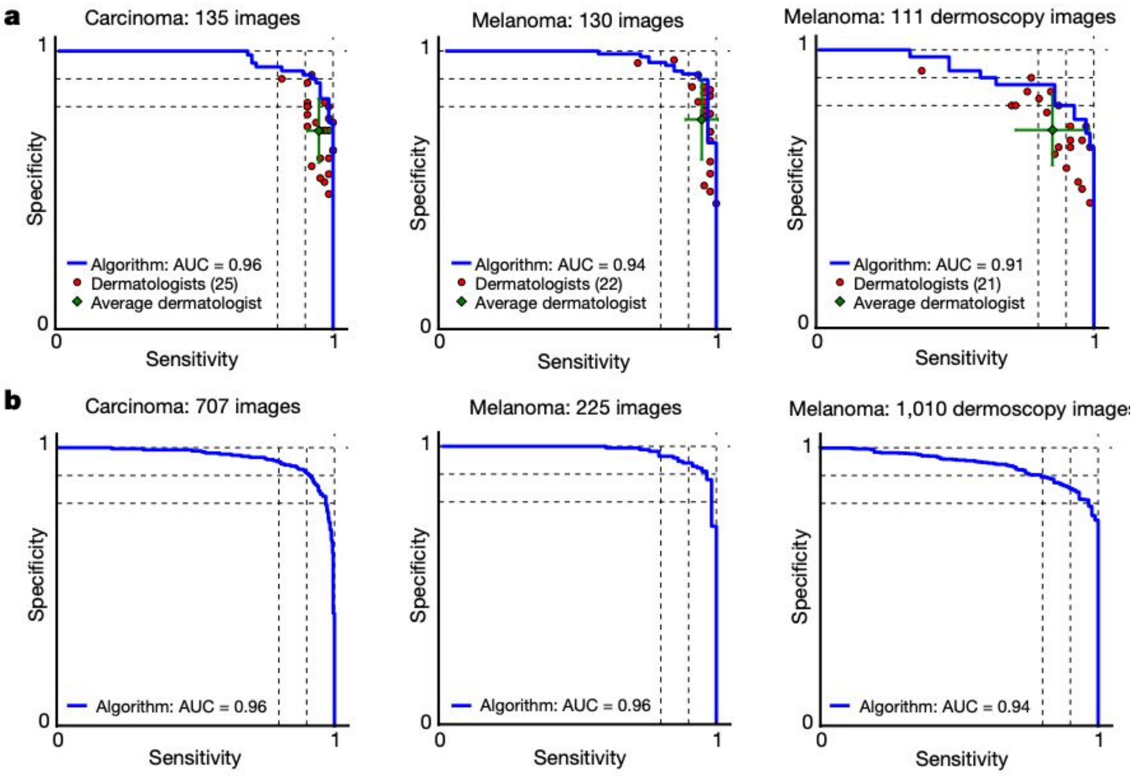
- Train on finer-grained classification (757 classes) but perform binary classification at inference time by summing probabilities of fine-grained sub-classes
- The stronger fine-grained supervision during the training stage improves inference performance!



Esteva*, Kuprel*, et al. Dermatologist-level classification of skin cancer with deep neural networks. Nature, 2017.

Esteva et al. 2017

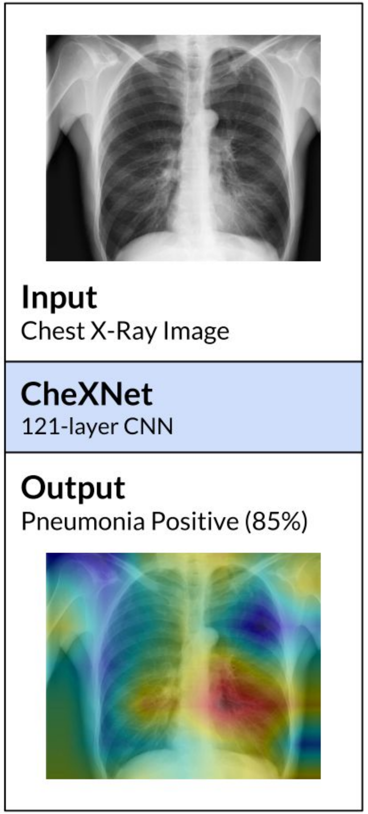
- Evaluation of algorithm vs. dermatologists



Esteva*, Kuprel*, et al. Dermatologist-level classification of skin cancer with deep neural networks. Nature, 2017.

Rajpurkar et al. 2017

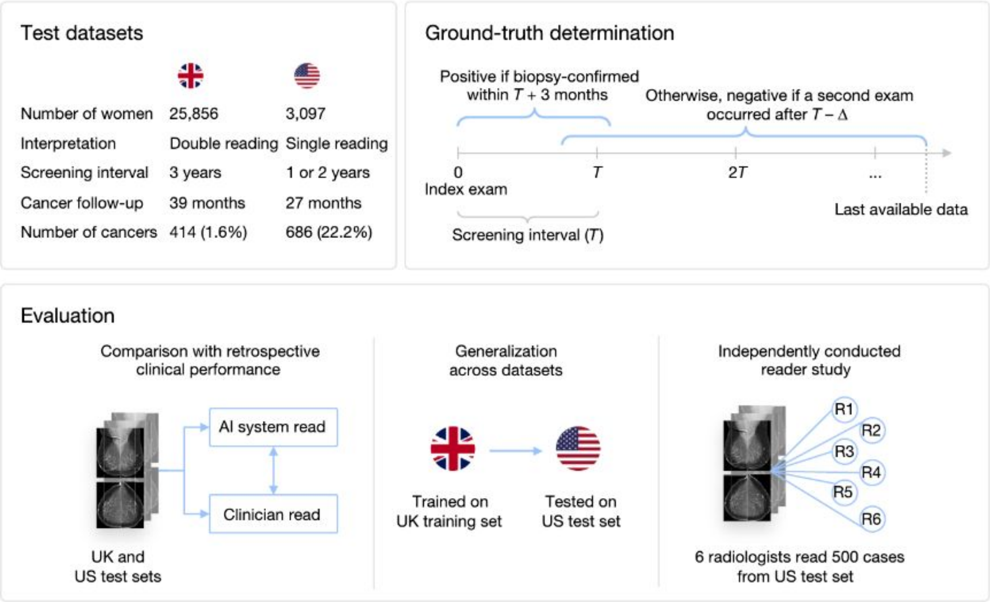
- Binary classification of pneumonia presence in chest **X-rays**
- Used ChestX-ray14 dataset with over 100,000 frontal X-ray images with 14 diseases
- 121-layer DenseNet CNN
- Compared algorithm performance with 4 radiologists
- Also applied algorithm to other diseases to surpass previous state-of-the-art on ChestX-ray14



Rajpurkar et al. CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning. 2017.

McKinney et al. 2020

- Binary classification of breast cancer in **mammograms**
- International dataset and evaluation, across UK and US



McKinney et al. International evaluation of an AI system for breast cancer screening. Nature, 2020.

Summary

Today we covered:

- Structure of neural network models
- Machine learning training loop and concept of *loss*, in the context of neural networks
- Minimizing the loss for complex neural networks: gradient descent and backpropagation
- Neural networks for a common type of input data: images (convolutional neural networks)

Next time: more on deep learning models for different types of input data and prediction tasks