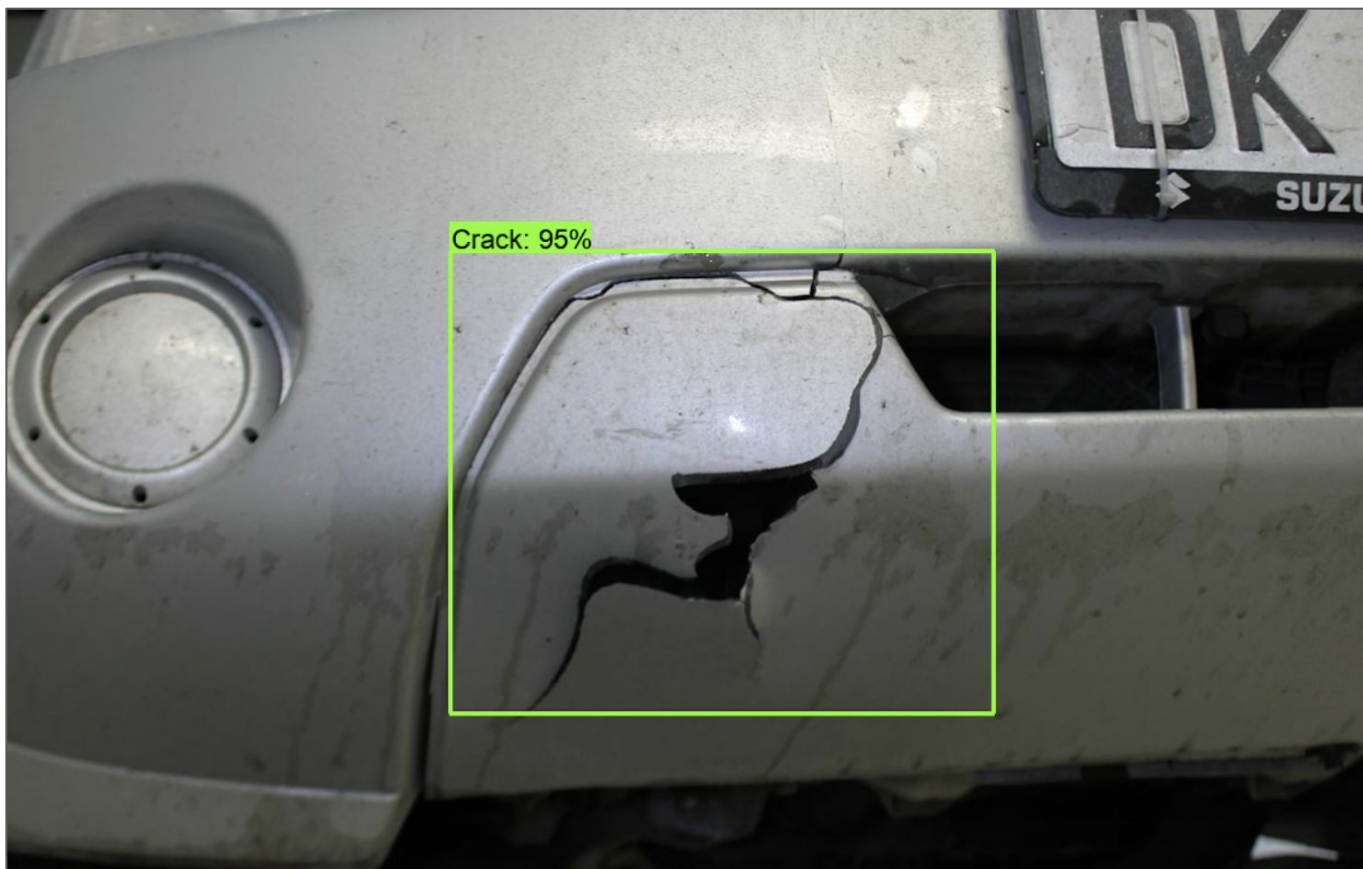
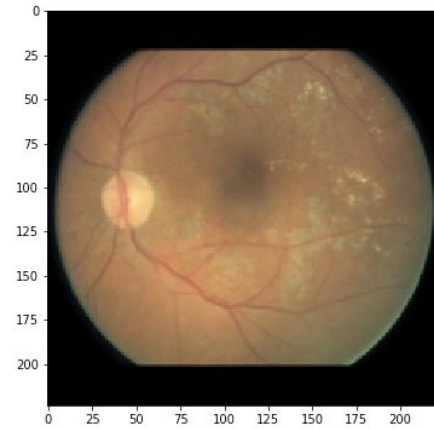
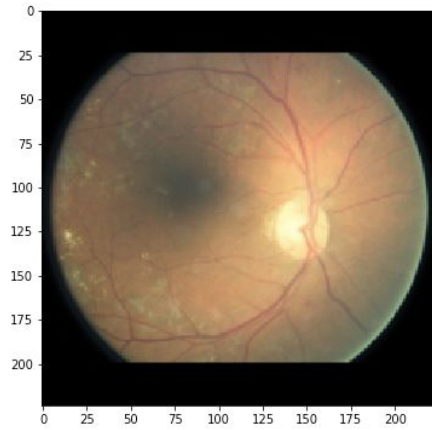
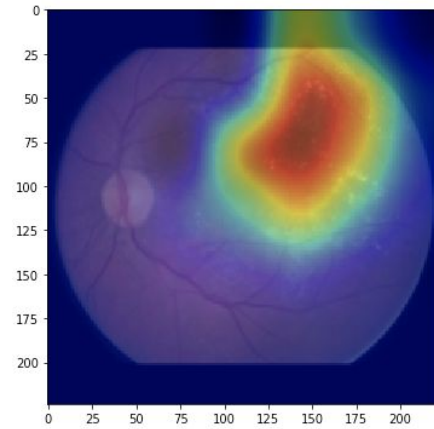
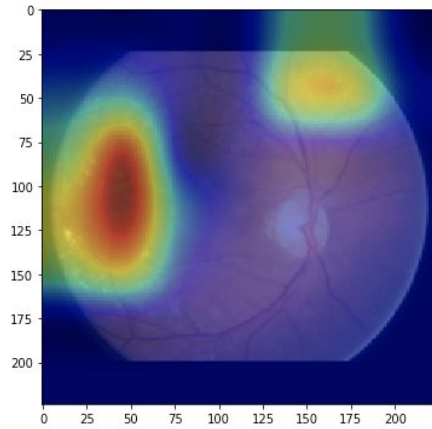
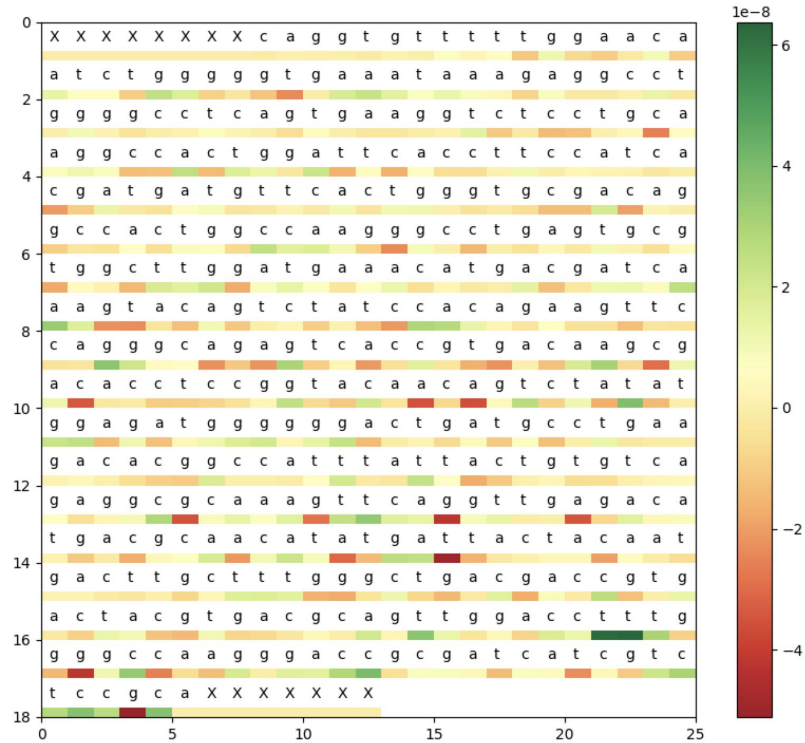


Machine learning in Python





HIV == HIV





Thank you

I really appreciate
the help :)

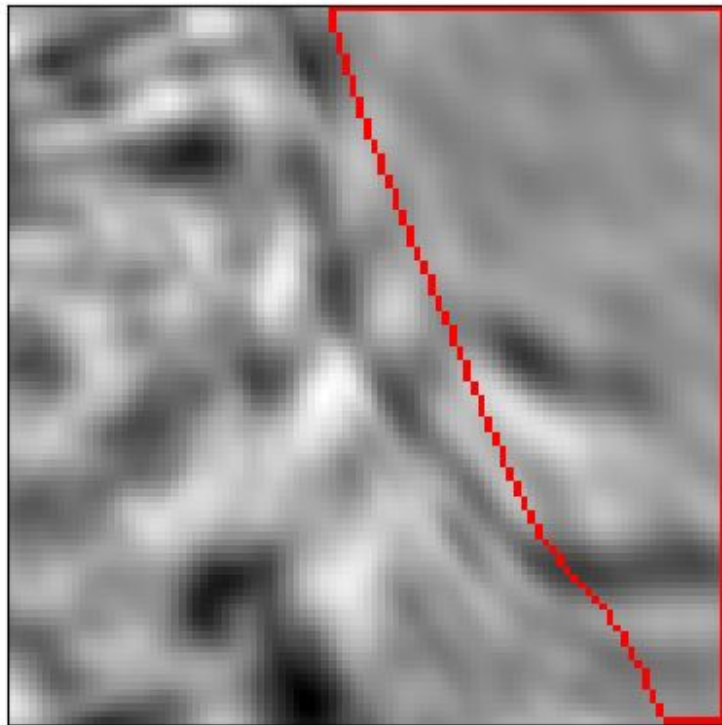
Amazing!!



What the f**k is
going on?!?

Thanks for
nothing. What a
horrible user
experience





01. Algorithms

$2 \Rightarrow \{2\}$

$3 \Rightarrow \{3\}$

$4 \Rightarrow \{2, 2\}$

$5 \Rightarrow \{5\}$

$6 \Rightarrow \{2, 3\}$

$7 \Rightarrow \{7\}$

$8 \Rightarrow \{2, 2, 2\}$

01. Algorithms

2 \Rightarrow {2}
3 \Rightarrow {3}
4 \Rightarrow {2, 2}
5 \Rightarrow {5}
6 \Rightarrow {2, 3}
7 \Rightarrow {7}
8 \Rightarrow {2, 2, 2}



```
input: positive integer  $N$ 
output: non-trivial factor of  $N$ 

Choose bound  $B$ 
Let  $P := \{p_1, p_2, \dots, p_k\}$  be all primes  $\leq B$ 

repeat
  for  $i = 1$  to  $k + 1$ 
    Choose  $0 < z_i < N$  such that  $z_i^2 \bmod N$  is  $B$ -smooth
    Let  $a_i := \{a_{i1}, a_{i2}, \dots, a_{ik}\}$  such that  $z_i^2 \bmod N = \prod_{p_j \in P} p_j^{a_{ij}}$ 
  end for

  Find non-empty  $T \subseteq \{1, 2, \dots, k + 1\}$  such that  $\sum_{i \in T} a_i \equiv \vec{0} \pmod{2}$ 
  Let  $x := \left( \prod_{i \in T} a_i \right) \bmod N$ 
   $y := \left( \prod_{p_j \in P} p_j^{(\sum_{i \in T} a_{ij})/2} \right) \bmod N$ 
while  $x \equiv \pm y \pmod{N}$ 

return  $\gcd(x + y, n)$ 
```

01. Algorithms

2 \Rightarrow {2}
3 \Rightarrow {3}
4 \Rightarrow {2, 2}
5 \Rightarrow {5}
6 \Rightarrow {2, 3}
7 \Rightarrow {7}
8 \Rightarrow {2, 2, 2}

```
input: positive integer  $N$ 
output: non-trivial factor of  $N$ 

Choose bound  $B$ 
Let  $P := \{p_1, p_2, \dots, p_k\}$  be all primes  $\leq B$ 

repeat
  for  $i = 1$  to  $k + 1$ 
    Choose  $0 < z_i < N$  such that  $z_i^2 \bmod N$  is  $B$ -smooth
    Let  $a_i := \{a_{i1}, a_{i2}, \dots, a_{ik}\}$  such that  $z_i^2 \bmod N = \prod_{p_j \in P} p_j^{a_{ij}}$ 
  end for

  Find non-empty  $T \subseteq \{1, 2, \dots, k + 1\}$  such that  $\sum_{i \in T} a_i \equiv \vec{0} \pmod{2}$ 
  Let  $x := \left( \prod_{i \in T} a_i \right) \bmod N$ 
   $y := \left( \prod_{p_j \in P} p_j^{(\sum_{i \in T} a_{ij})/2} \right) \bmod N$ 
while  $x \equiv \pm y \pmod{N}$ 

return gcd( $x + y, n$ )
```

Q.E.D

02. Machine Learning

2 => {2}

3 => {7}

4 => {}

5 => {2, 2, 2, 2, 2, ...}

6 => {"airplane"}

7 => {🐼, □}

8 => {ó}

02. Machine Learning

2 => {2}

3 => {7}

4 => {}

5 => {2, 2, 2, 2, 2, ...} →


6 => {"airplane"}


7 => {🐱, □}


8 => {ó}


?


03. Linear regression

 Mulighetenes marked

 Varslinger

 Ny annonse



 Meldinger

 Min FINN

Eiendom / Bolig til salgs

527 treff i 202 annonser

Lagre søk


  Kart

Sortér på

Publisert


Publisert
☐ Nye i dag (68)

Område, by eller sted
☐ Søk etter adresse eller sted




Norge
Faroe Islands
Finland
Leafløt | © Norkart

Område
☐ Akershus (0)
☐ Aust-Agder (0)
☐ Buskerud (0)
☐ Finnmark (0)
☐ Hedmark (0)
☐ Hordaland (0)
☐ Møre og Romsdal (0)
☐ Nordland (0)



Ukens bolig

Visning i dag





Markveien 1 A, Oslo, Grünerløkka - Sofienberg


ROLIG PÅ ØVRE GRÜNERLØKKA: Lys og lekkert 3-roms loftsleilighet med privat solrik takterrasse og felles takterrasse. -Idyllisk bakgård -Hems -

78 m² 6 200 000,-

Andel • Leilighet • 2 soverom
Schala & Partners Grünerløkka
Fellesgjeld: 28 064,- • Fellesutg.: 5 226,-









Sverdrups gate 10A, Oslo, Grünerløkka - Sofienberg

GRÜNERLØKKA - Unik 2-roms selveier over tre plan i sjarmerende omgjort stall. Terrasse, romslig hems og herlig town-


42 m² 3 400 000,-

DNB Eiendom AS
Eier (Selveier) • Leilighet • 1 soverom
Fellesgjeld: 65 000,- • Fellesutg.: 2 215,-





Visning i dag



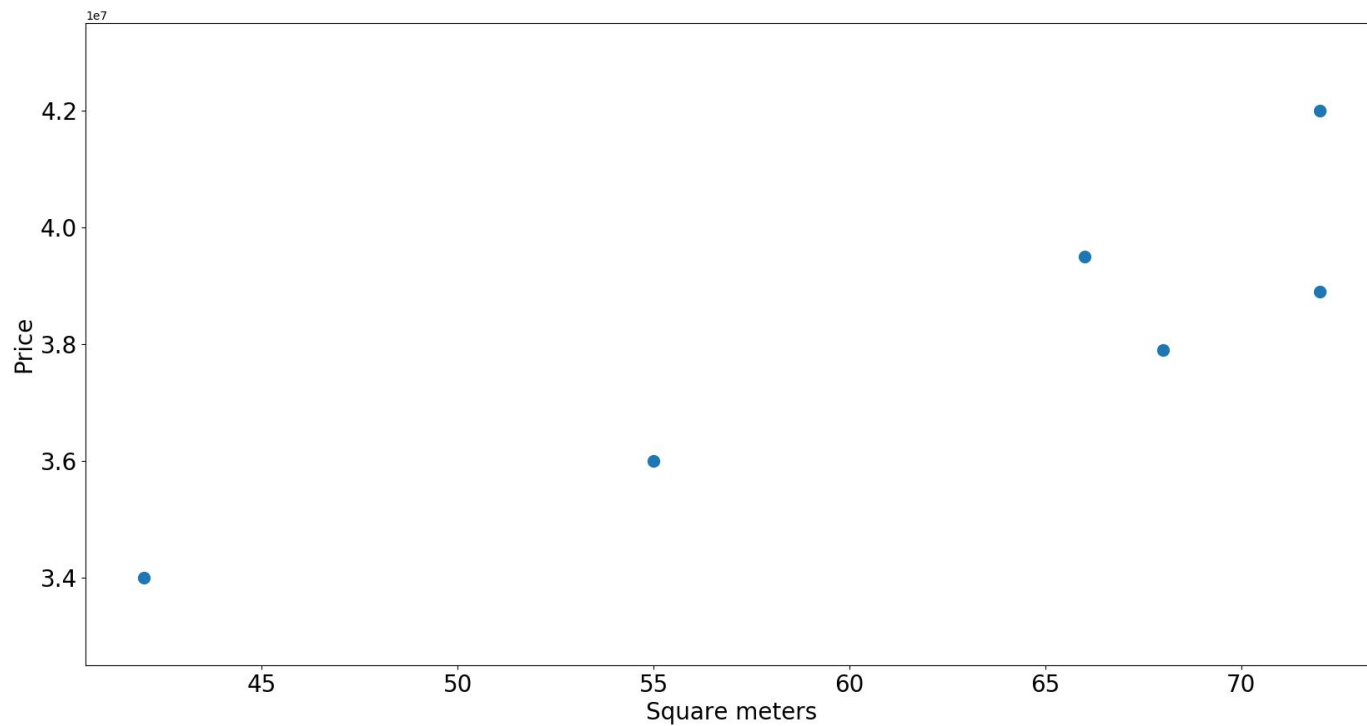
Lørenveien 57-63, leil. B2-35, Oslo

Fastpris: Lørenporten - Ny 2R m/ca. 15 kvm vestvendt terrasse. 31.12.2020 tas det sikte på overtakelse.

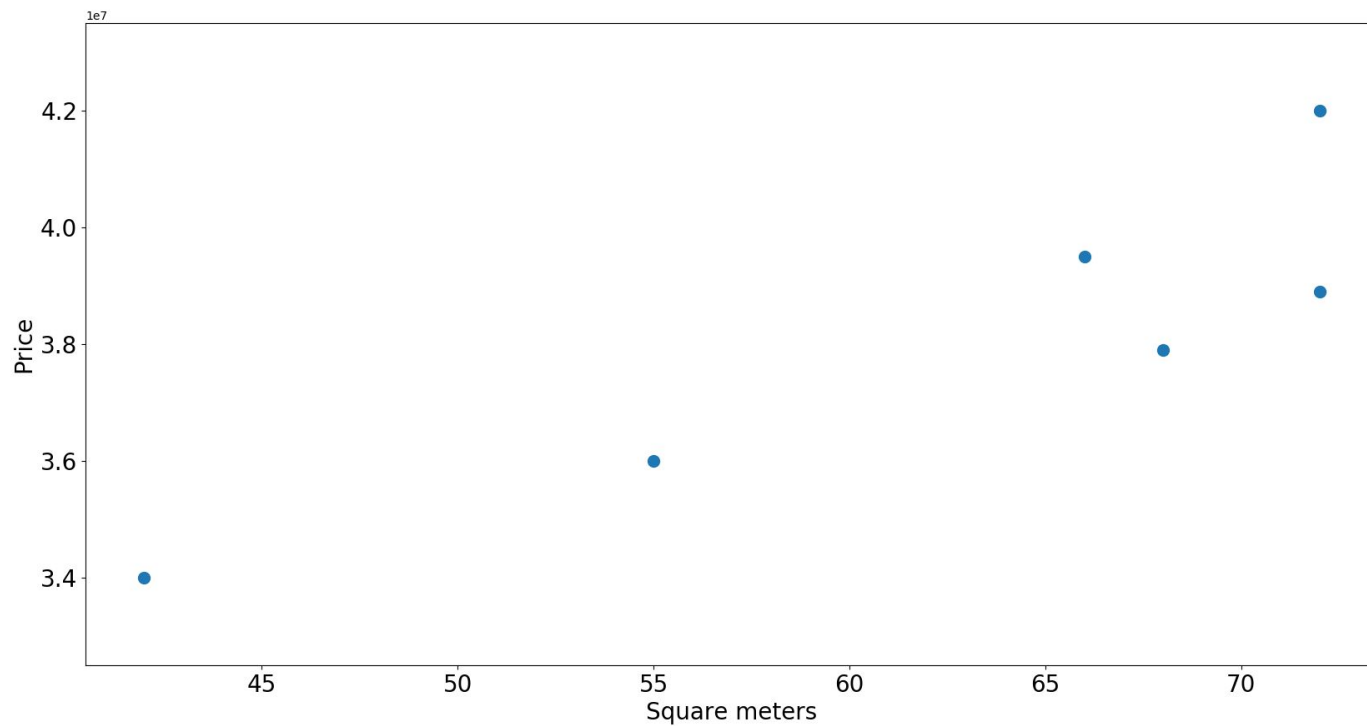
03. Linear regression

<u>m²</u>	<u>NOK</u>
72	4 200 000
68	3 790 000
72	3 890 000
42	3 400 000
66	4 950 000
55	3 600 000

03. Linear regression

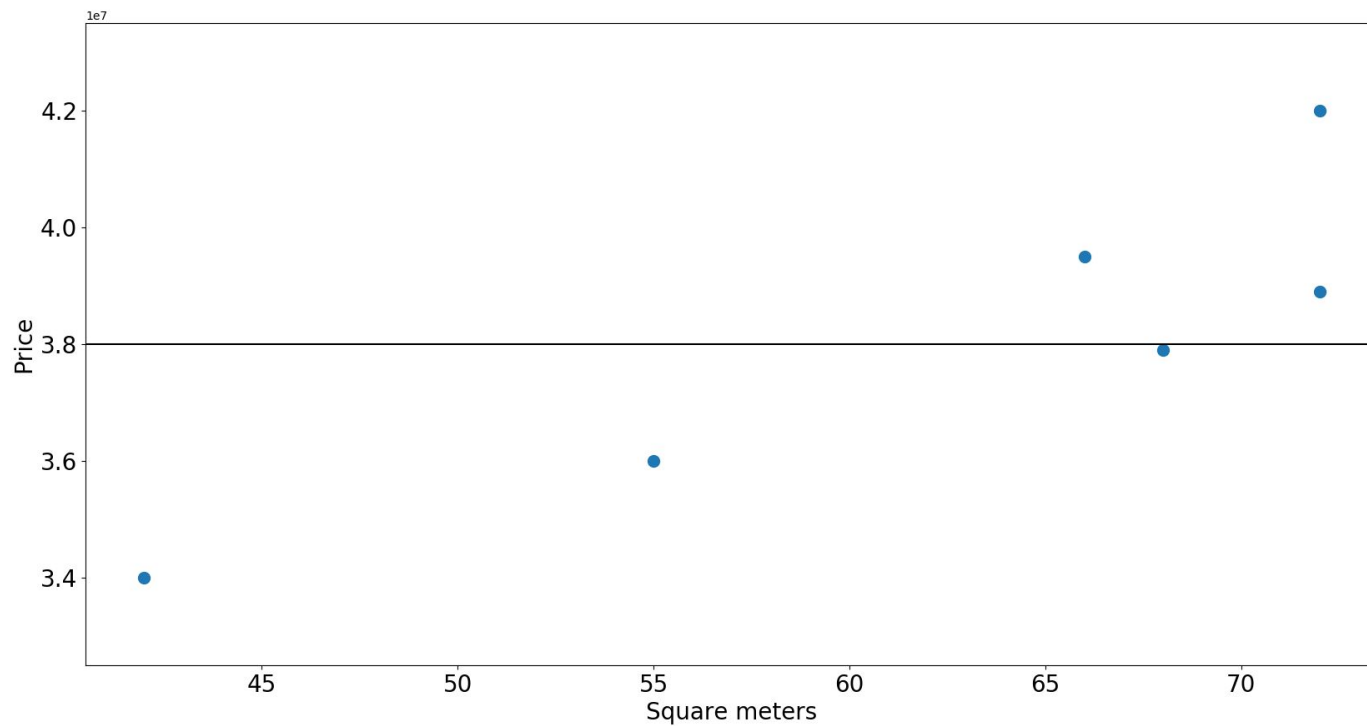


03. Linear regression



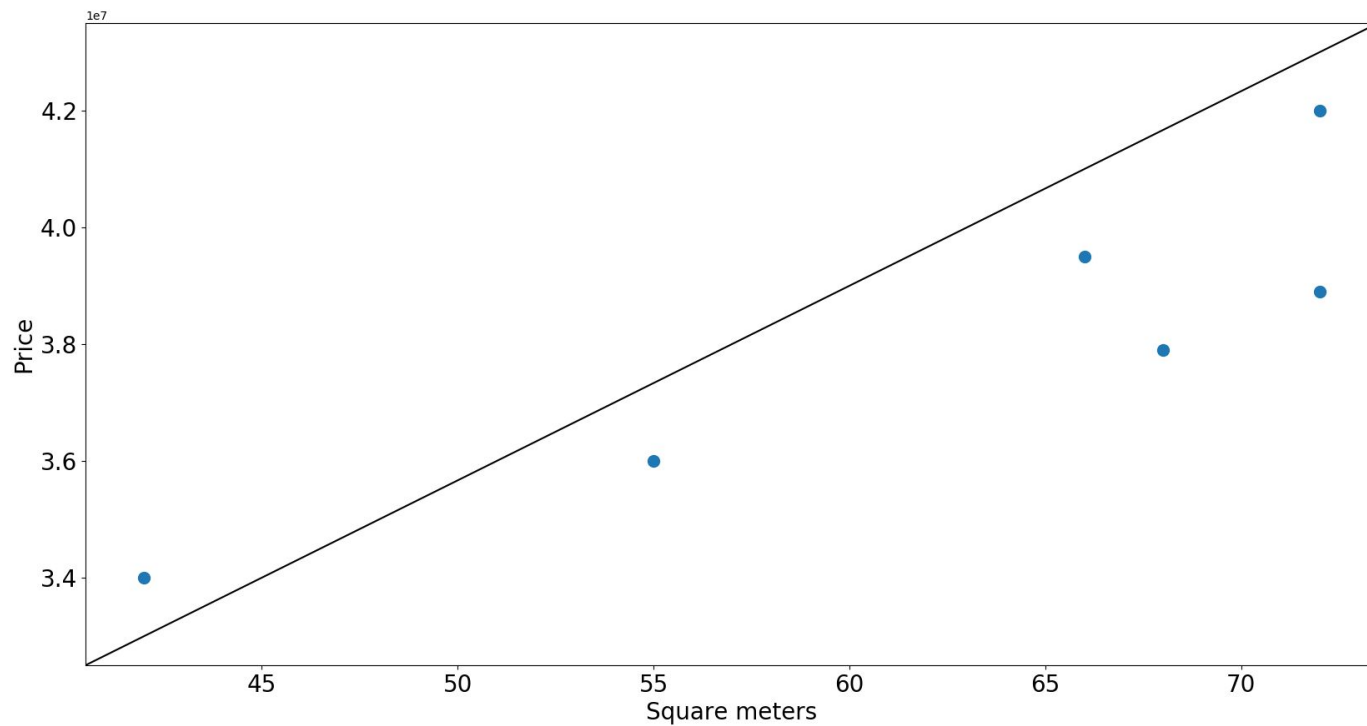
$$f(x) = ax+b$$

03. Linear regression

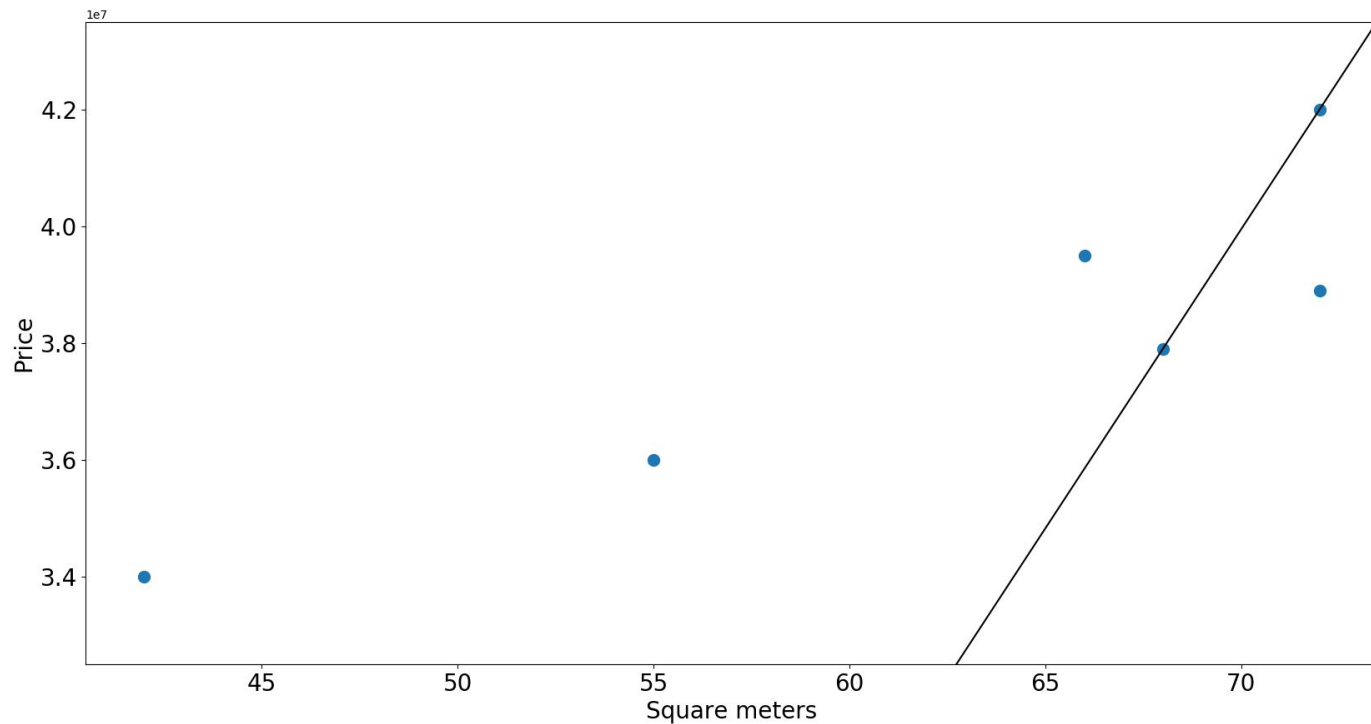


$$f(x) = 0x + 3800000$$

03. Linear regression

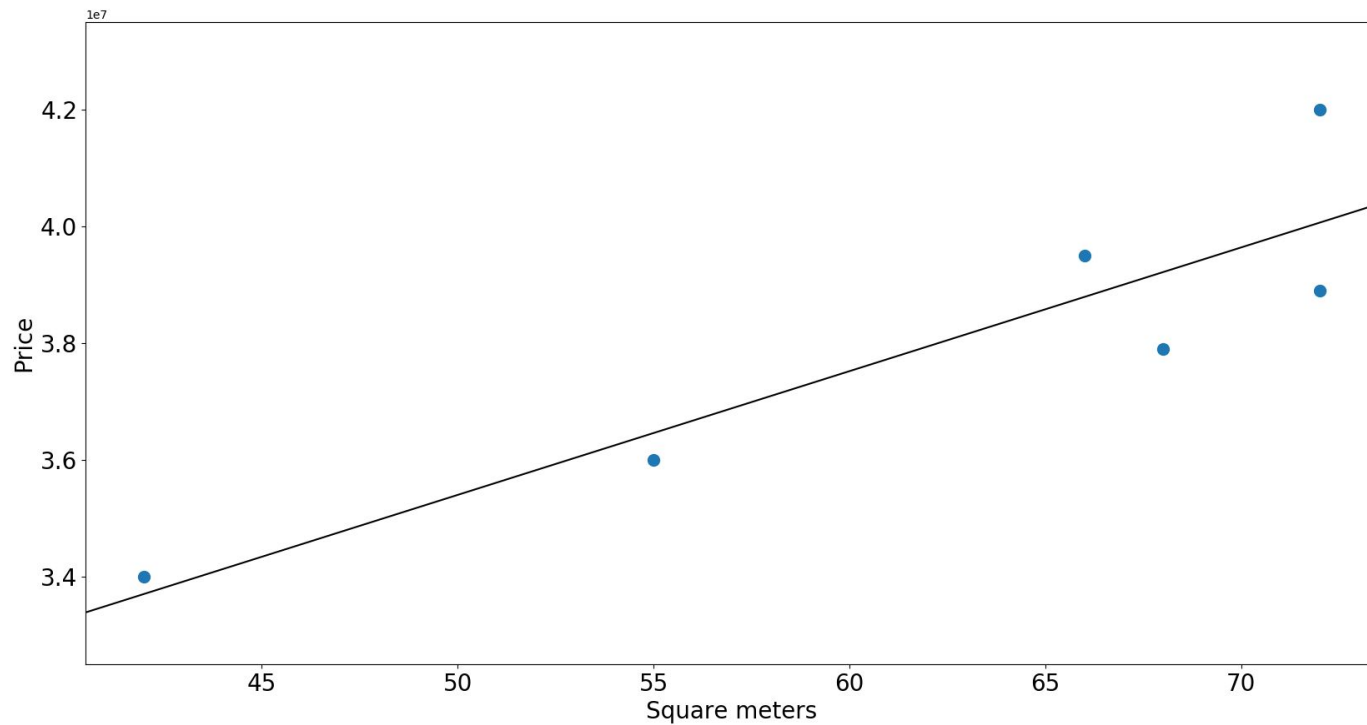


03. Linear regression



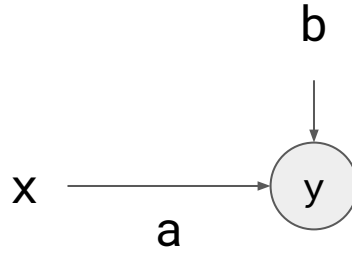
$$\text{cost}(x, f(x)) = \begin{cases} 0 & \text{if } f(x) = x \\ 1 & \end{cases}$$

03. Linear regression



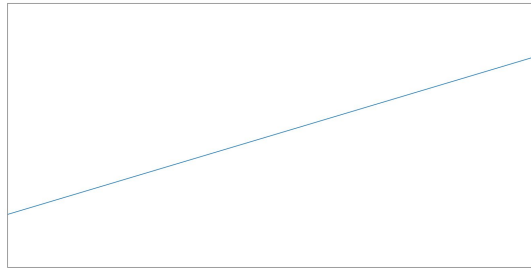
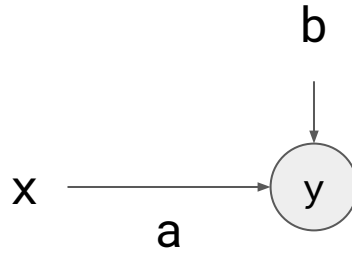
$$\text{cost}(x, f(x)) = (f(x) - x)^2$$

04. Neural nets

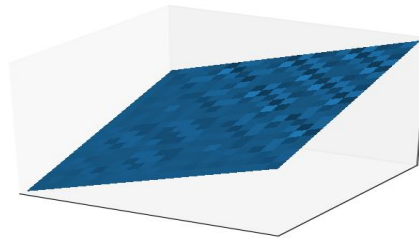
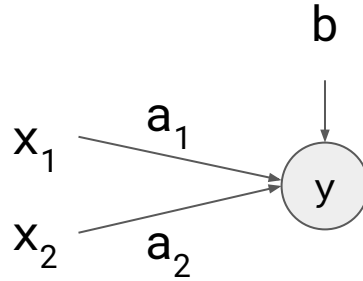


$$f(x) = ax+b$$

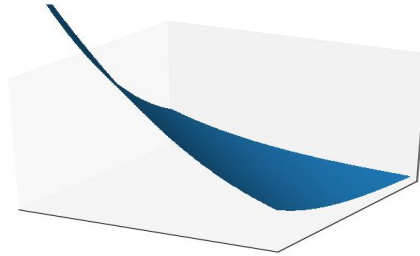
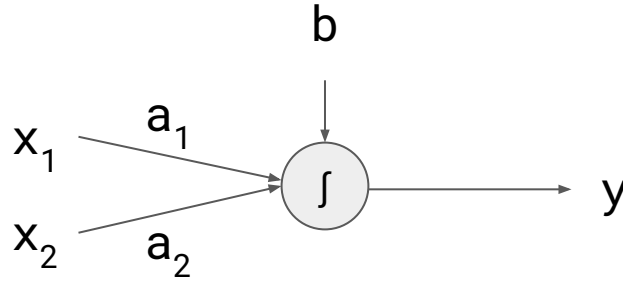
04. Neural nets



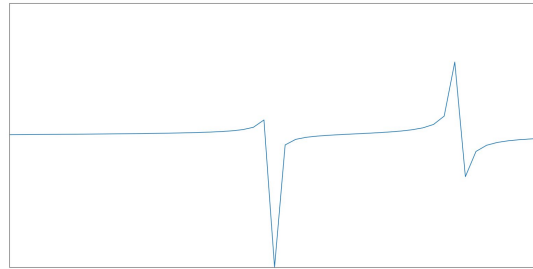
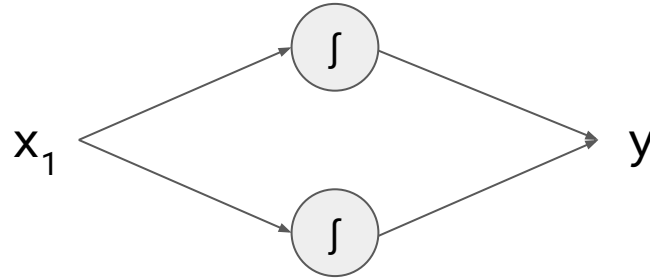
04. Neural nets



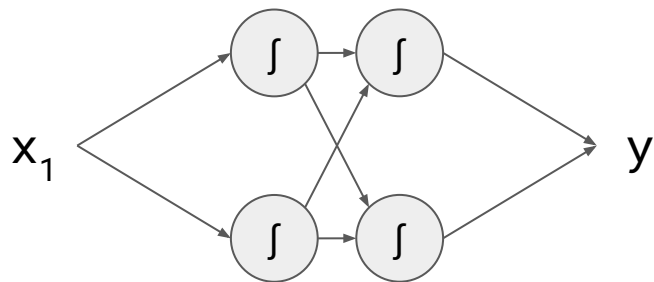
04. Neural nets



04. Neural nets

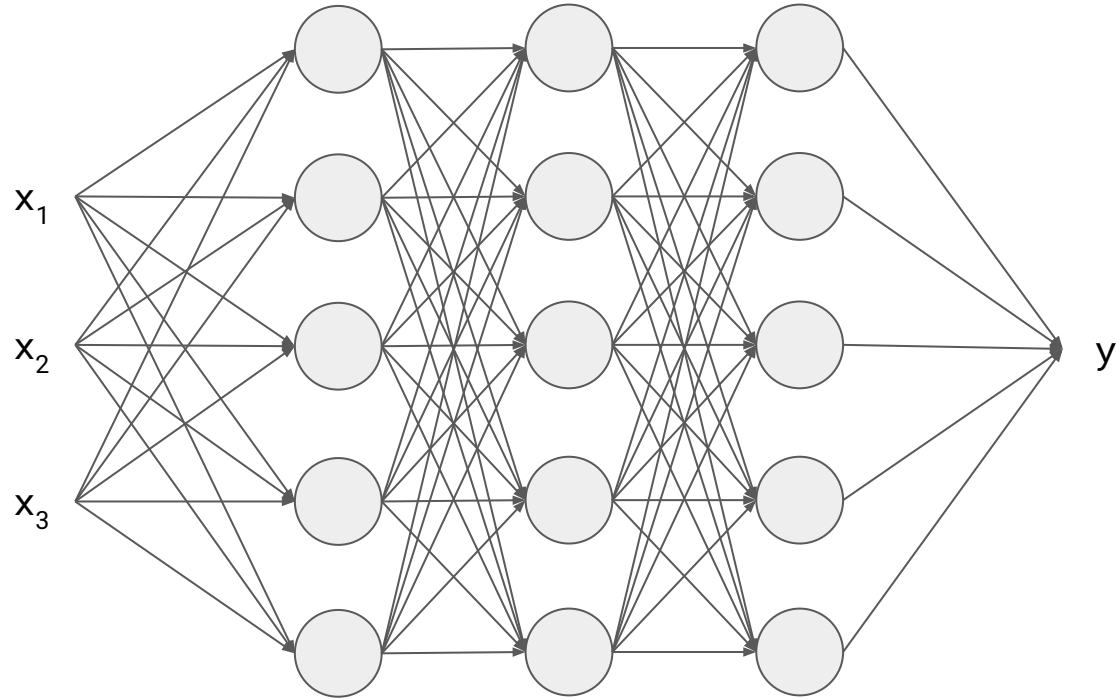


04. Neural nets

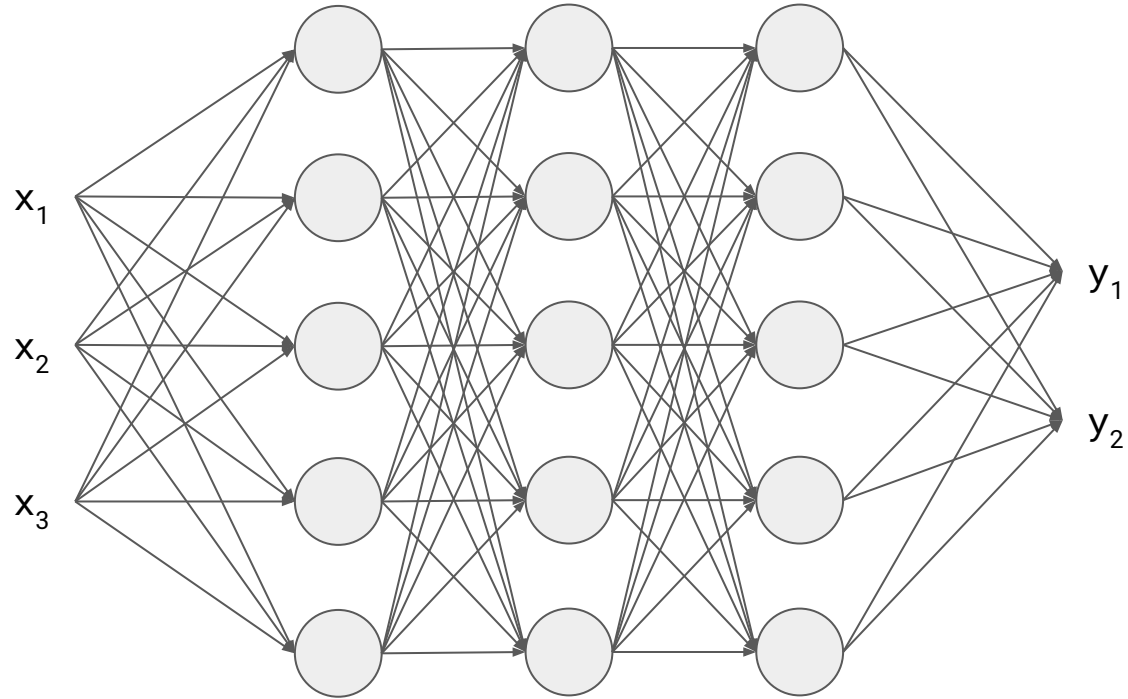


?

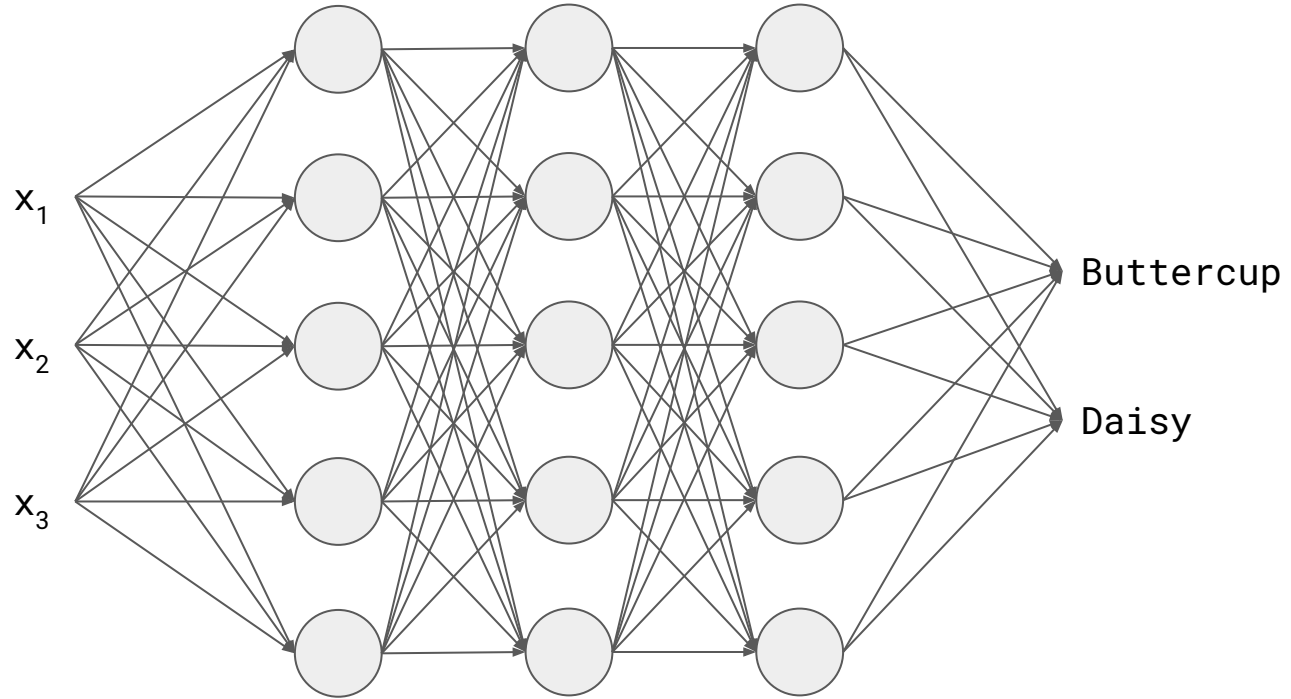
04. Neural nets



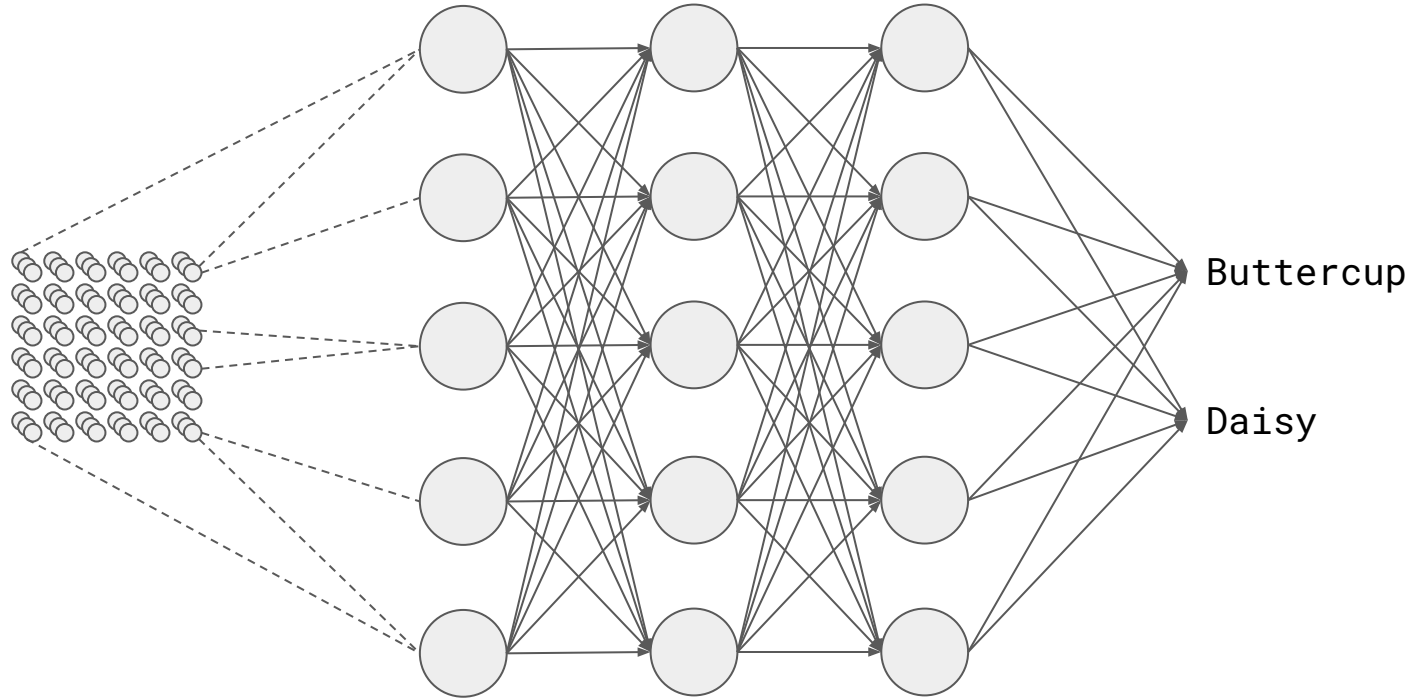
04. Neural nets



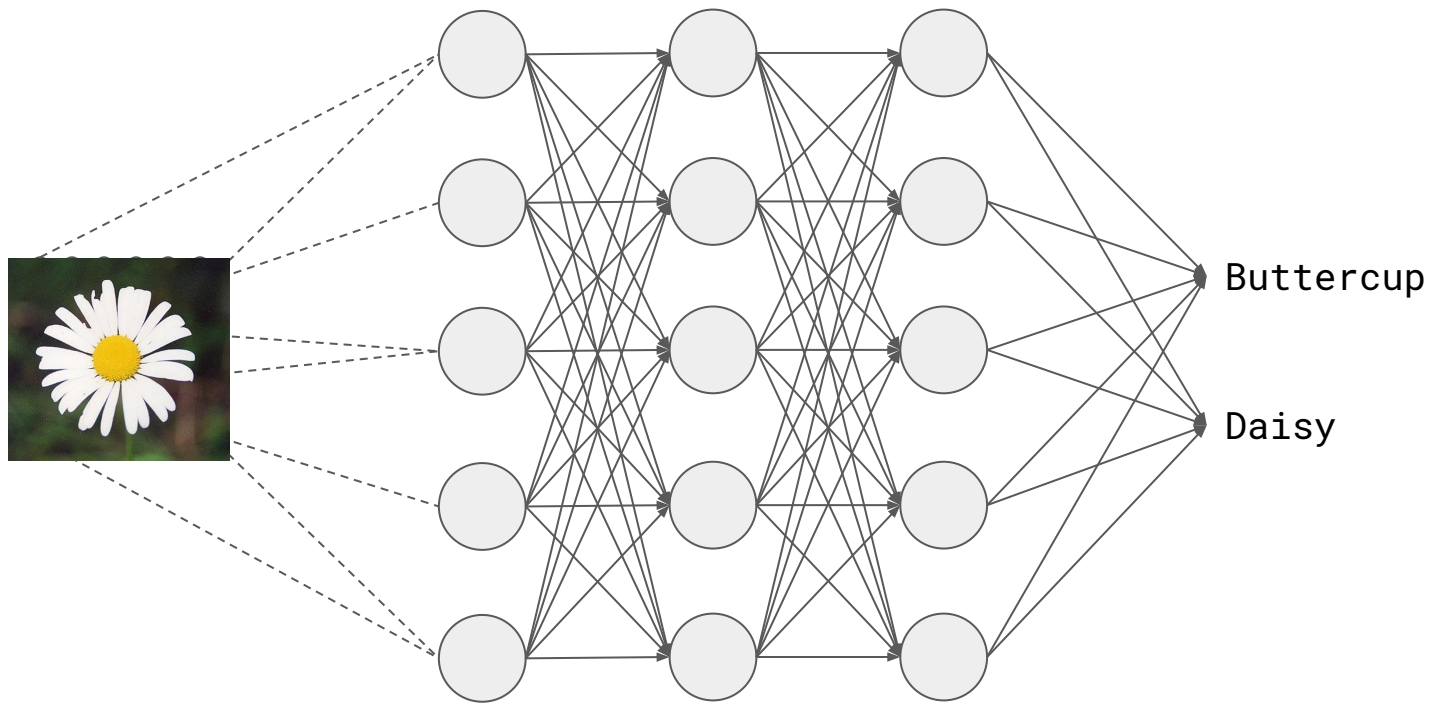
04. Neural nets



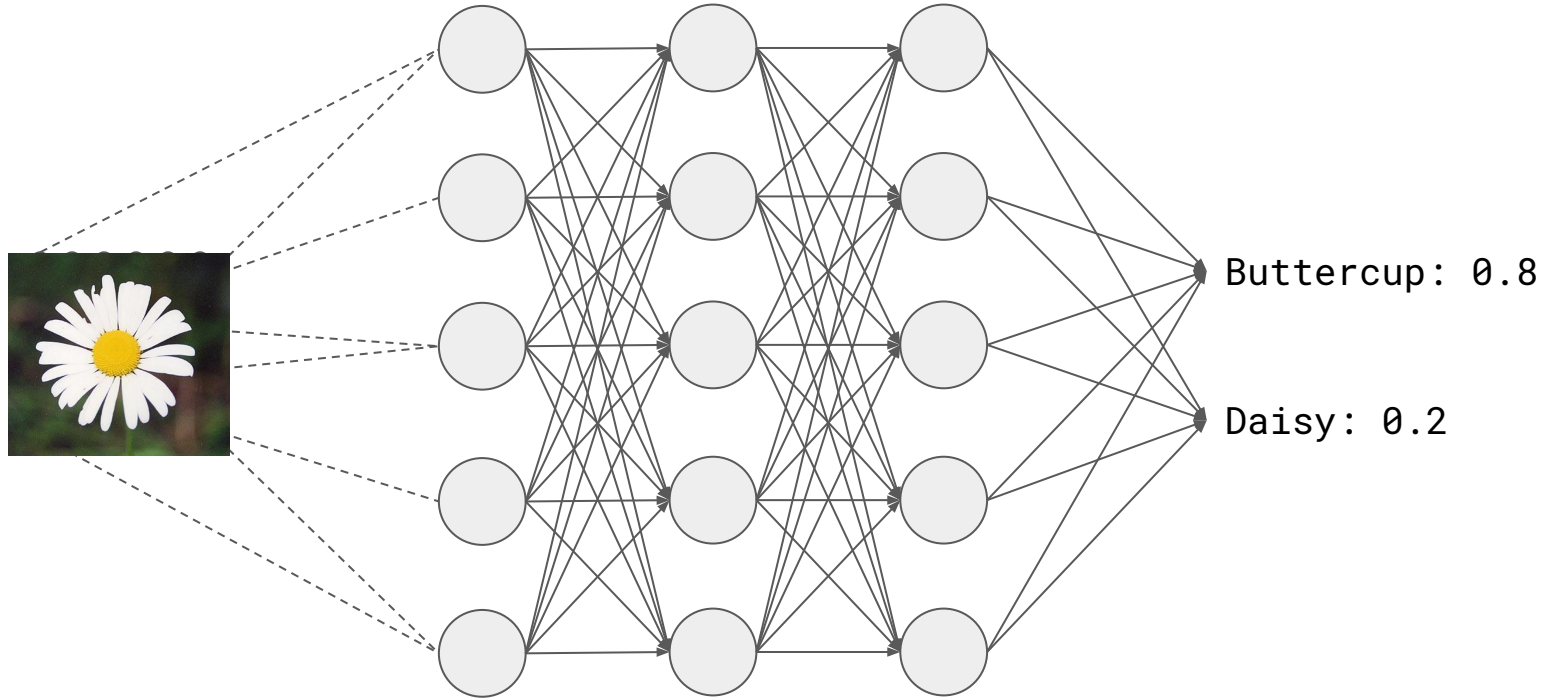
04. Neural nets



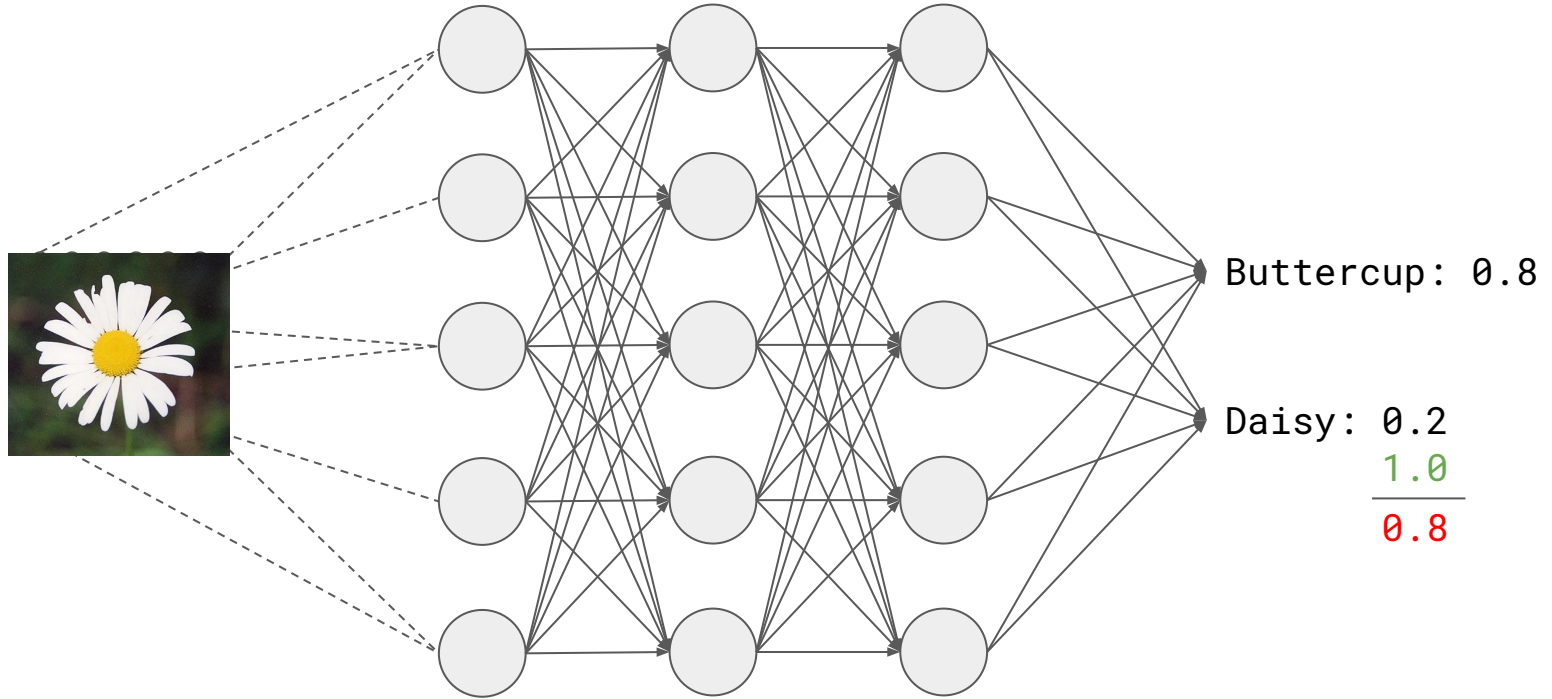
04. Neural nets



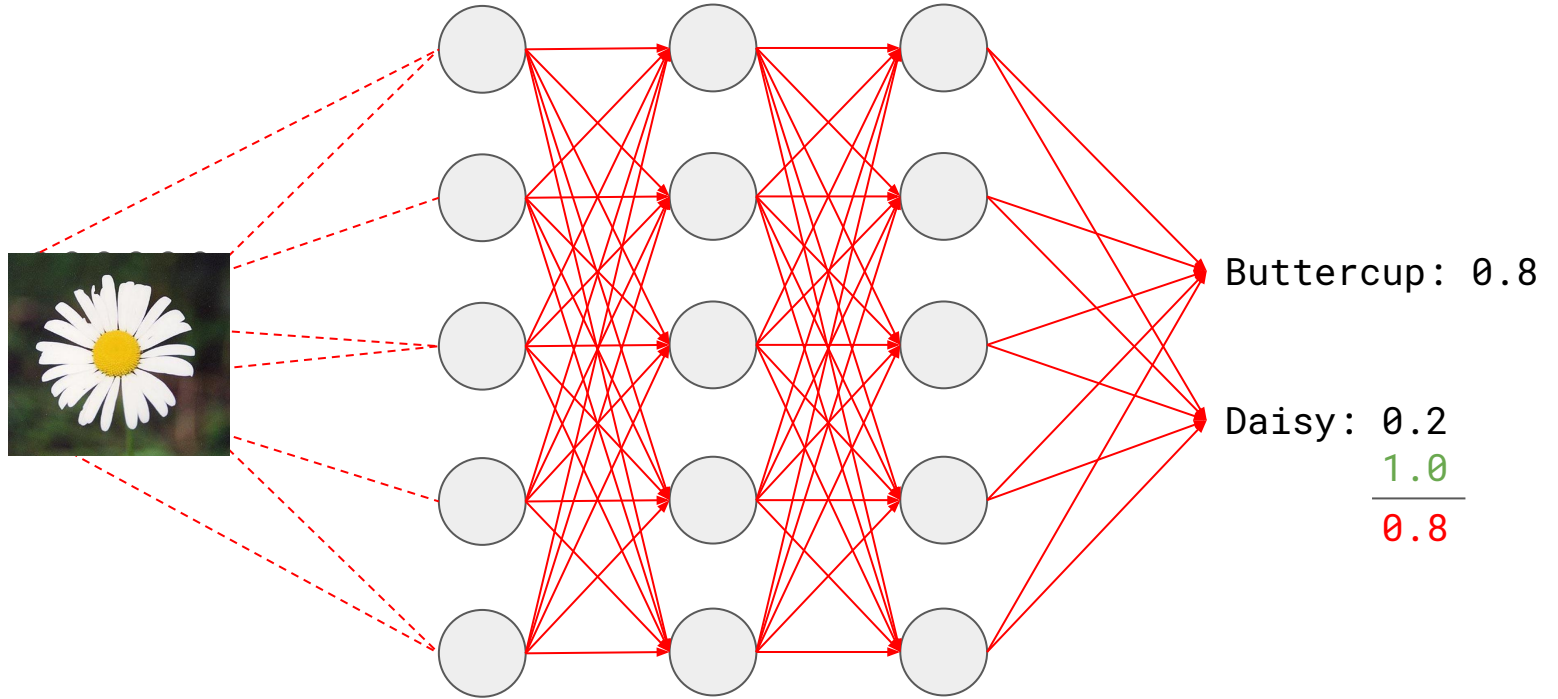
04. Neural nets



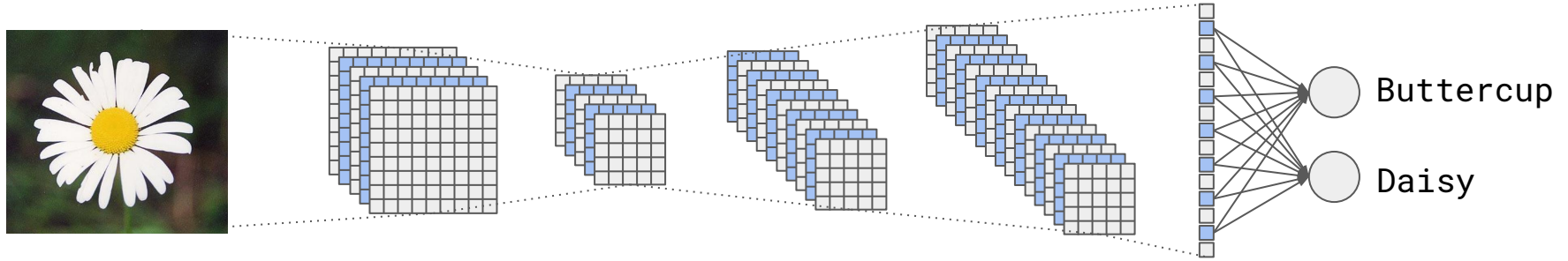
04. Neural nets



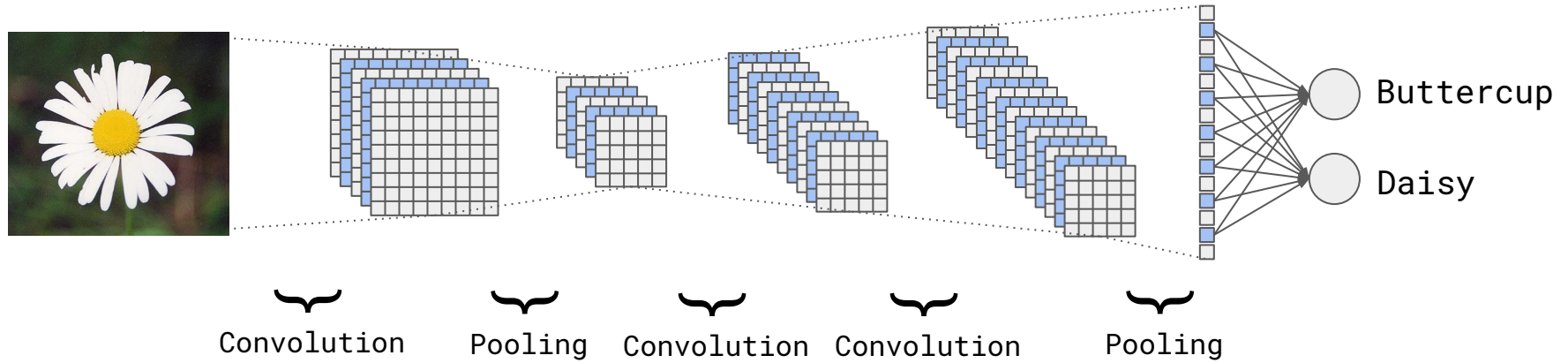
04. Neural nets



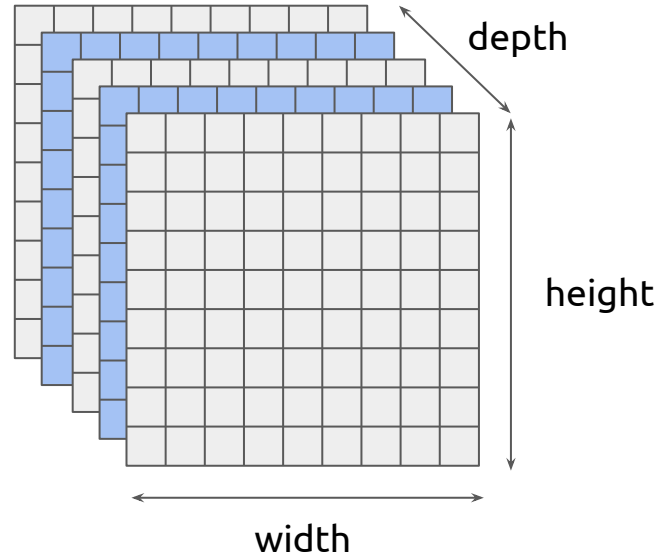
05. Convolutional Neural Nets



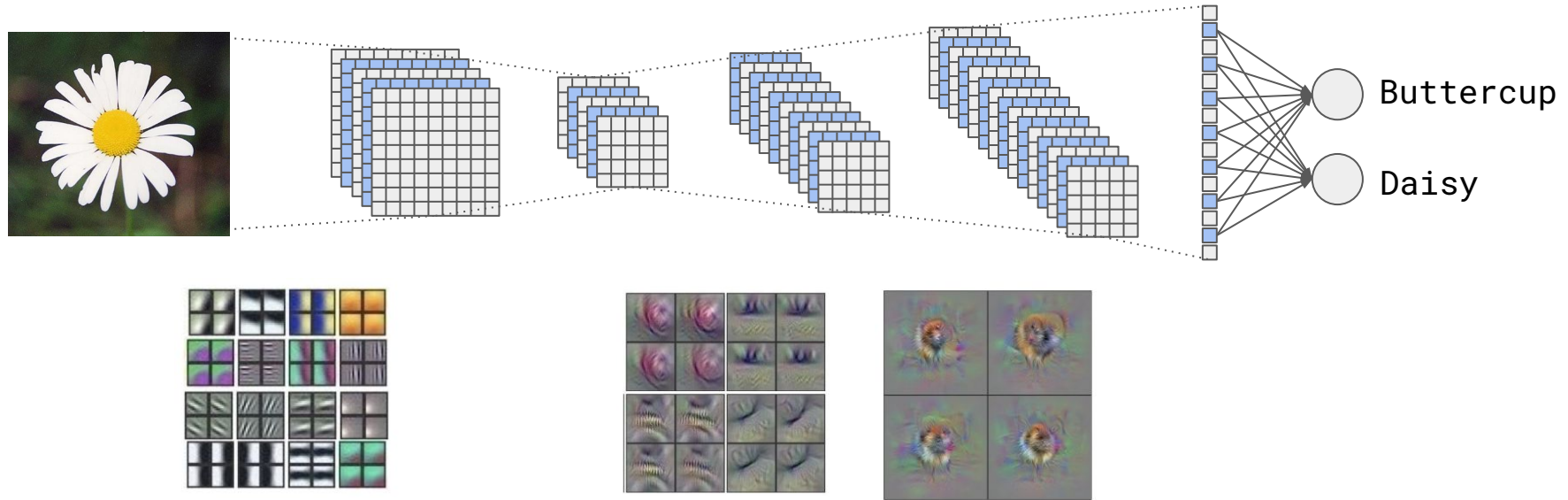
05. Convolutional Neural Nets



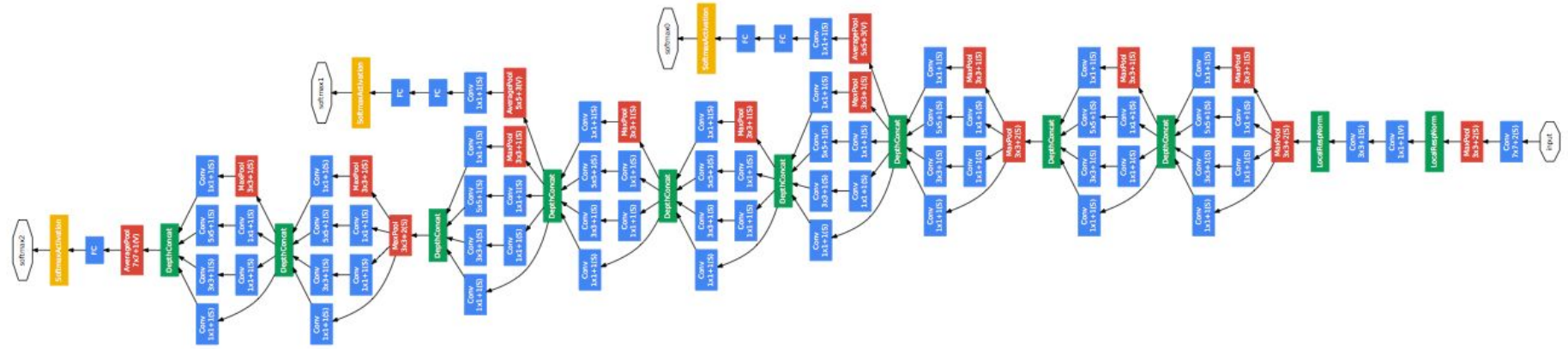
05. Convolutional Neural Nets



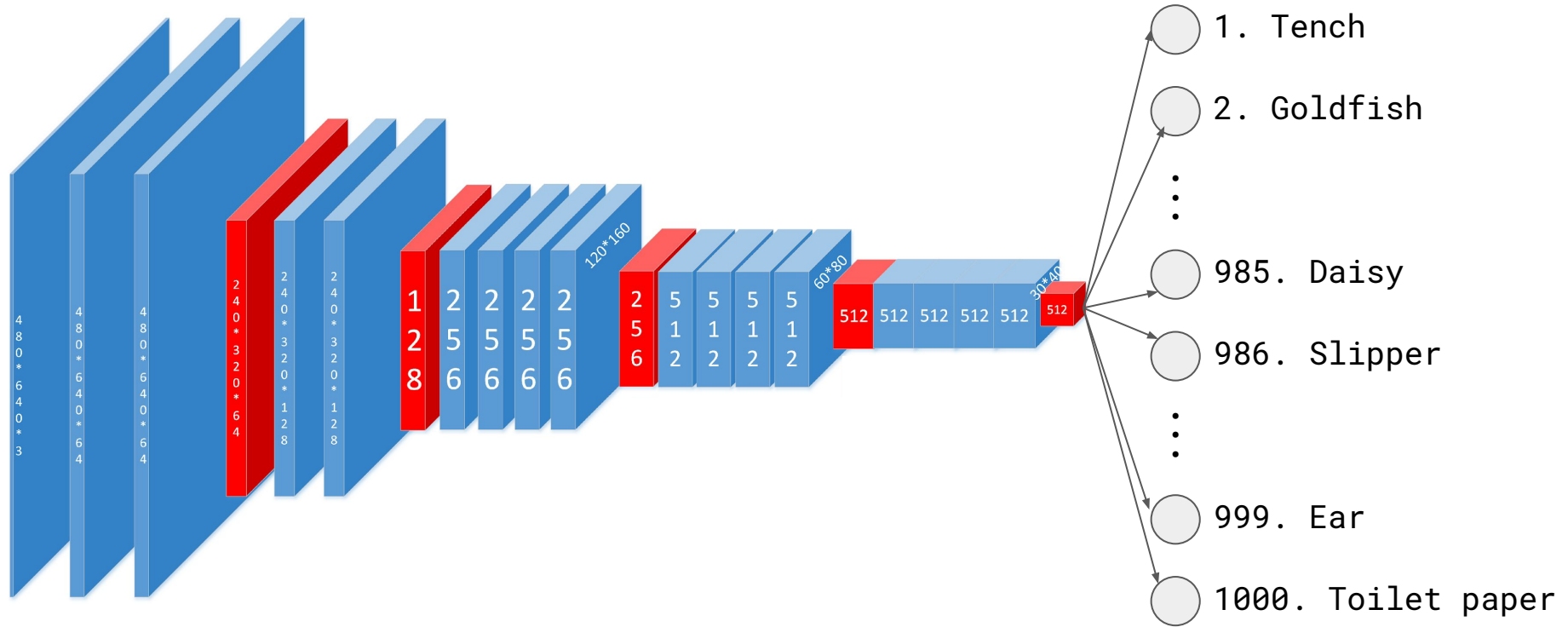
05. Convolutional Neural Nets



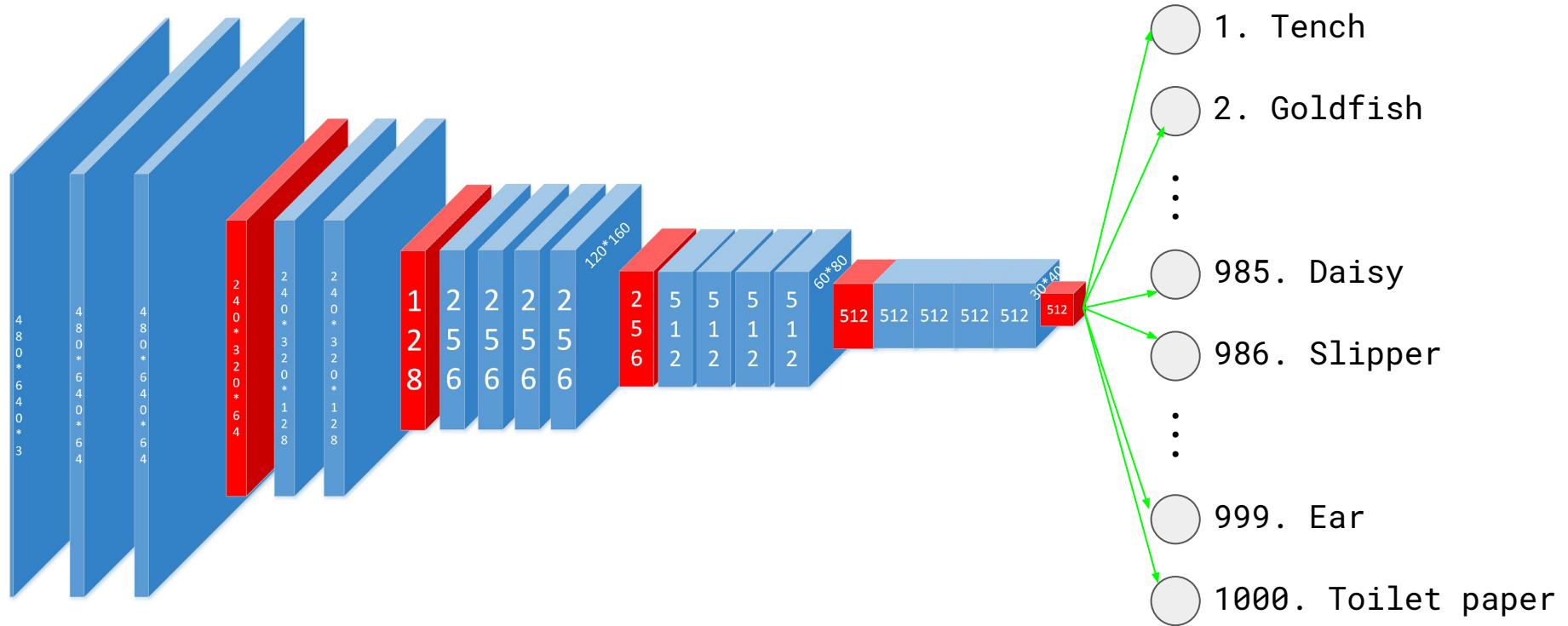
05. Convolutional Neural Nets



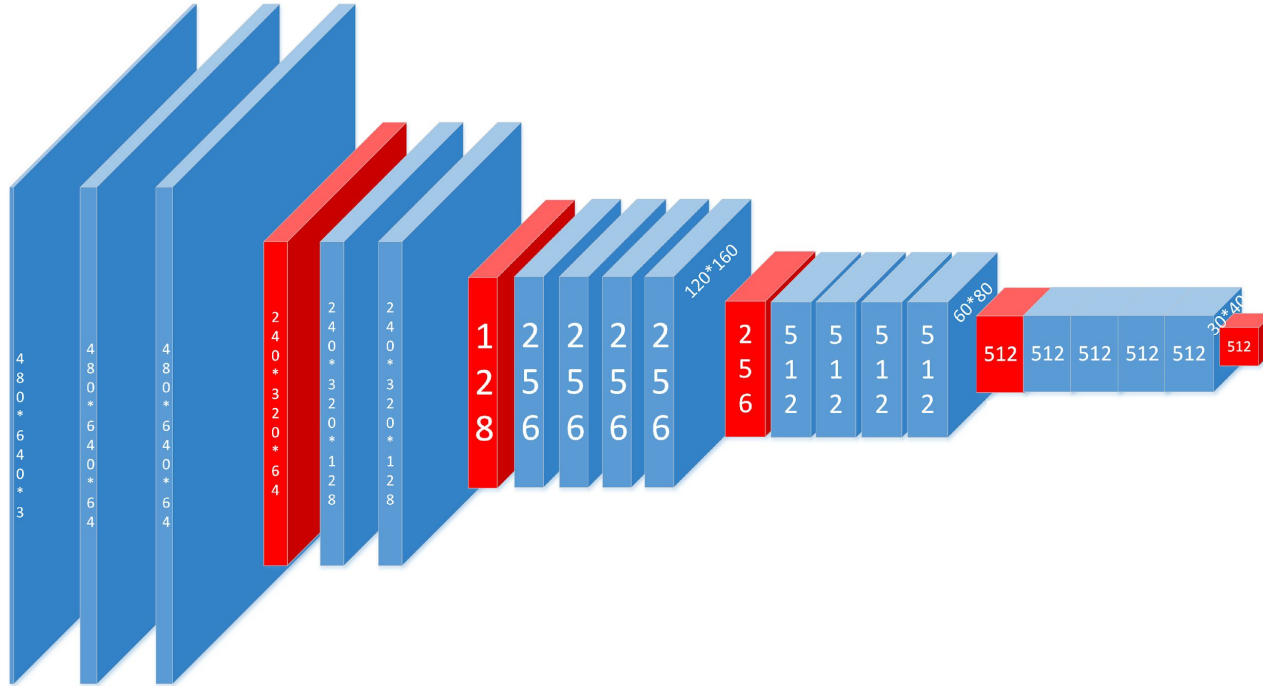
06. Transfer learning



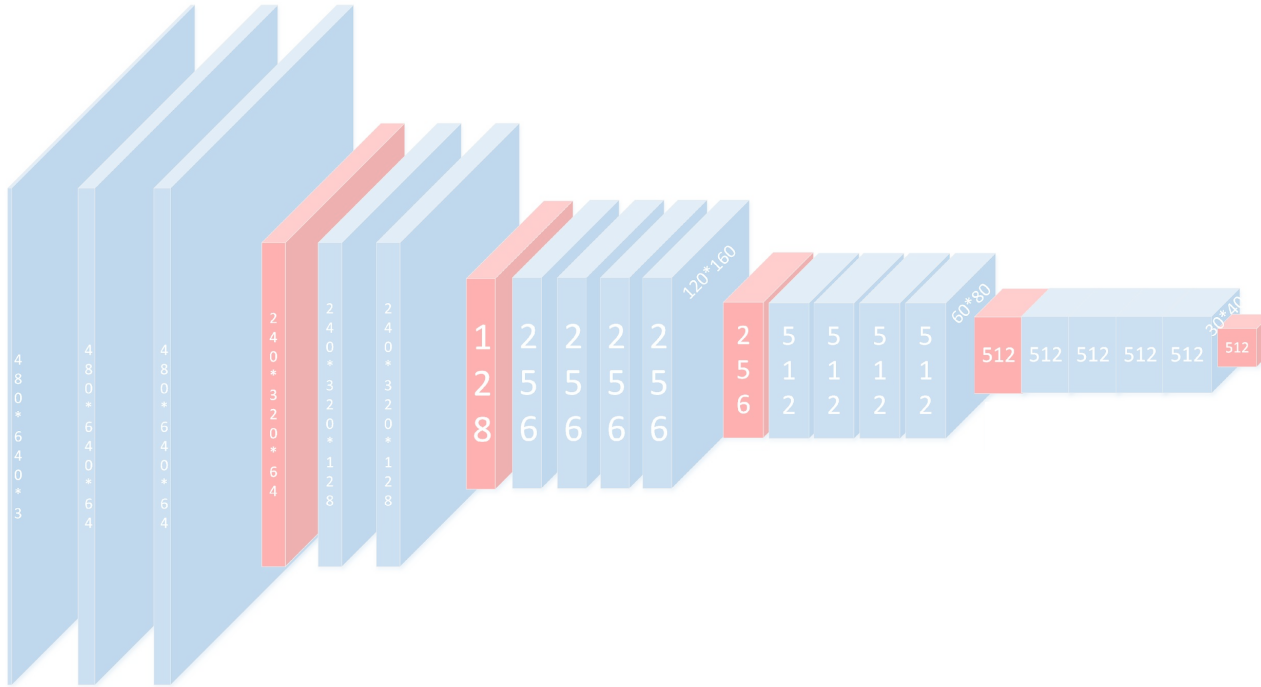
06. Transfer learning



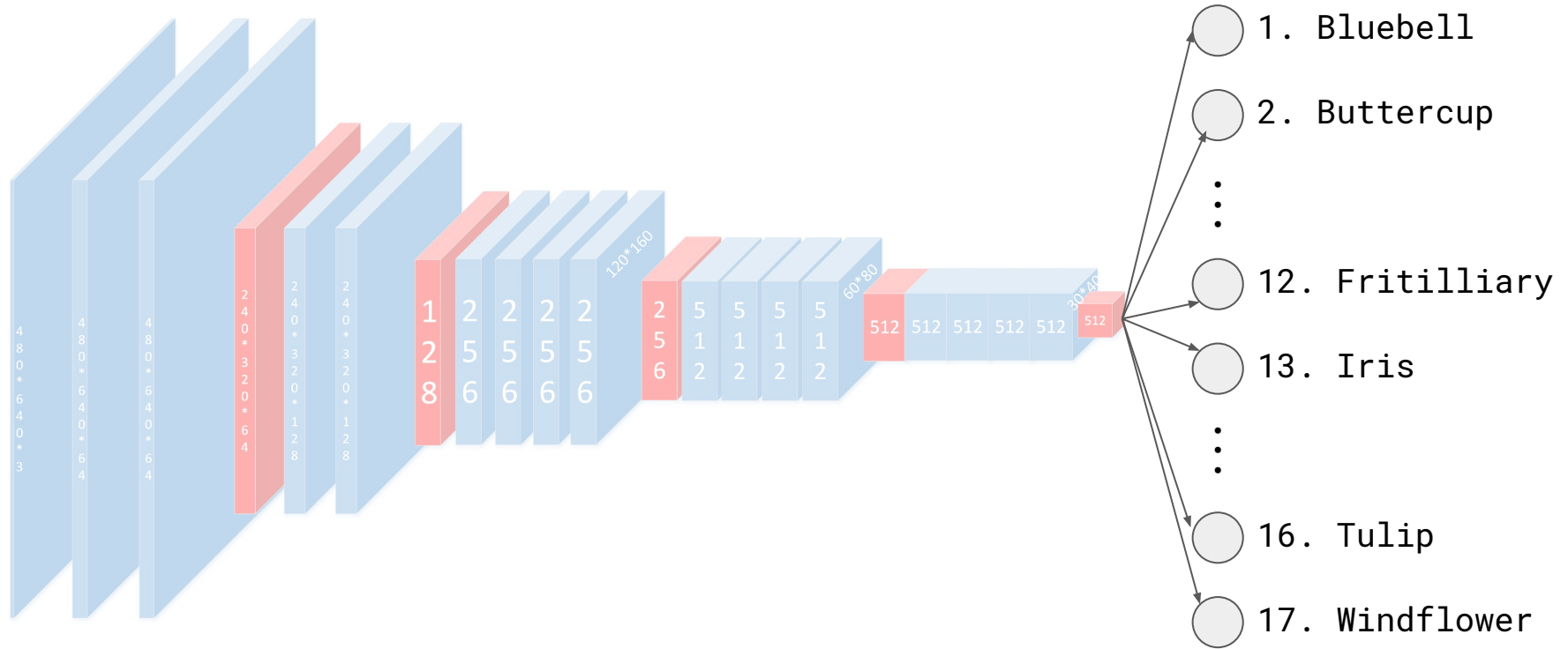
06. Transfer learning



06. Transfer learning



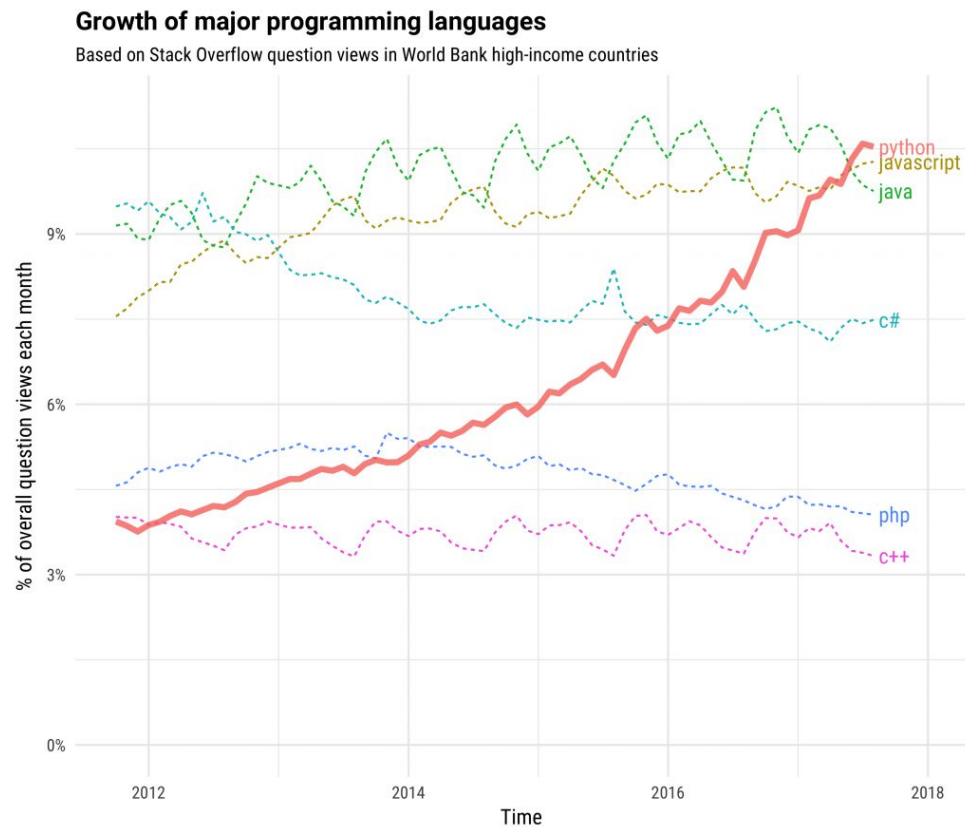
06. Transfer learning

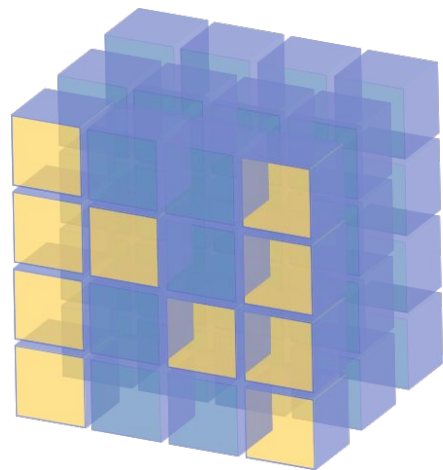


07. Python

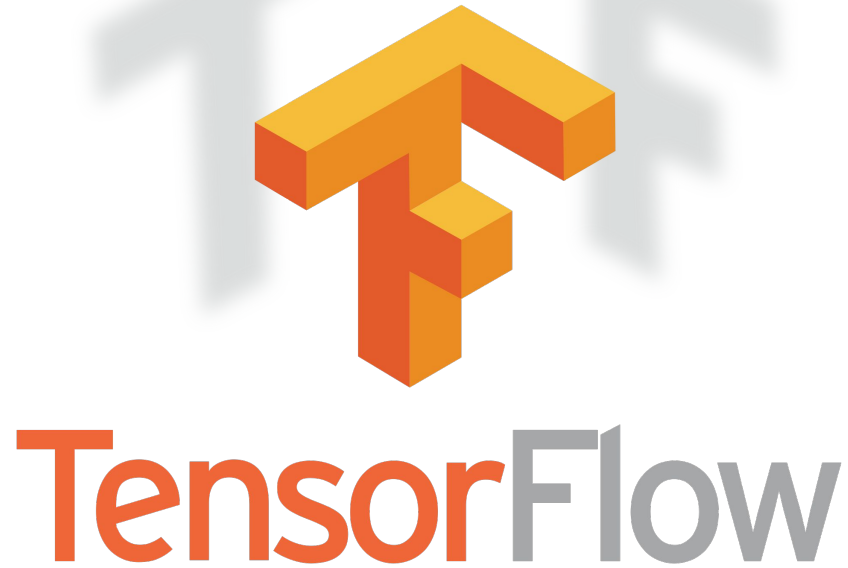


07. Python





NumPy



07. Python

1	0	5	0	3	4
9	0	12	4	6	19
2	1	27	4	2	0
8	3	8	5	11	1
13	8	4	6	7	3
1	0	2	12	0	4

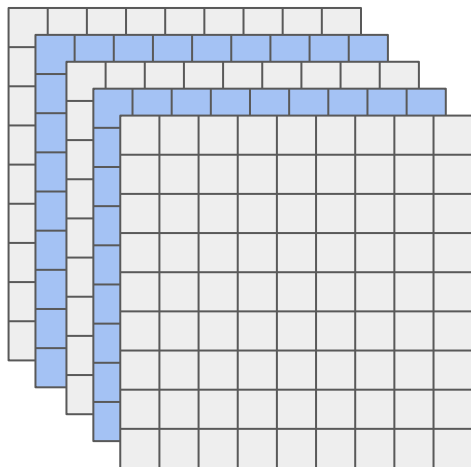
07. Python

$$y = ax + b$$

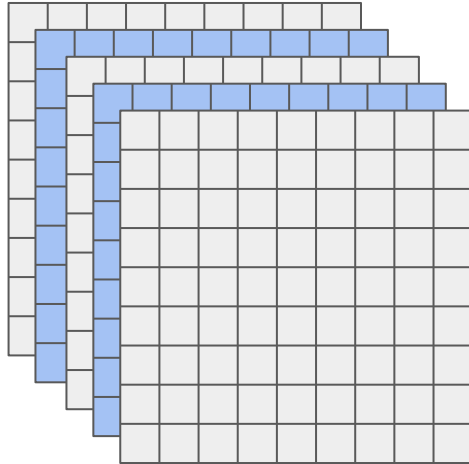
```
x = tf.placeholder(  
    dtype=tf.float32,  
    shape=(4, 4)  
)  
a = tf.Variable(  
    initial_value=tf.zeros((4, 4)),  
    dtype=tf.float32,  
    trainable=True  
)  
b = tf.Variable(  
    initial_value=tf.zeros((4, 4)),  
    dtype=tf.float32,  
    trainable=True  
)  
  
y = tf.add(tf.matmul(x, a), b)
```



07. Python



07. Python



```
BATCH_SIZE = 4
IMAGE_SIZE = (256, 256, 3)

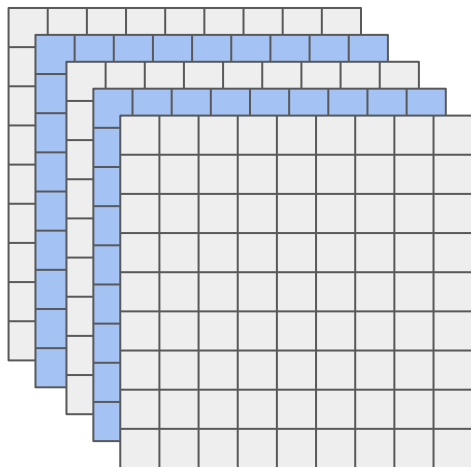
inputs = tf.placeholder(
    shape=(BATCH_SIZE,) + IMAGE_SIZE,
    dtype=tf.float32
)

weight_shape = tf.stack([5, 5, 3, 16])
weight_initializer = tf.random_normal(
    shape=weight_shape,
    stddev=.03
)
weights = tf.Variable(
    weight_initializer,
    trainable=True,
)

bias_initializer = tf.zeros(16)
bias = tf.Variable(
    bias_initializer,
    trainable=True
)

conv = tf.nn.conv2d(inputs, weights,
    strides=[1, 1, 1, 1],
    padding='SAME'
)
conv = tf.nn.bias_add(conv, bias)
```

07. Python



```
BATCH_SIZE = 4
IMAGE_SIZE = (256, 256, 3)

inputs = tf.placeholder(
    shape=(BATCH_SIZE,) + IMAGE_SIZE,
    dtype=tf.float32
)

weight_shape = tf.stack([5, 5, 3, 16])
weight_initializer = tf.random_normal(
    shape=weight_shape,
    stddev=.03
)
weights = tf.Variable(
    weight_initializer,
    trainable=True,
)

bias_initializer = tf.zeros(16)
bias = tf.Variable(
    bias_initializer,
    trainable=True
)

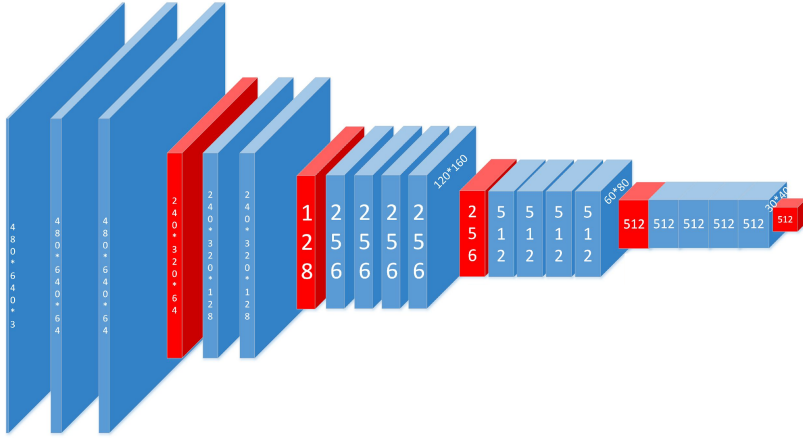
conv = tf.nn.conv2d(inputs, weights,
    strides=[1, 1, 1, 1],
    padding='SAME'
)
conv = tf.nn.bias_add(conv, bias)
```

```
IMAGE_SIZE = (256, 256, 3)

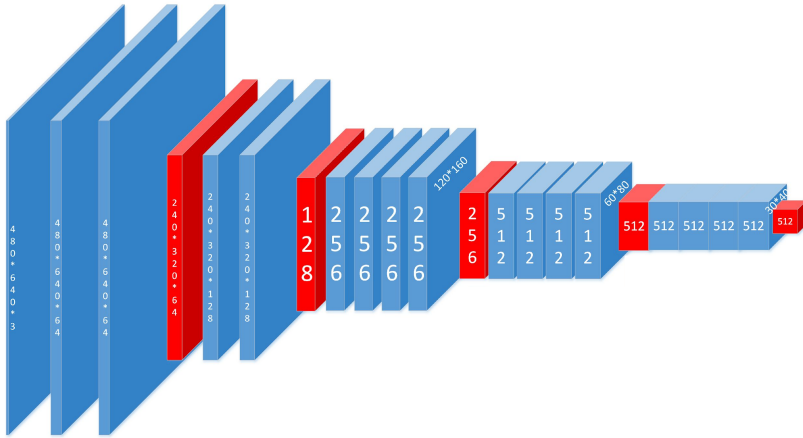
inputs = Inputs(shape=IMAGE_SIZE)

conv = Conv2D(kernels=16, filters=(3, 3))(inputs)
```

07. Python



07. Python



```
from keras.applications.vgg19 import VGG19
model = VGG19()
```


07. Python

```
from keras.applications.vgg19 import VGG19

model = ... # Create a model

model.summary() # Print the structure of the model

model.fit() # Train the model

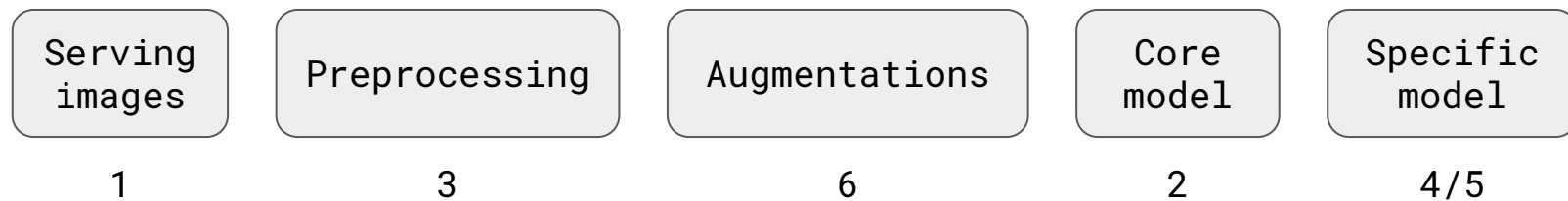
model.evaluate() # Evaluate performance on a validation set

model.predict() # Run predictions on new data

model.save() # Save the model to file

model.load() # Load the model into memory
```

07. Python



07. Python

'bluebell'	0	[1, 0, 0, 0, 0, 0]
'buttercup'	1	[0, 1, 0, 0, 0, 0]
'colts_foot'	2	[0, 0, 1, 0, 0, 0]
'cowslip'	3	[0, 0, 0, 1, 0, 0]
'crocus'	4	[0, 0, 0, 0, 1, 0]
'daffodil'	5	[0, 0, 0, 0, 0, 1]

07. Python

