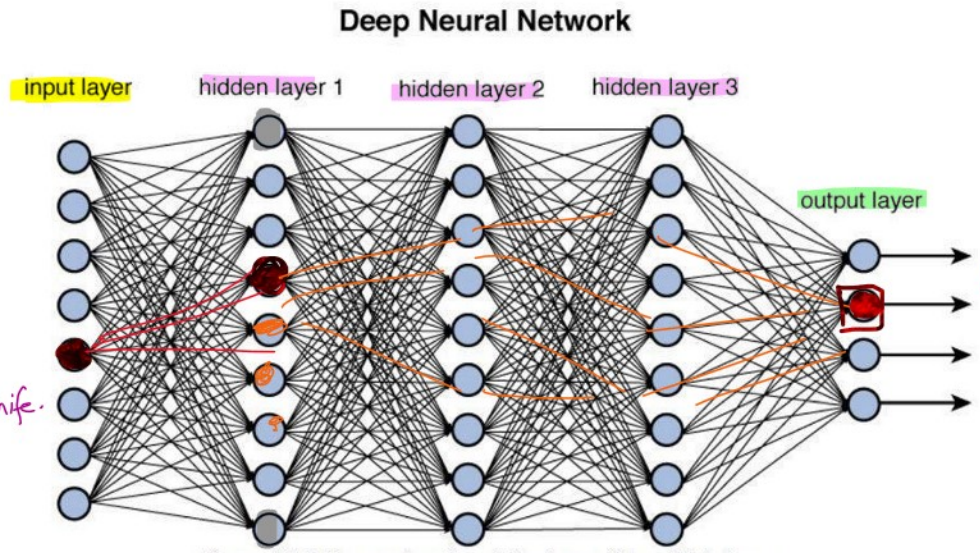
Understanding Neural Network Terminologies

24 November 2024 20:44



stabbing left arm with a knife.

'Pull away left arm'

Figure 12.2 Deep network architecture with multiple layers.

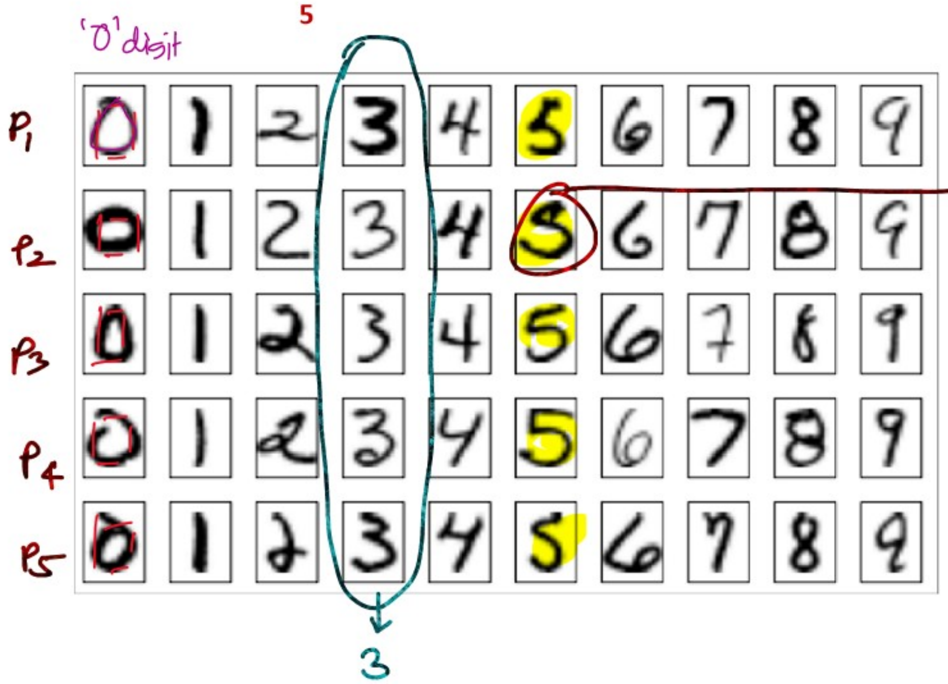
(looks like a cob-web.)

Handwritten Digits Recognition

0, 1, 2, 3, 4, 5, 6, 7, 8, 9 — APC

5

0 — 9

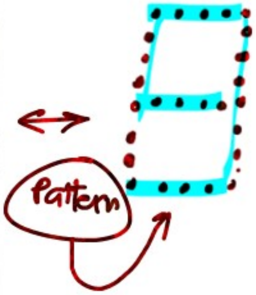


If model is trained to predict digit or alphabets.

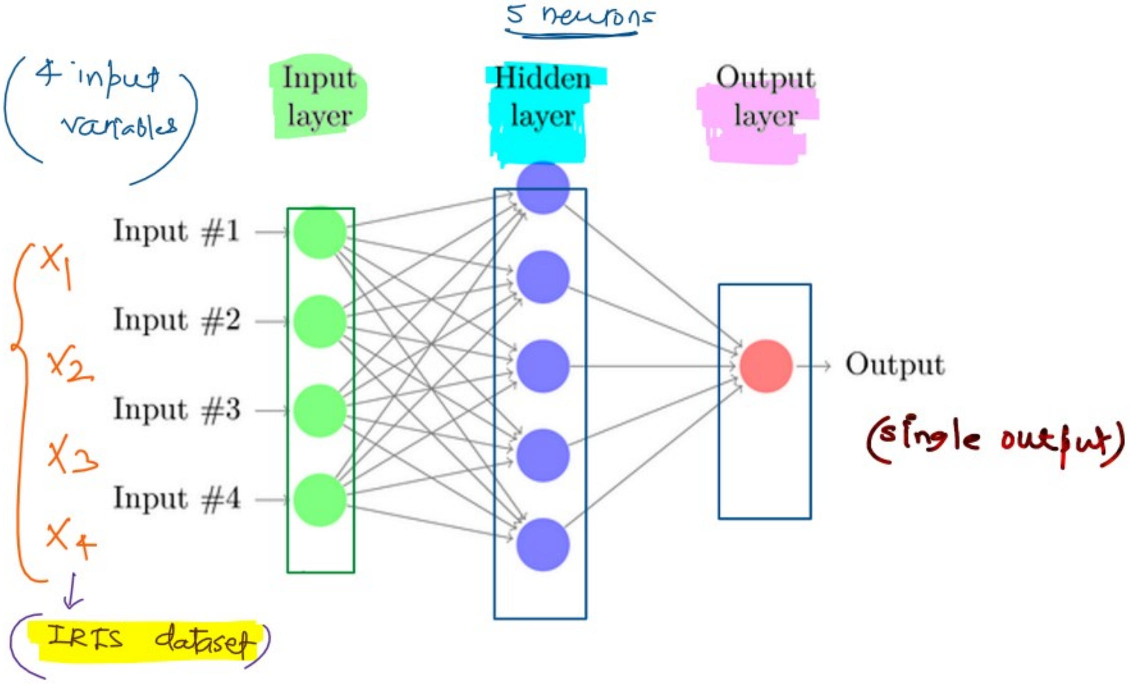
5 or s

so, in this particular example model might do the mistake and predict it as letter 's'.

→ Pattern

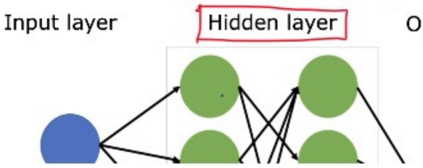


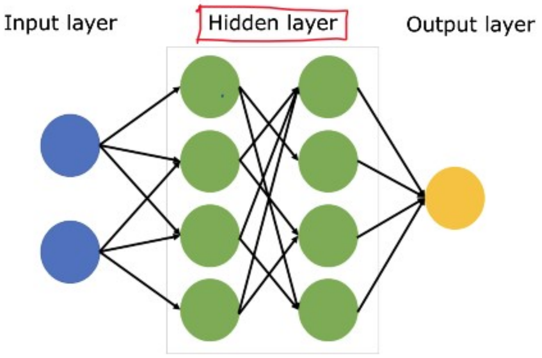
Neural Network Terminologies



2 hidden layers

NN can have quite a few hidden layers. Thanks to NVIDIA





Artificial neural networks

#① Input Layer

For any dataset → Training set → Neural N/w model
→ Testing set.

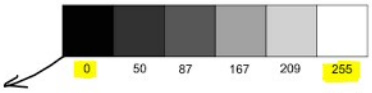
- To input the training data features into the neural network model, data source (input variables) ($X_1, X_2, - X_i$'s) is converted from raw images to pixelated matrex form (0–255)

↓ Handwritten digits recognition usecase.

↓ range for shades of black and white

0 - Black ■

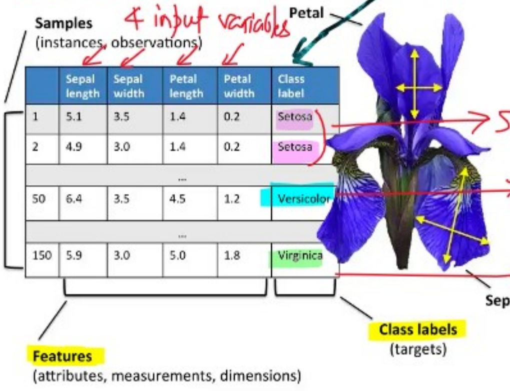
255 - White □



IRIS dataset

→ One of classification problems dataset

IRIS



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

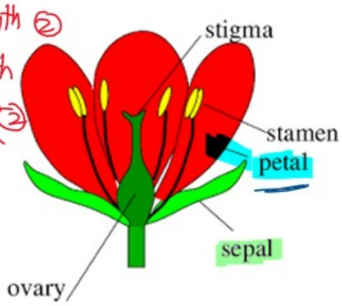
#150 rows

→ 50
→ 50
→ 50

(Balanced dataset)

sepal → length ②
→ width

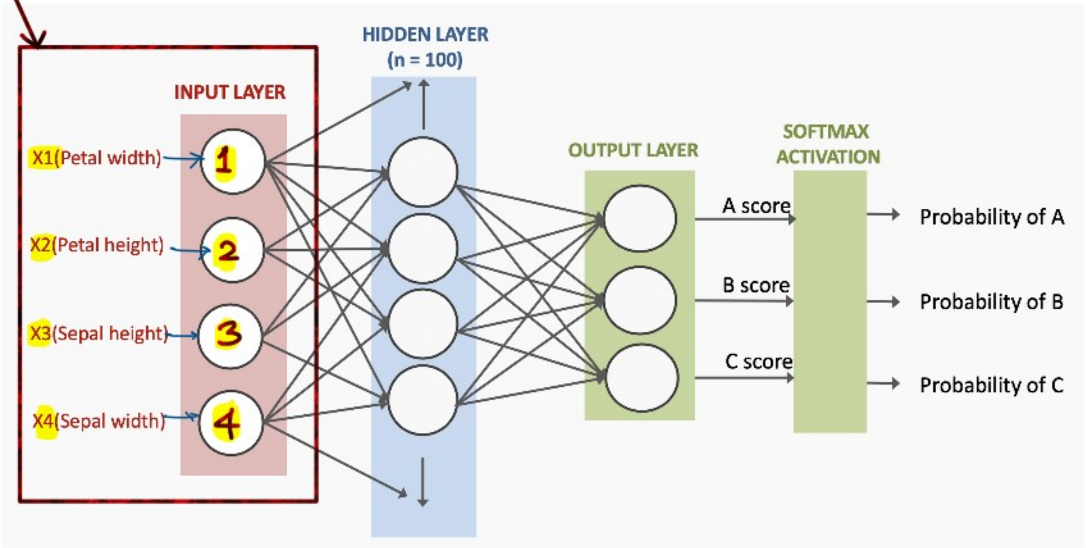
petal → length ②
→ width



✓ ✓ ✓

- Input layer is responsible to accept the data and pass it to the rest of the network.

Focus pls



Each neuron in the input layer represents one feature (x_i) of the input data (training data)

* No learnable parameters: → [weights, biases]

Unlike hidden or output layers, the input layer does not have weights or biases.

↓

No calculation is being done

↓

(input layer simply passes the input data to the next layer.)

In general, No of neurons in input layer = No. of features in the input data

In general, No of neurons in input layer = No. of features in the input data
(training dataset)

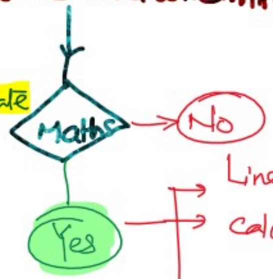
#② Hidden Layer

- This is the second layer in neural network architecture diagram
- Hidden layer(s) can be one or more than one.

* Common perception: It is a black box!!!

Will open it → going to be overwhelming!!!

- Hidden layer is the intermediate layer b/w input and output layer.



→ Linear Algebra Implementation using Numpy
→ Calculus — Gradient Descent Algorithm.
a) Maxima & Minima
b) Partial Derivative
c) Differentiation Chain Rule.

(Share Khan Academy)
→ Probabilities (log of odds)
+
Logistic Regression

Purpose of Hidden Layer(s)

Hidden layer is the critical layer where most of the computation happens, allowing the model to learn representations and patterns from the data.

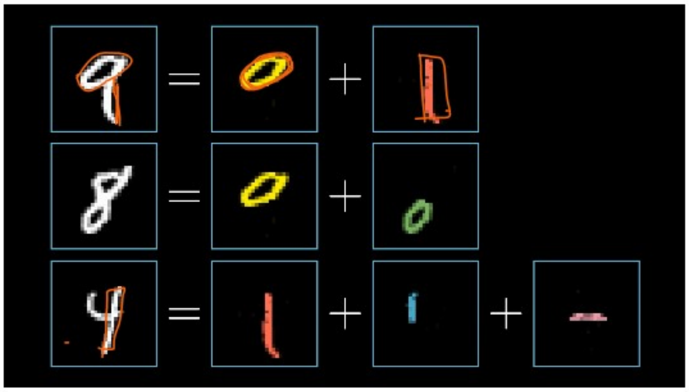
Feature Extraction

→ Hidden layers identify important features or patterns in the data which are not explicitly visible in the raw input.

→ Each succesive hidden layer learns increasingly complex features

→ edges in images in early layers, followed by shapes and then objects in later layers step#① edges and corners

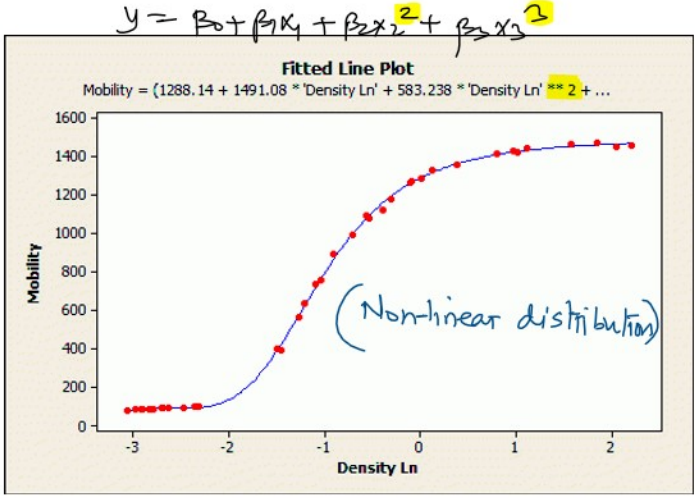
step#② Combining these features identified in step#1 to detect shapes like line, circle, semicircle, squares etc

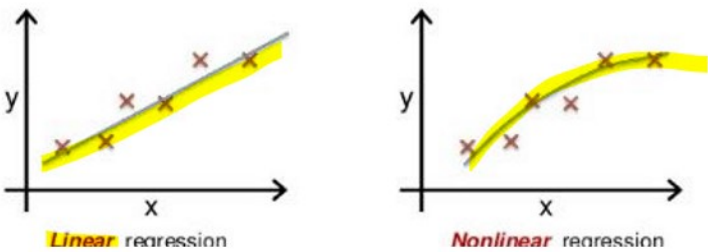


Non-Linearity

Hidden layers enable the network to model complex, non-linear relationships b/w inputs and outputs by applying non-linear activation functions

$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2^{2} + \beta_3 x_3^{3}$





Linear regression

Nonlinear regression

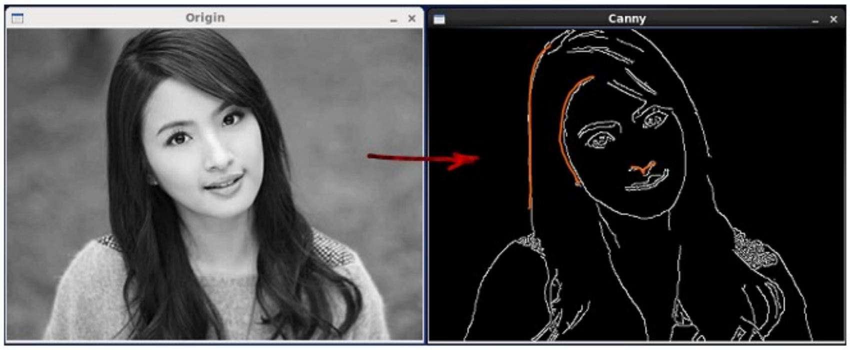
Hierarchical Representation

Hidden layers build a hierarchy of representations:

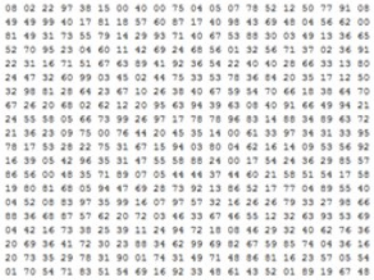
- Lower layers learn simple patterns (e.g. edges in images)
- Higher layers learn abstract concepts (e.g., objects or semantics)

↓
car/airplane
(CNN)

will come later!

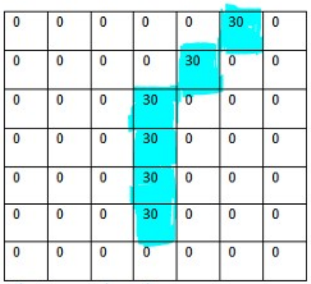


What We See



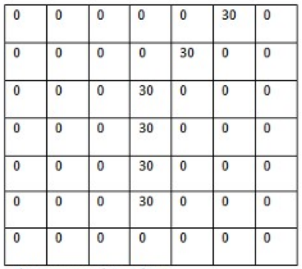
```
08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00
81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65
52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91
22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80
24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50
32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70
67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 21
24 55 58 05 66 73 99 26 97 17 78 78 96 83 14 88 34 89 63 72
21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95
78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57
86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58
19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40
04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66
88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69
04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36
20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16
20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54
01 70 54 71 83 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48
```

What Computers See



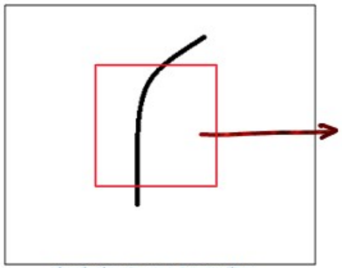
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter



0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter



'detected an edge'

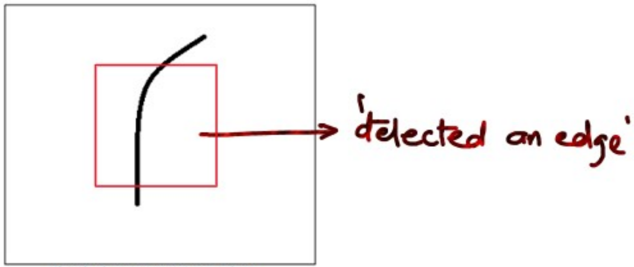
Visualization of a curve detector filter

0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

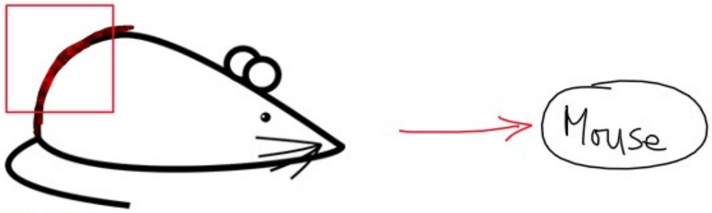
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter



Visualization of a curve detector filter

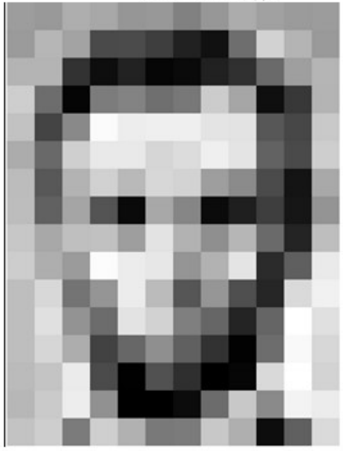
'detected an edge'



Original image

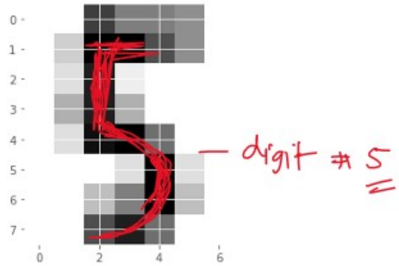
Mouse

Abraham Lincoln



(0–255 range)

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

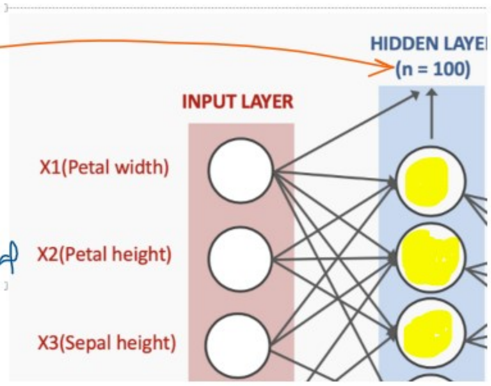


— digit # 5

Structure of Hidden Layer(s)

a) Neurons

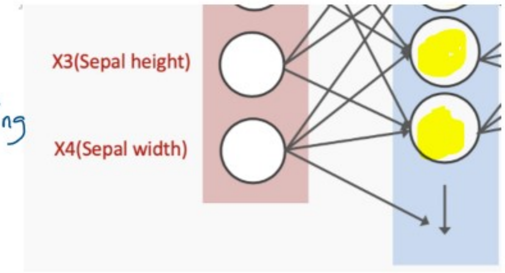
- Each hidden layer consists of multiple neurons(ex: 100 neurons) which are computational units.



- Each neuron h

units.

- Each neuron processes inputs by performing σ(weighted sum + bias) followed by activation function



$$\boxed{z = \sum_{i=1}^{n} w_i x_i + b}$$ → weighted sum of inputs with bias added to it.

where:

w_i: weights

x_i: inputs

b: bias

$\sum$: sigma: summation

Linear Regression

weight | coefficients and bias(intercept)