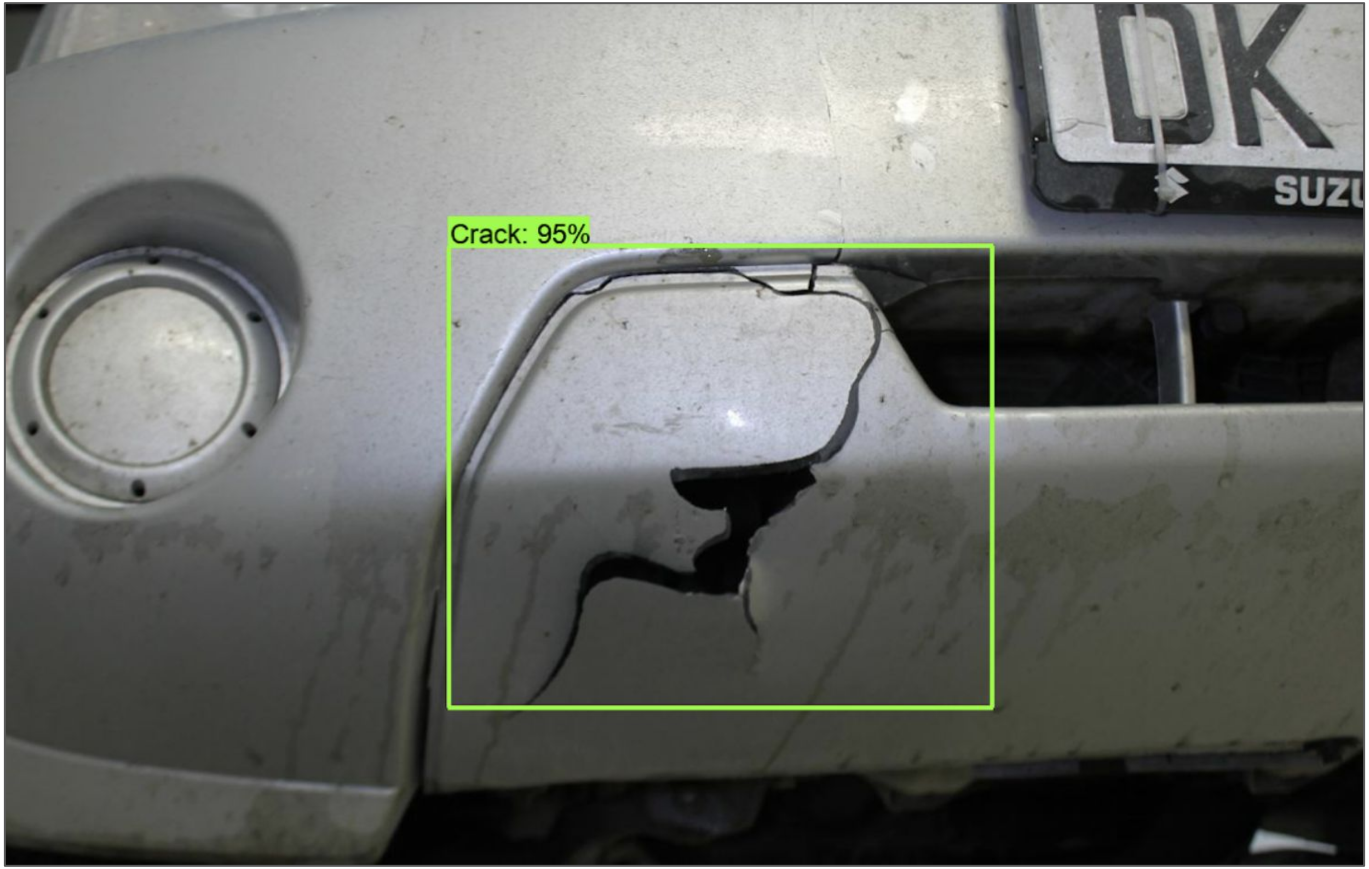
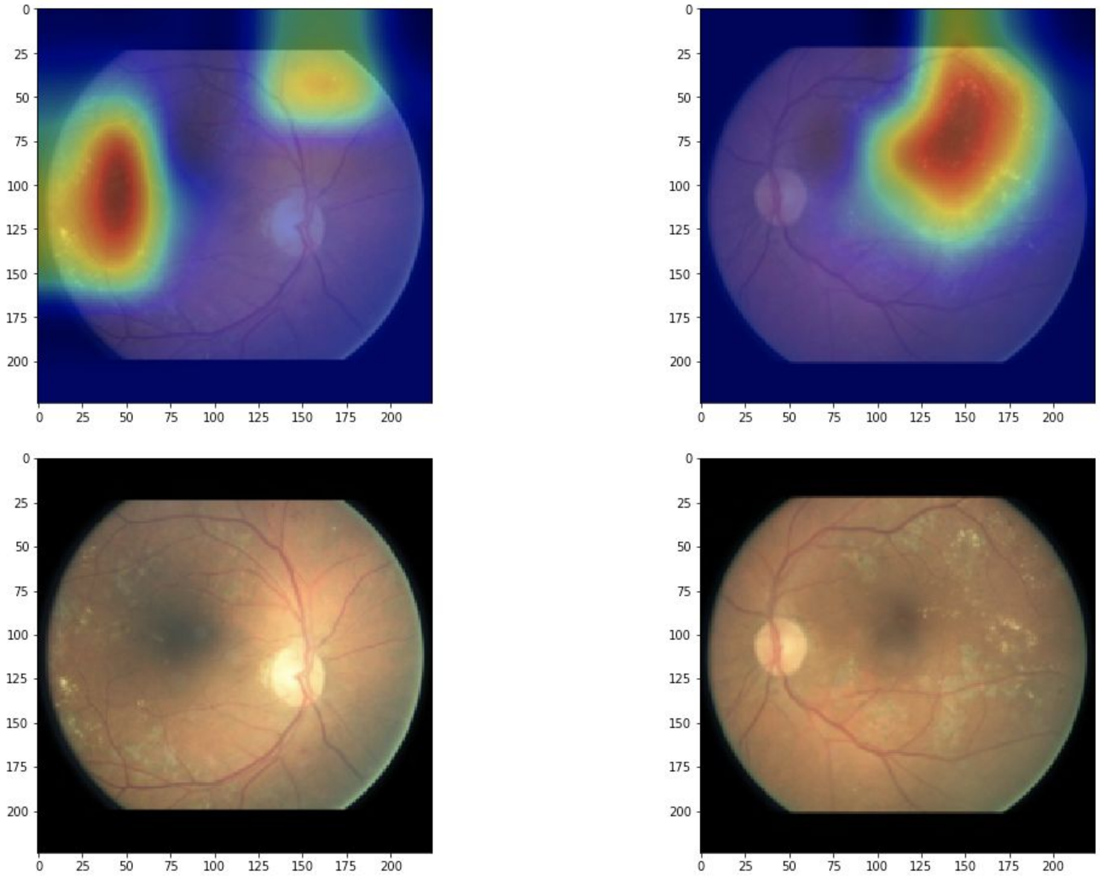
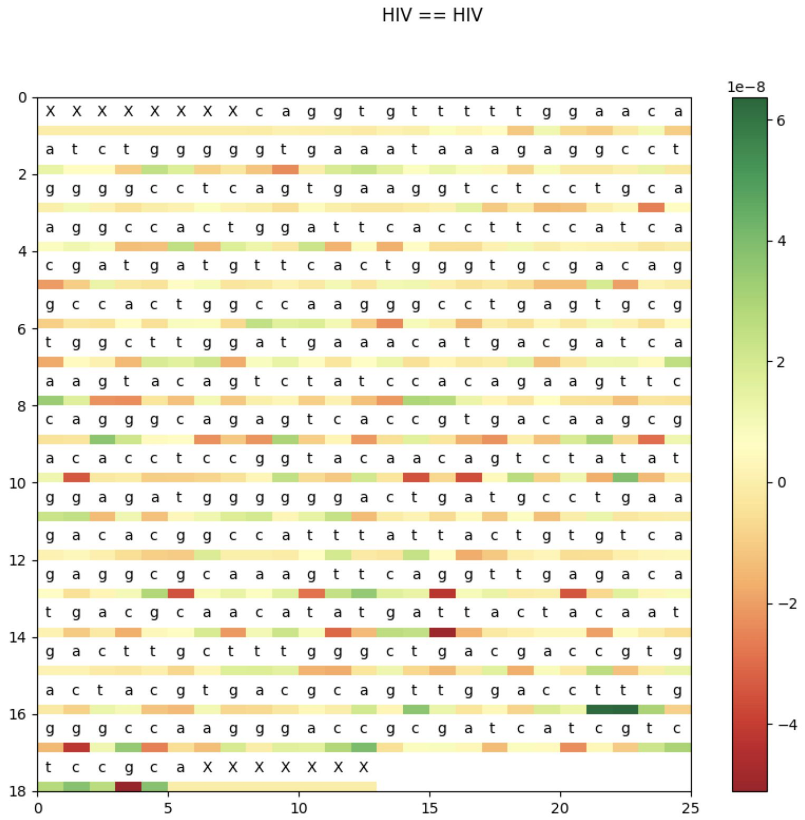
Machine learning in Python


Crack: 95%
DK
SUZU


0
25
50
75
100
125
150
175
200
0
25
50
75
100
125
150
175
200
0
25
50
75
100
125
150
175
200
0
25
50
75
100
125
150
175
200
0
25
50
75
100
125
150
175
200
0
25
50
75
100
125
150
175
200
0
25
50
75
100
125
150
175
200
0
25
50
75
100
125
150
175
200


HIV == HIV
X X X X X X X X c a g g t g t t t t t g g a a c a
a t c t g g g g g t g a a a t a a a g a g g c c t
g g g g c c t c a g t g a a g g t c t c c t g c a
a g g c c a c t g g a t t c a c c t t c c a t c a
c g a t g a t g t t c a c t g g g t g c g a c a g
g c c a c t g g c c a a g g g c c t g a g t g c g
t g g c t t g g a t g a a a c a t g a c g a t c a
a a g t a c a g t c t a t c c a c a g a a g t t c
c a g g g c a g a g t c a c c g t g a c a a g c g
a c a c c t c c g g t a c a a c a g t c t a t a t
g g a g a t g g g g g g a c t g a t g c c t g a a
g a c a c g g c c a t t t a t t a c t g t g t c a
g a g g c g c a a a g t t c a g g t t g a g a c a
t g a c g c a a c a t a t g a t t a c t a c a a t
g a c t t g c t t t g g g c t g a c g a c c g t g
a c t a c g t g a c g c a g t t g g a c c t t t g
g g g c c a a g g g a c c g c g a t c a t c g t c
t c c g c a X X X X X X X
0
2
4
6
8
10
12
14
16
18
0
5
10
15
20
25
1e−8
6
4
2
0
−2
−4

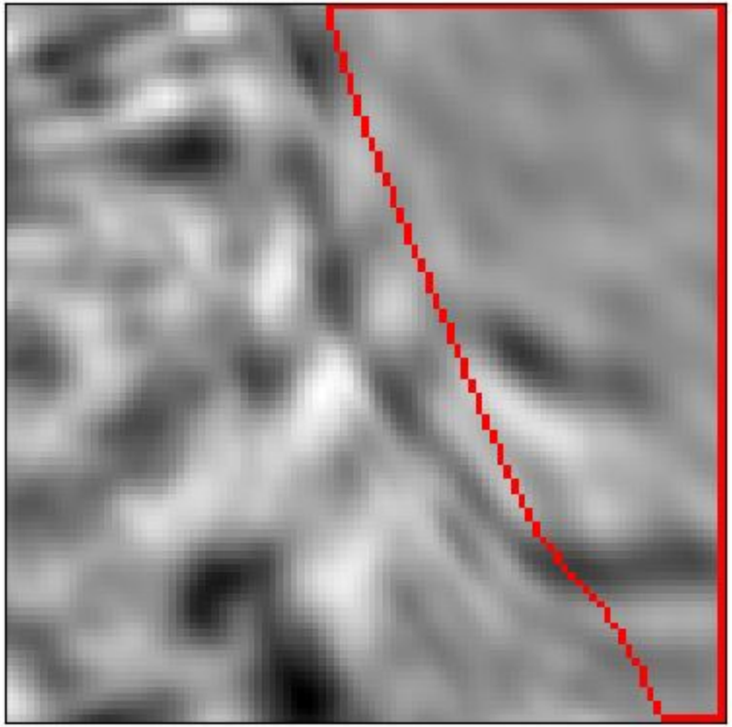
Thank you

I really appreciate the help :)

Amazing!!

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

Thanks for nothing. What a horrible user experience

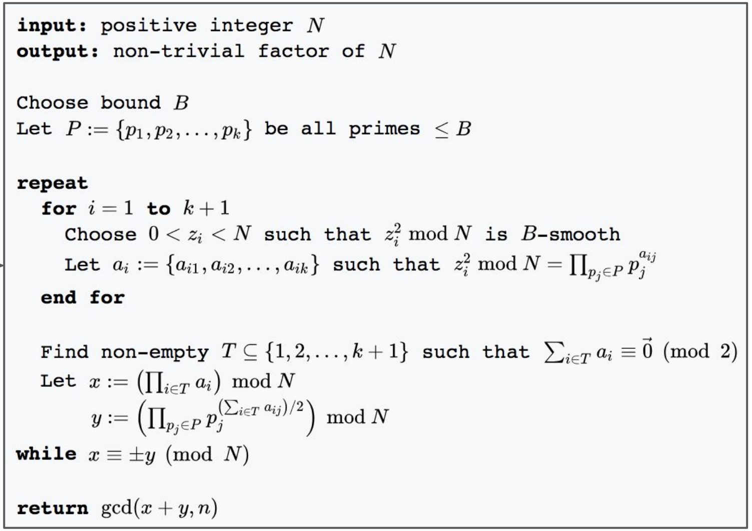


01. Algorithms

2 => {2}
3 => {3}
4 => {2, 2}
5 => {5}
6 => {2, 3}
7 => {7}
8 => {2, 2, 2}

01. Algorithms

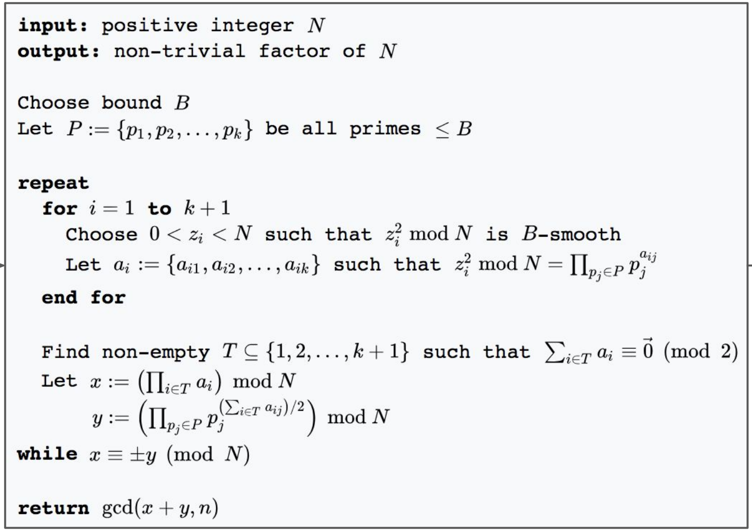
2 => {2}
3 => {3}
4 => {2, 2}
5 => {5}
6 => {2, 3}
7 => {7}
8 => {2, 2, 2}



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

Choose bound $B$
Let $P := \{p_1, p_2, \ldots, p_k\}$ 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 $a_i := \{a_{i1}, a_{i2}, \ldots, a_{ik}\}$ such that $z_i^2 \bmod N = \prod_{p_j \in P} p_j^{a_{ij}}$
  end for

  Find non-empty $T \subseteq \{1, 2, \ldots, k + 1\}$ 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 => {2}
3 => {3}
4 => {2, 2}
5 => {5}
6 => {2, 3}
7 => {7}
8 => {2, 2, 2}



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

Choose bound $B$
Let $P := \{p_1, p_2, \ldots, p_k\}$ 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 $a_i := \{a_{i1}, a_{i2}, \ldots, a_{ik}\}$ such that $z_i^2 \bmod N = \prod_{p_j \in P} p_j^{a_{ij}}$
  end for

  Find non-empty $T \subseteq \{1, 2, \ldots, k+1\}$ 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

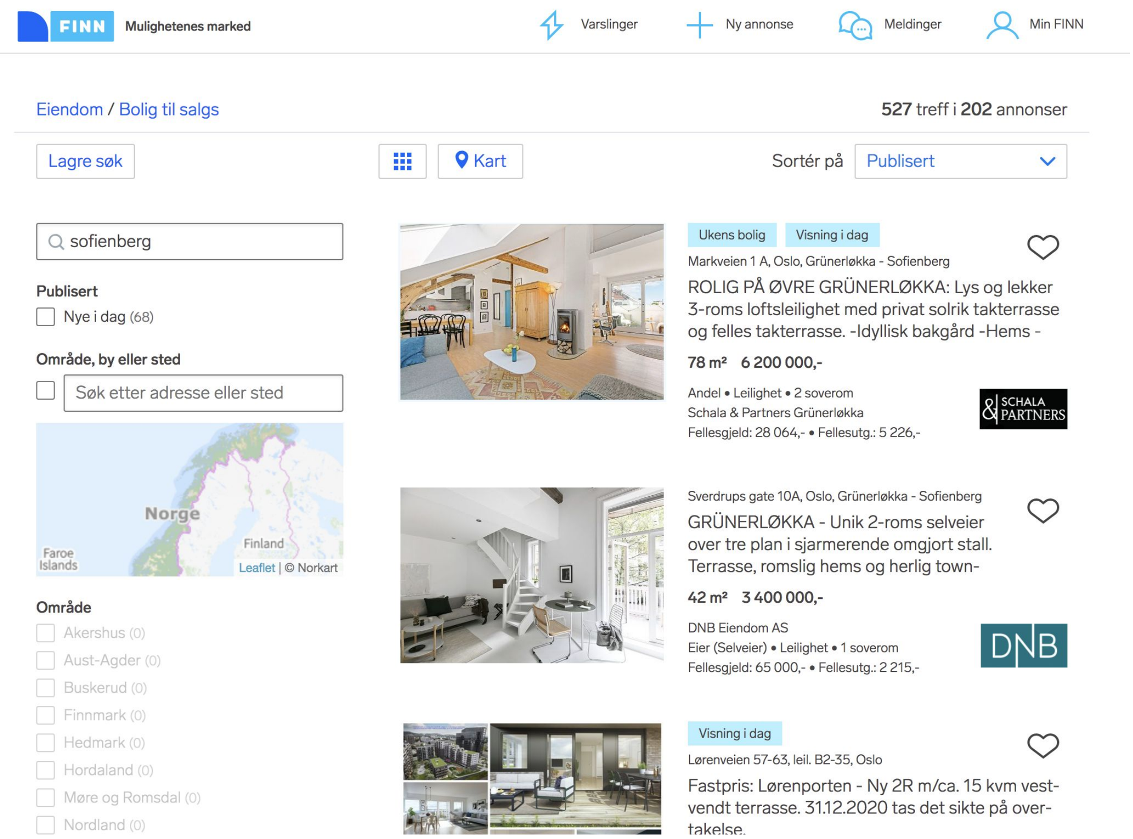
02. Machine Learning

2 => {2}
3 => {7}
4 => {}
5 => {2, 2, 2, 2, 2, ...}
6 => {“airplane”}
7 => {🐻, 🐈}
8 => {ώ}

02. Machine Learning

2 => {2}
3 => {7}
4 => {}
5 => {2, 2, 2, 2, 2, ...} → ?
6 => {“airplane”}
7 => {🐻, 🐈}
8 => {ώ}

03. Linear regression



FINN Mulighetenes marked

Varslinger | Ny annonse | Meldinger | Min FINN

Eiendom / Bolig til salgs

527 treff i **202** annonser

Lagre søk | Kart | Sortér på: Publisert

sofienberg

Publisert

□ Nye i dag (68)

Område, by eller sted

□ Søk etter adresse eller sted

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 lekker 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

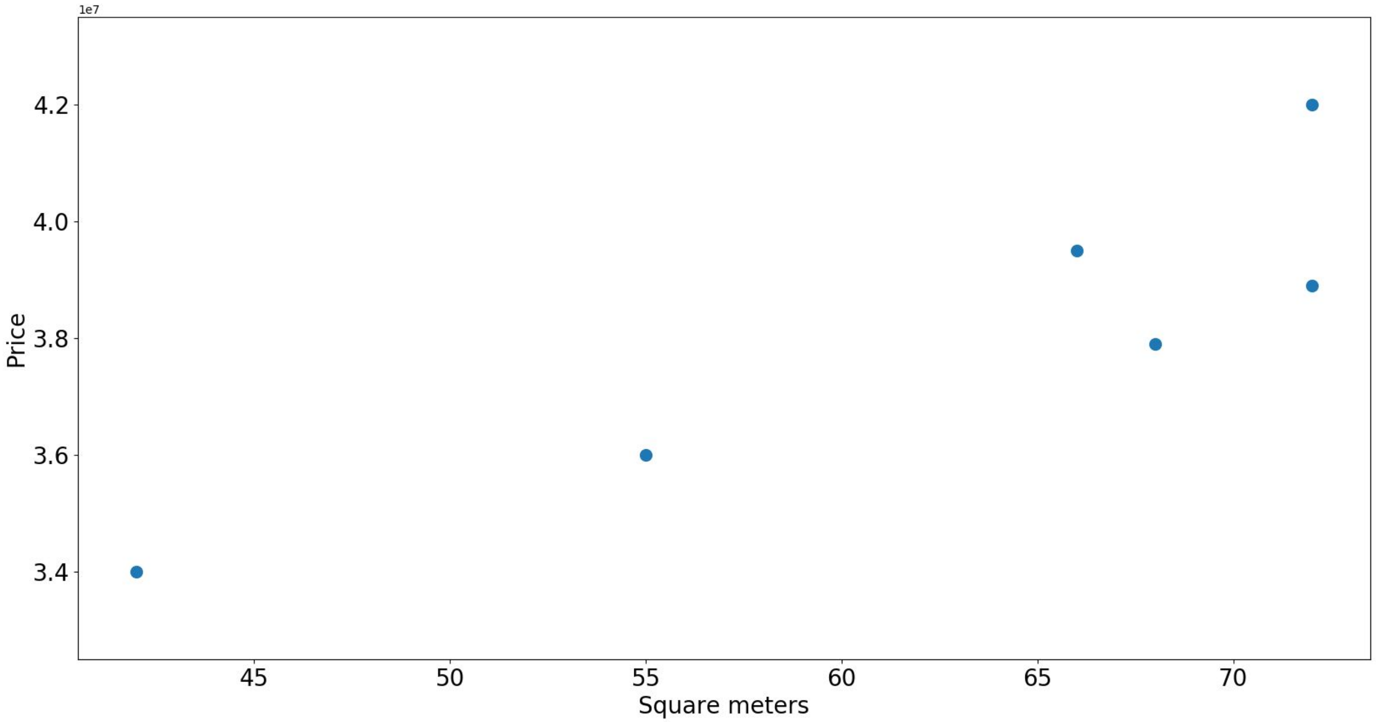
Lørenveien 57-63, leil. B2-35, Oslo

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

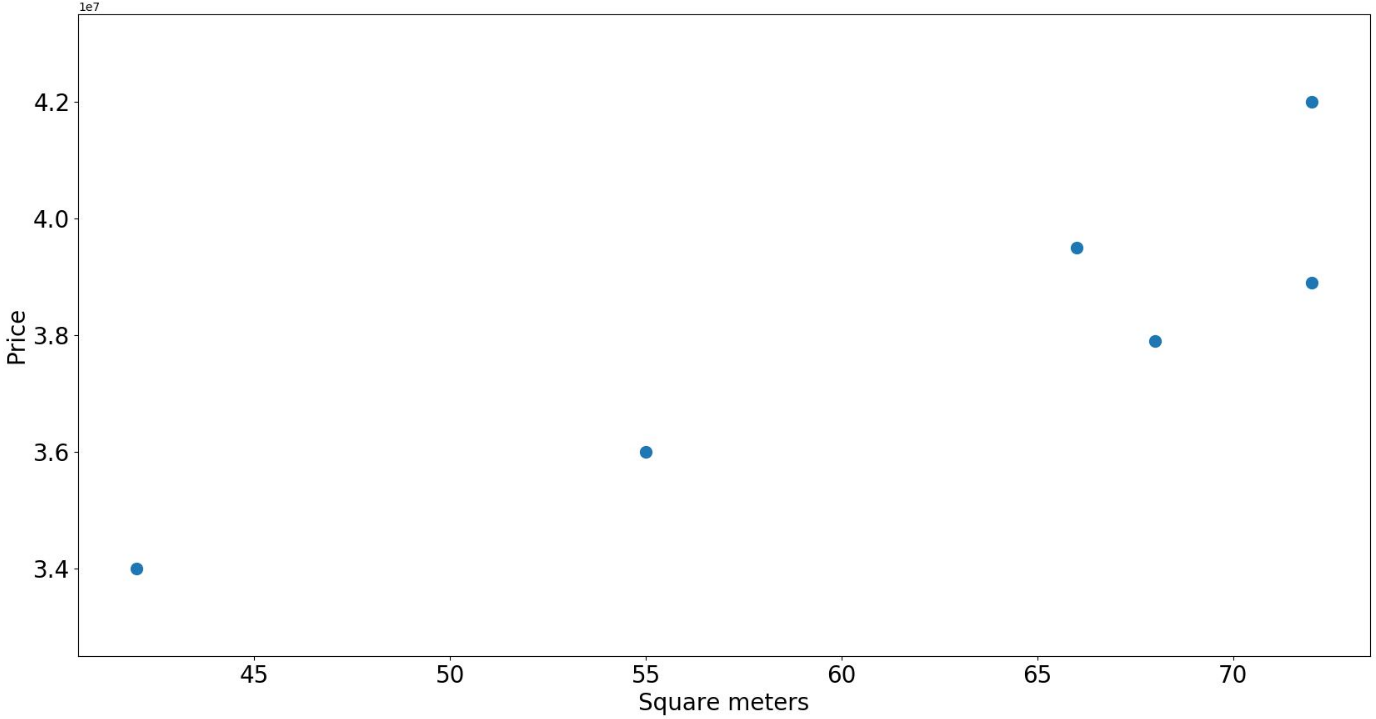
03. Linear regression

m^2	**NOK**
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

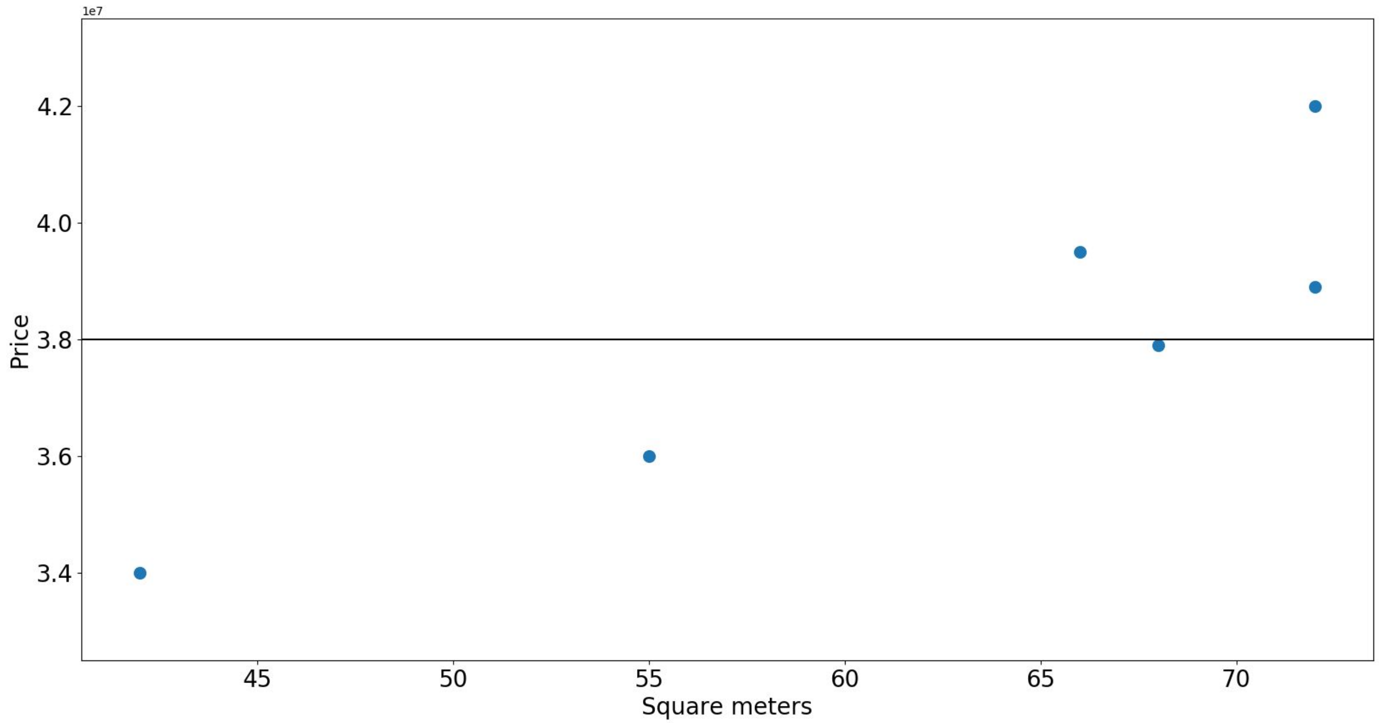


03. Linear regression



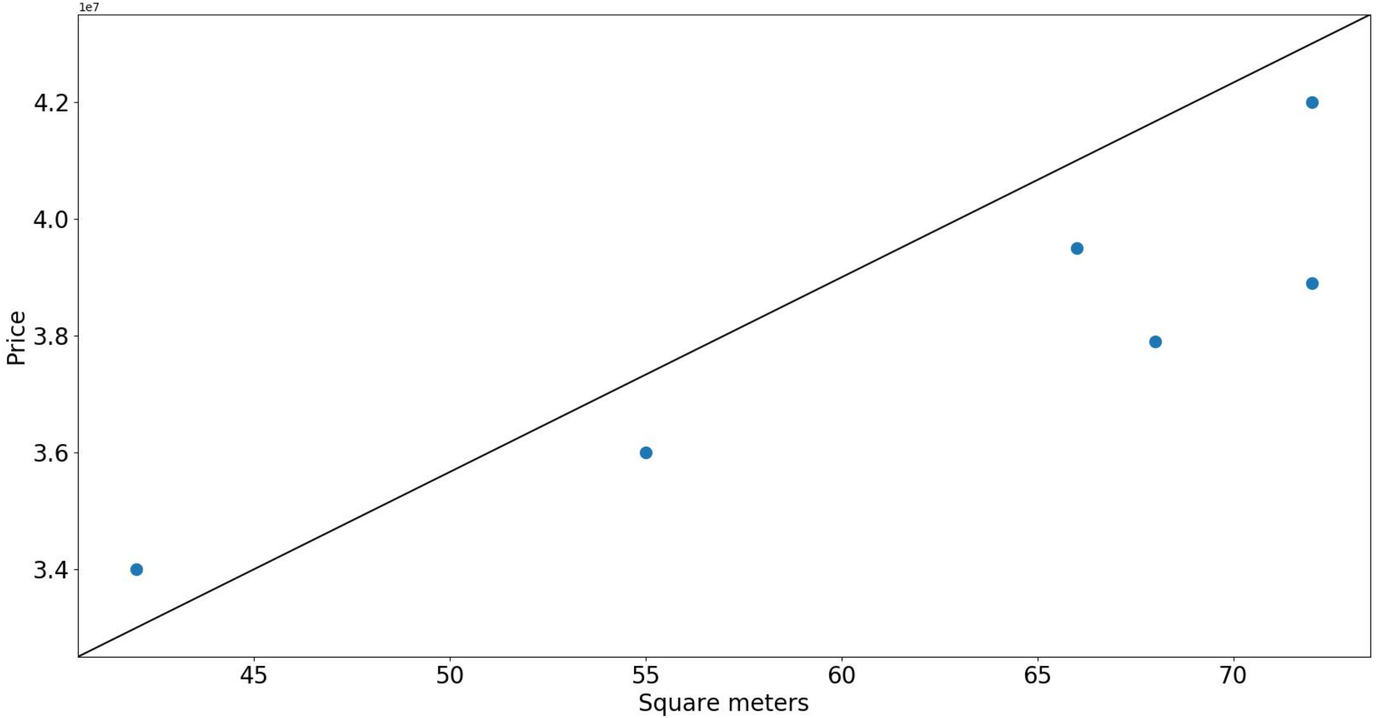
$$f(x) = ax+b$$

03. Linear regression



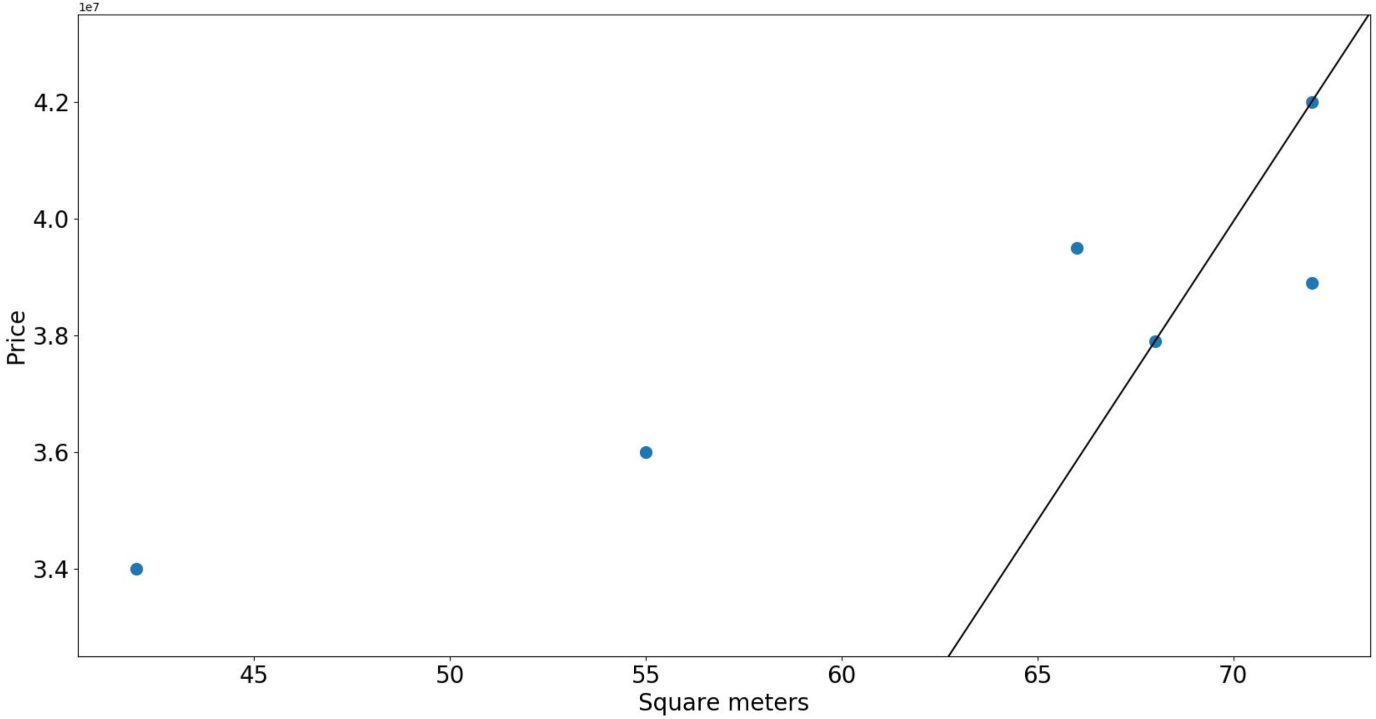
$$f(x) = 0x+3800000$$

03. Linear regression



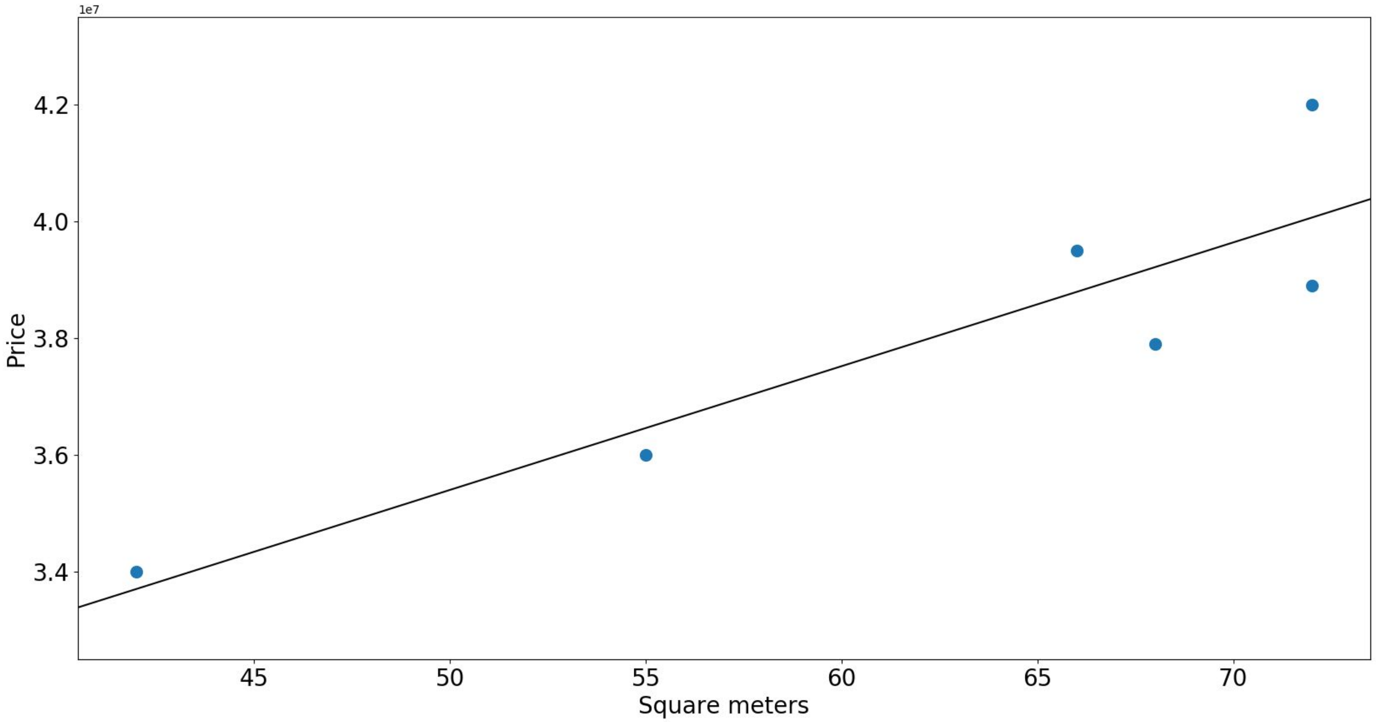
$$f(x) = 27272x+2145484$$

03. Linear regression



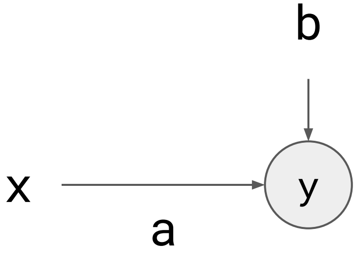
$$\text{cost}(x, f(x)) = \begin{cases} 0 \text{ if } f(x) = x \\ 1 \end{cases}$$

03. Linear regression



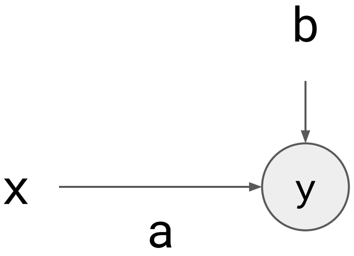
$$\mathrm{cost}(x,\ f(x)) = (f(x)\ -\ x)^2$$

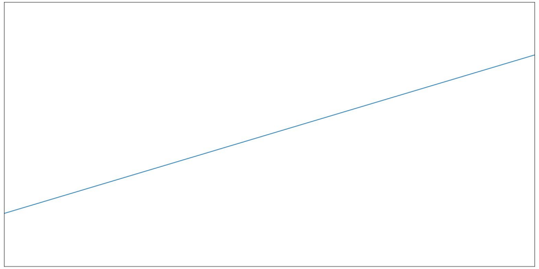
04. Neural nets



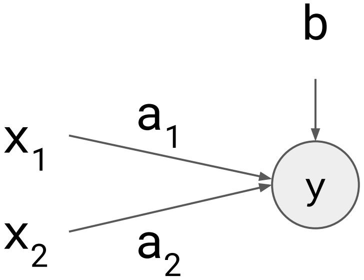
$$f(x) = ax+b$$

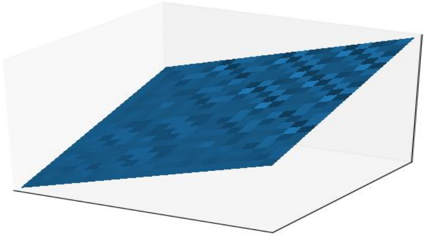
04. Neural nets



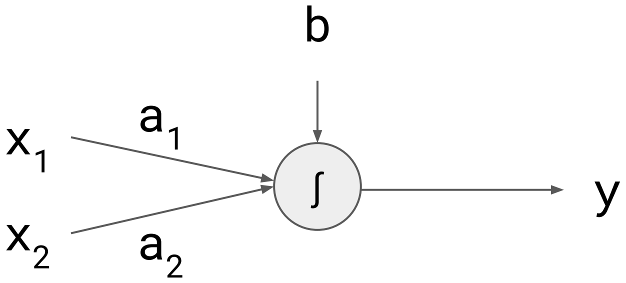


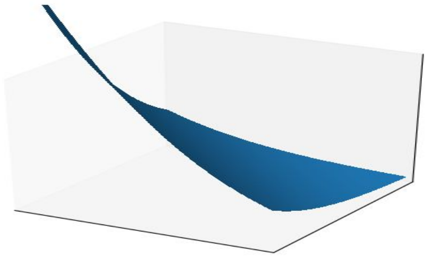
04. Neural nets



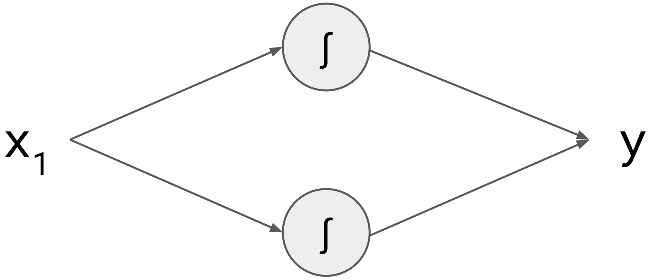


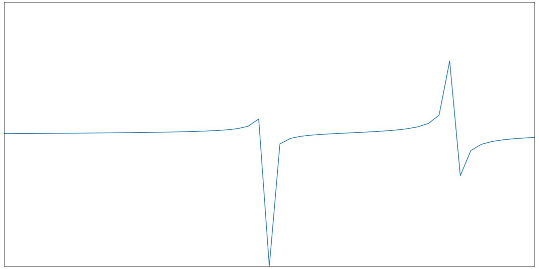
04. Neural nets



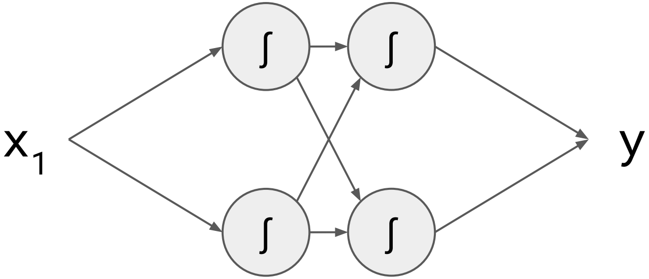


04. Neural nets



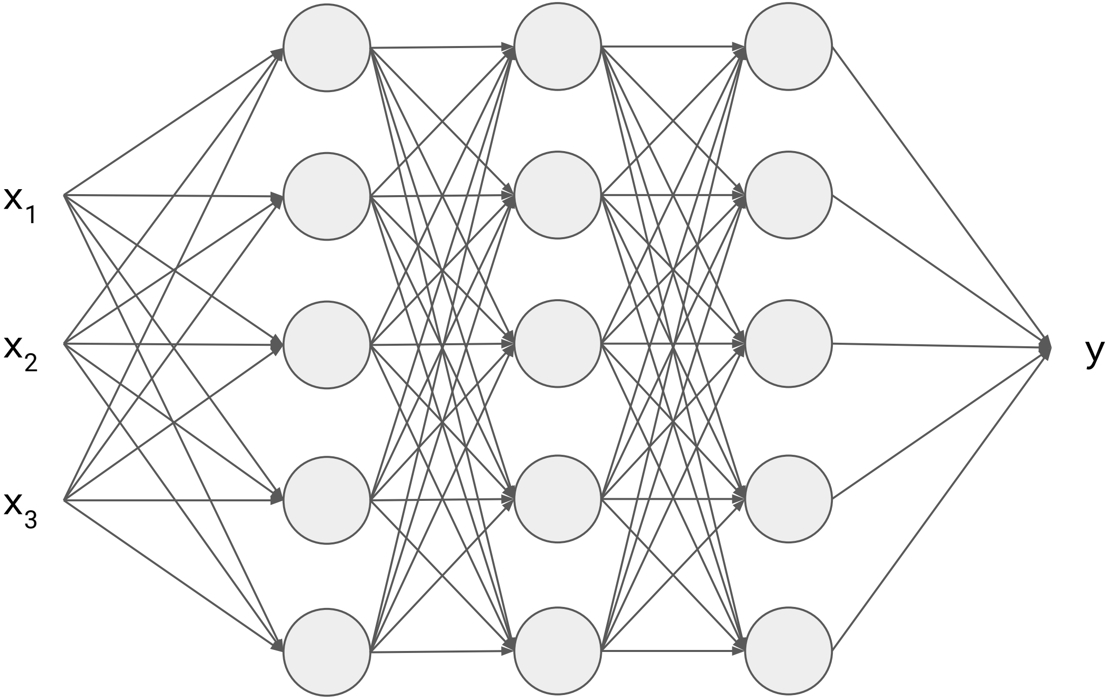


04. Neural nets

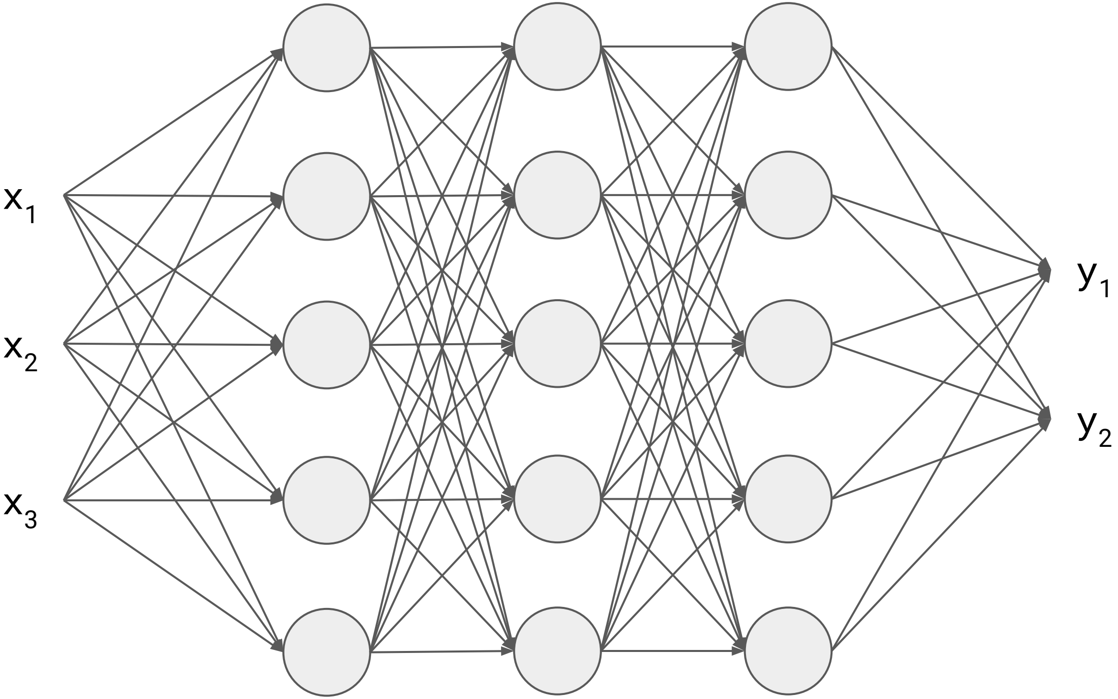


?

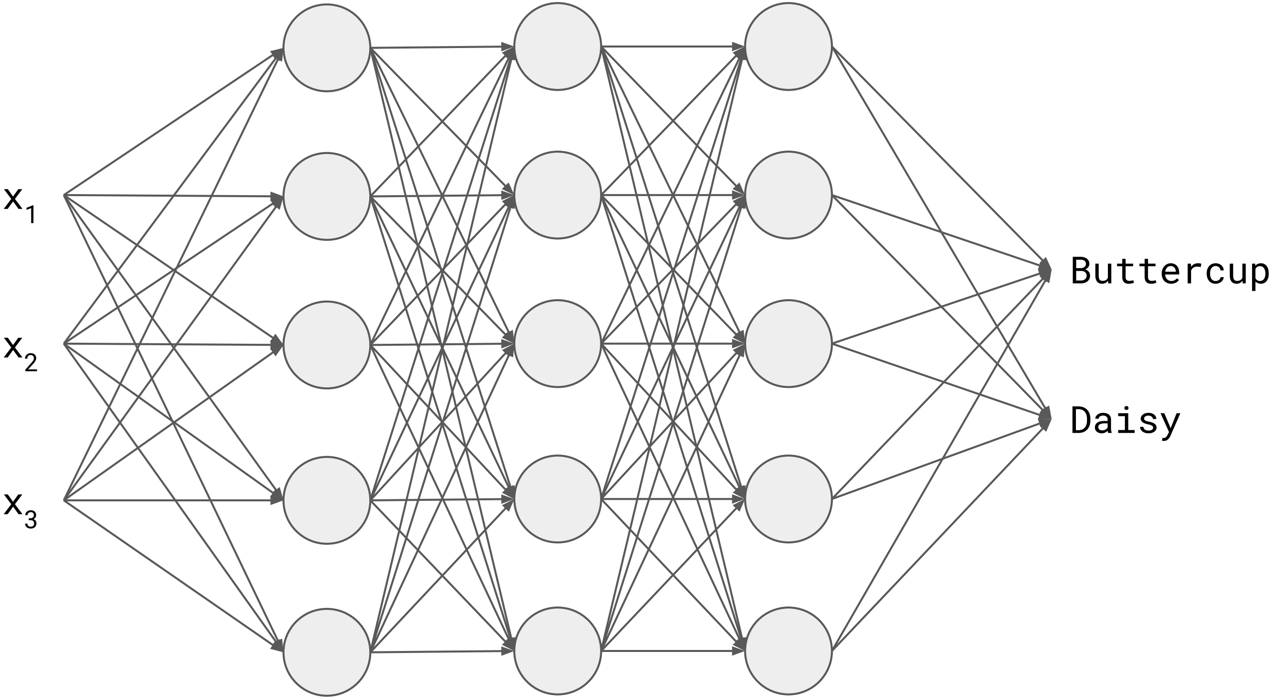
04. Neural nets



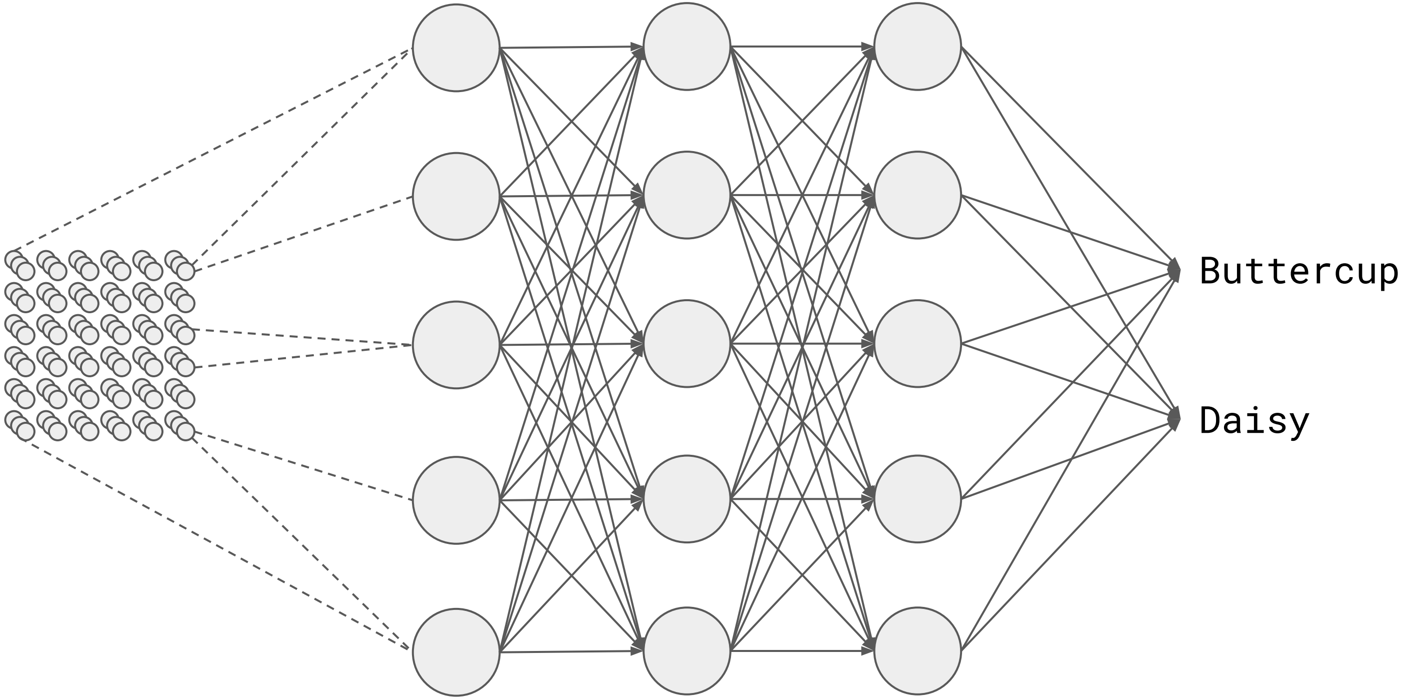
04. Neural nets



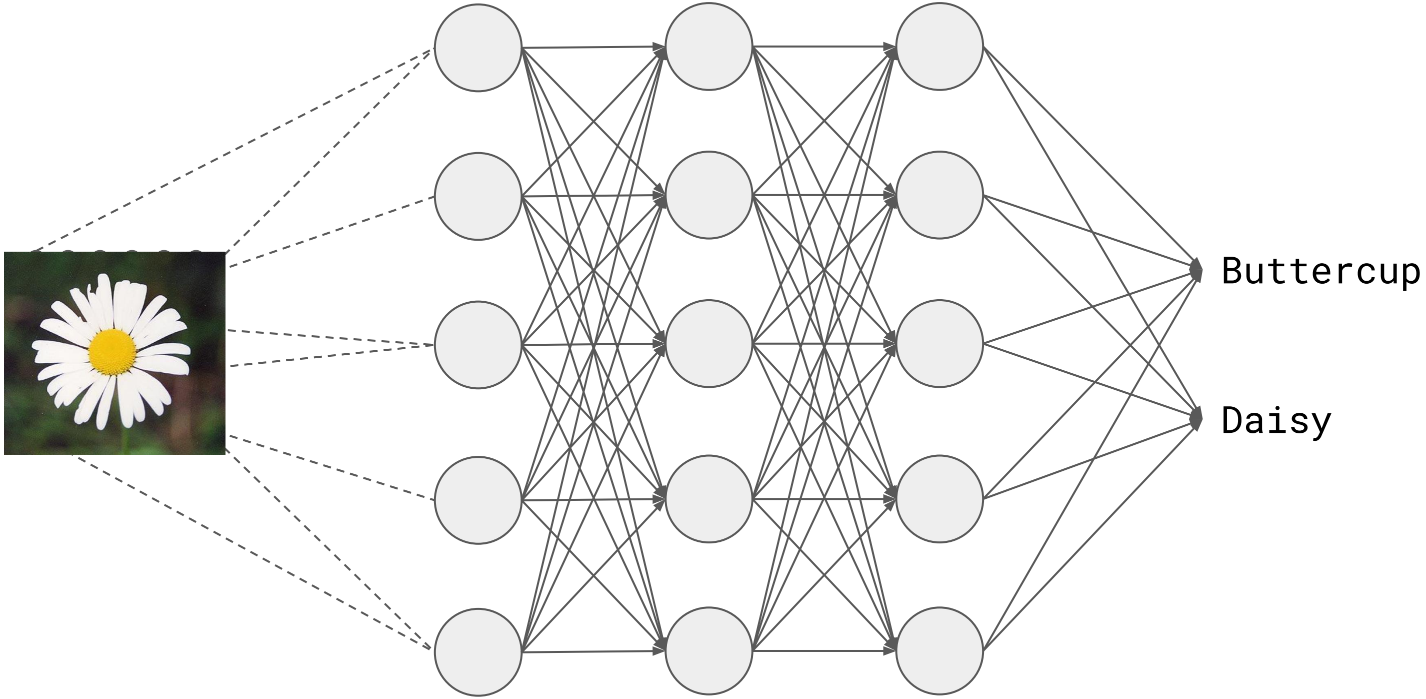
04. Neural nets



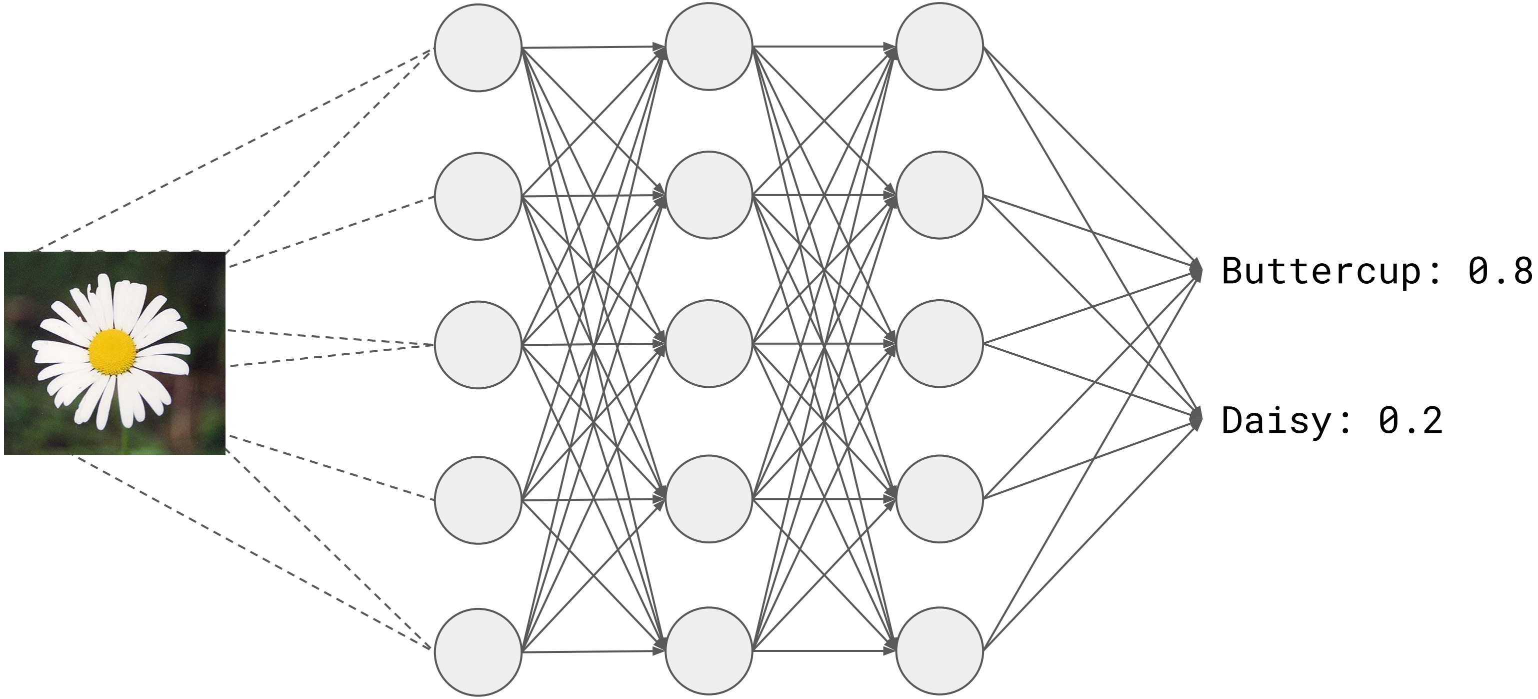
04. Neural nets



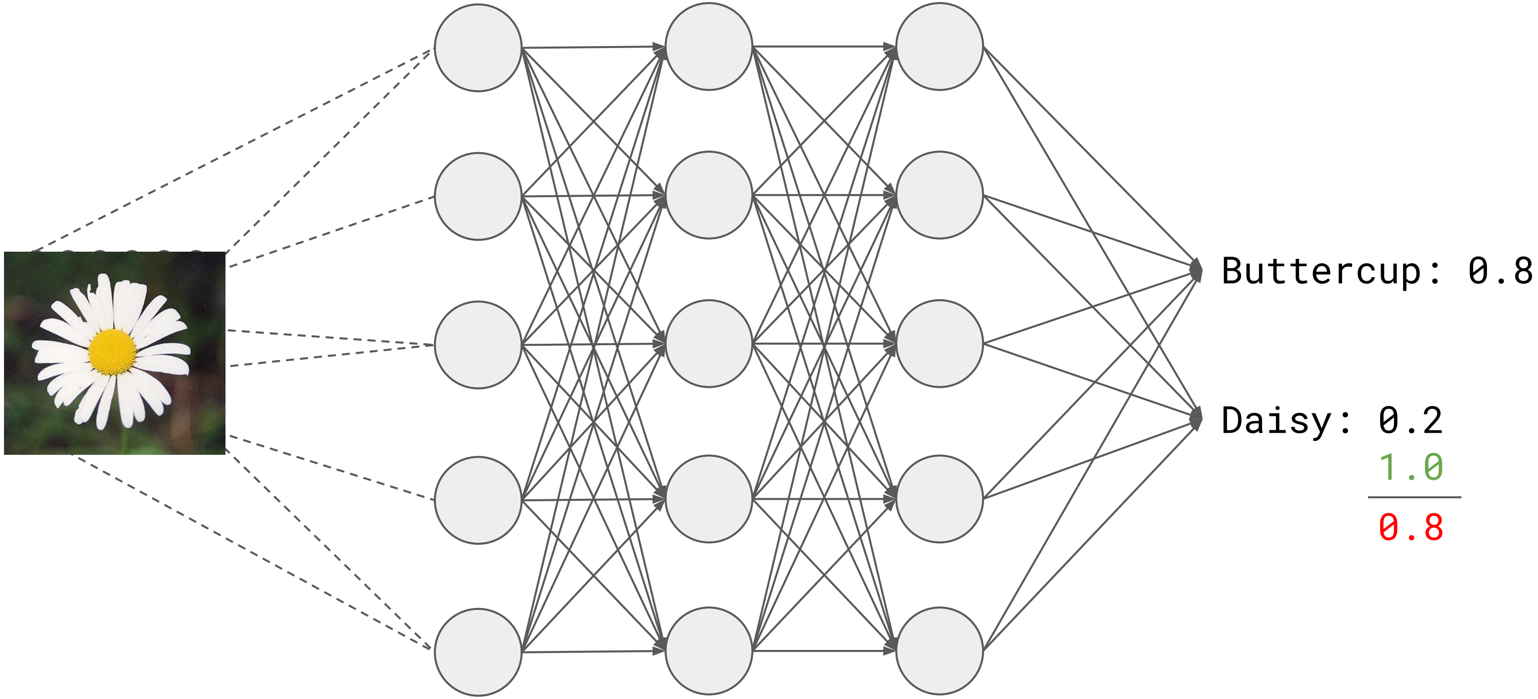
04. Neural nets



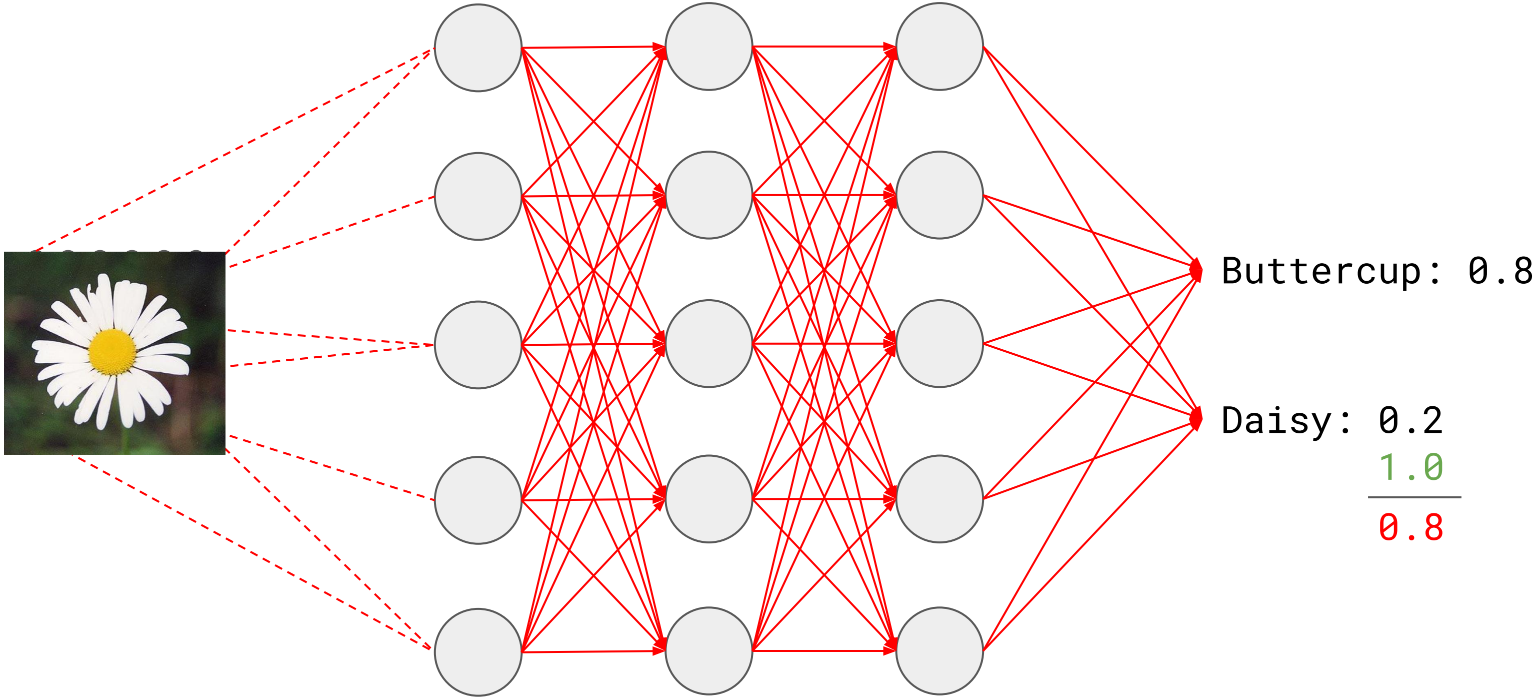
04. Neural nets



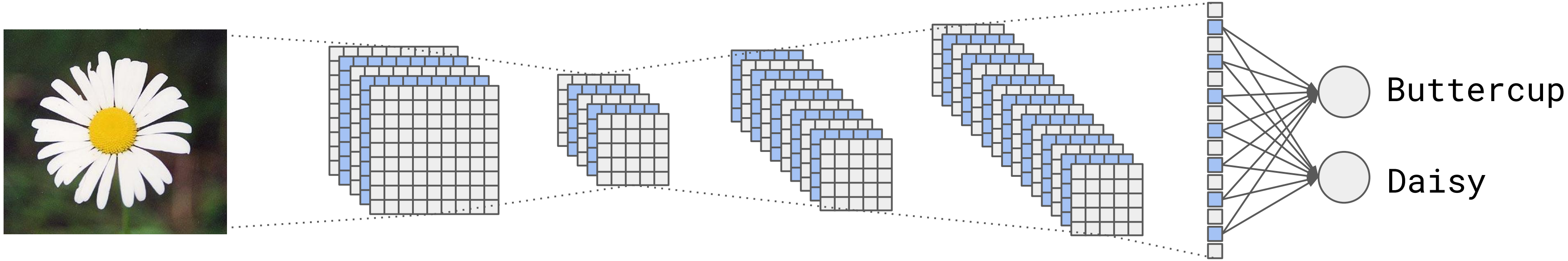
04. Neural nets



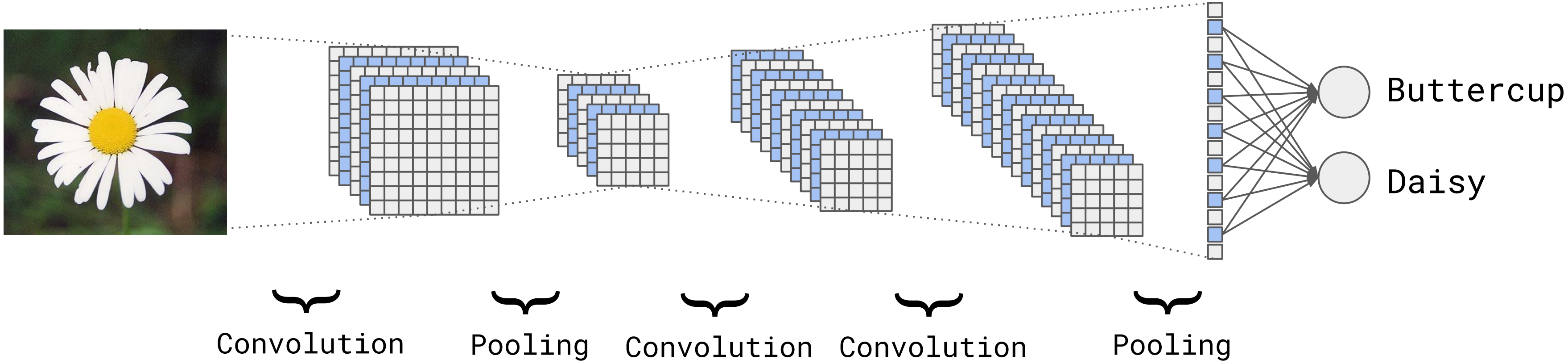
04. Neural nets



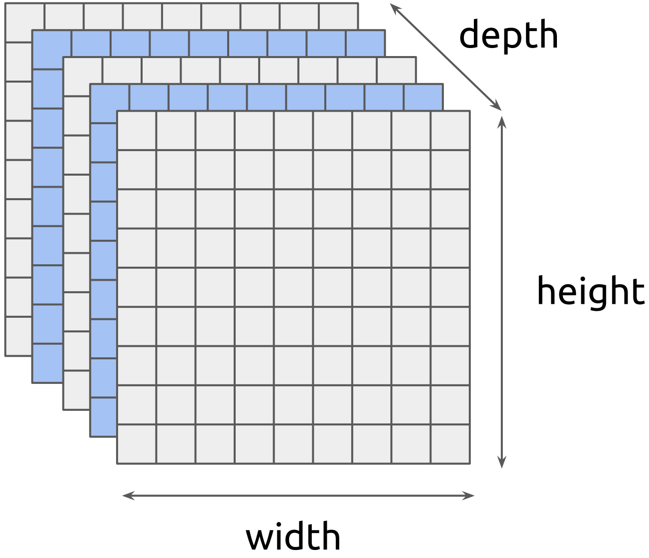
05. Convolutional Neural Nets



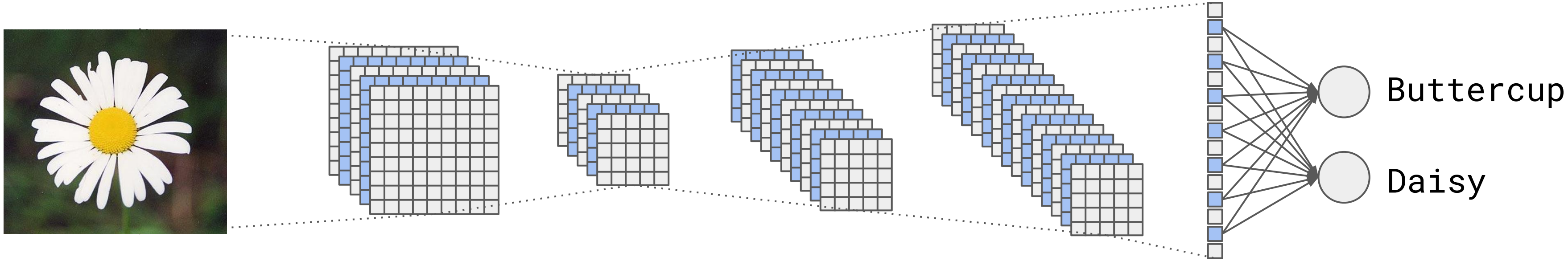
05. Convolutional Neural Nets



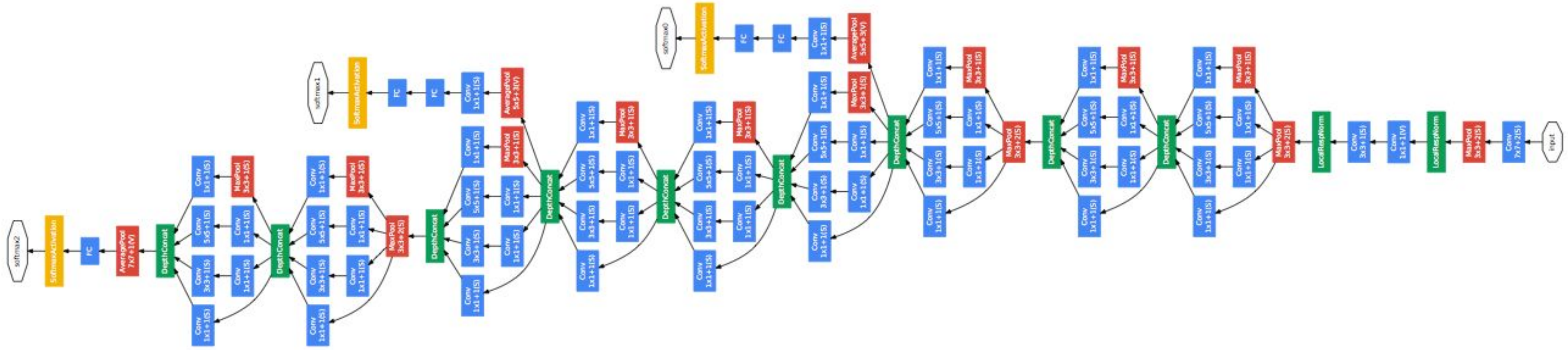
05. Convolutional Neural Nets



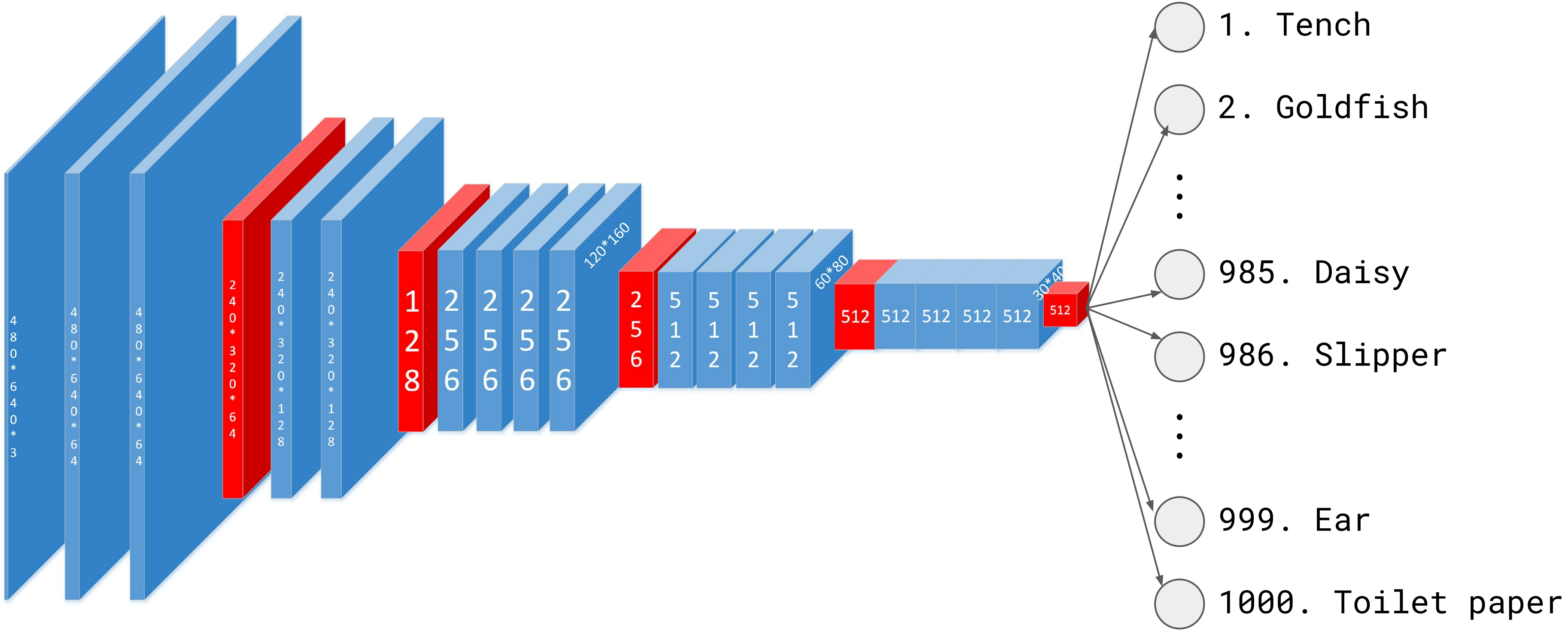
05. Convolutional Neural Nets



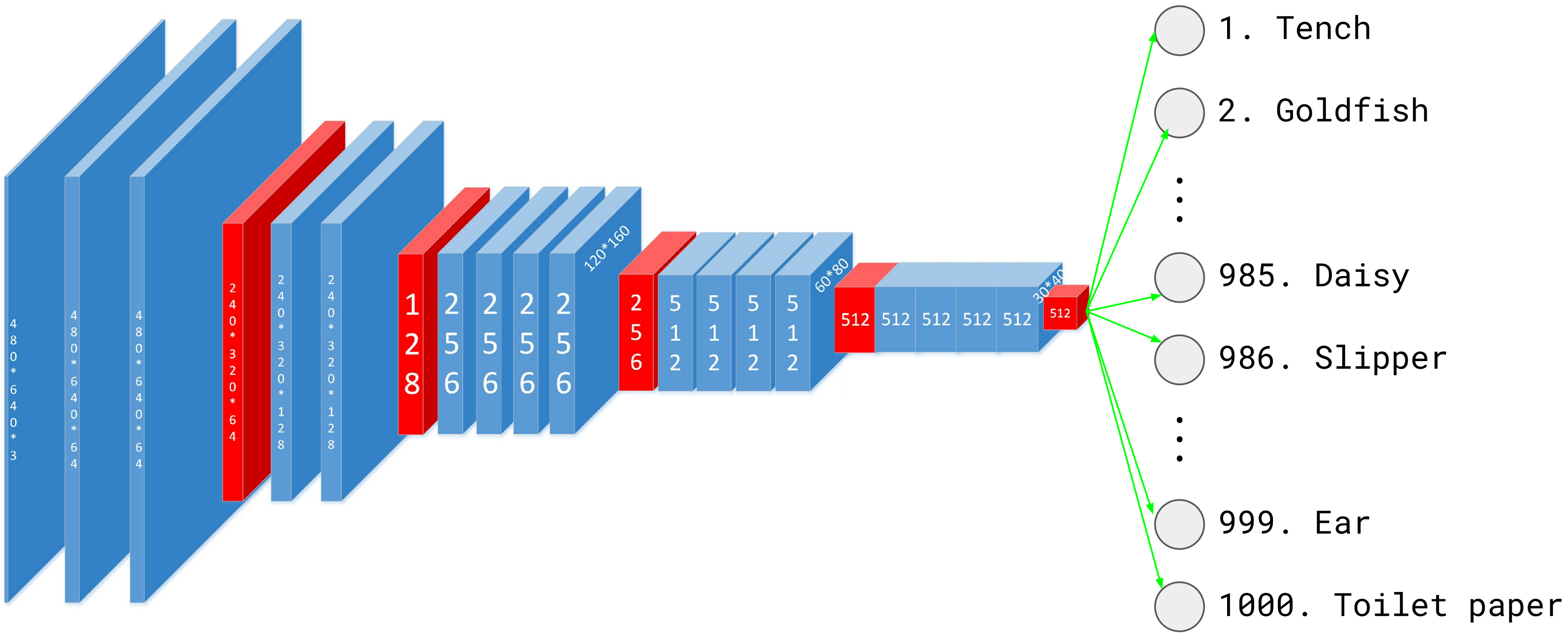
05. Convolutional Neural Nets



06. Transfer learning



06. Transfer learning



1. Tench

2. Goldfish

⋮

985. Daisy

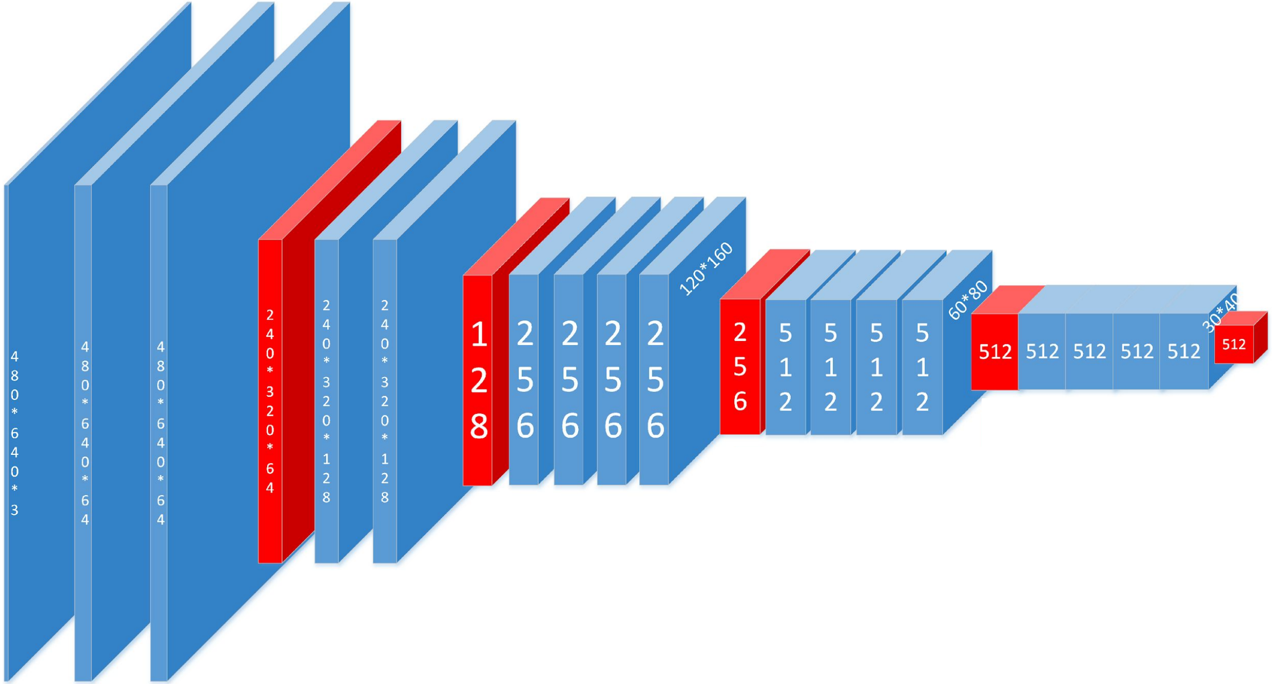
986. Slipper

⋮

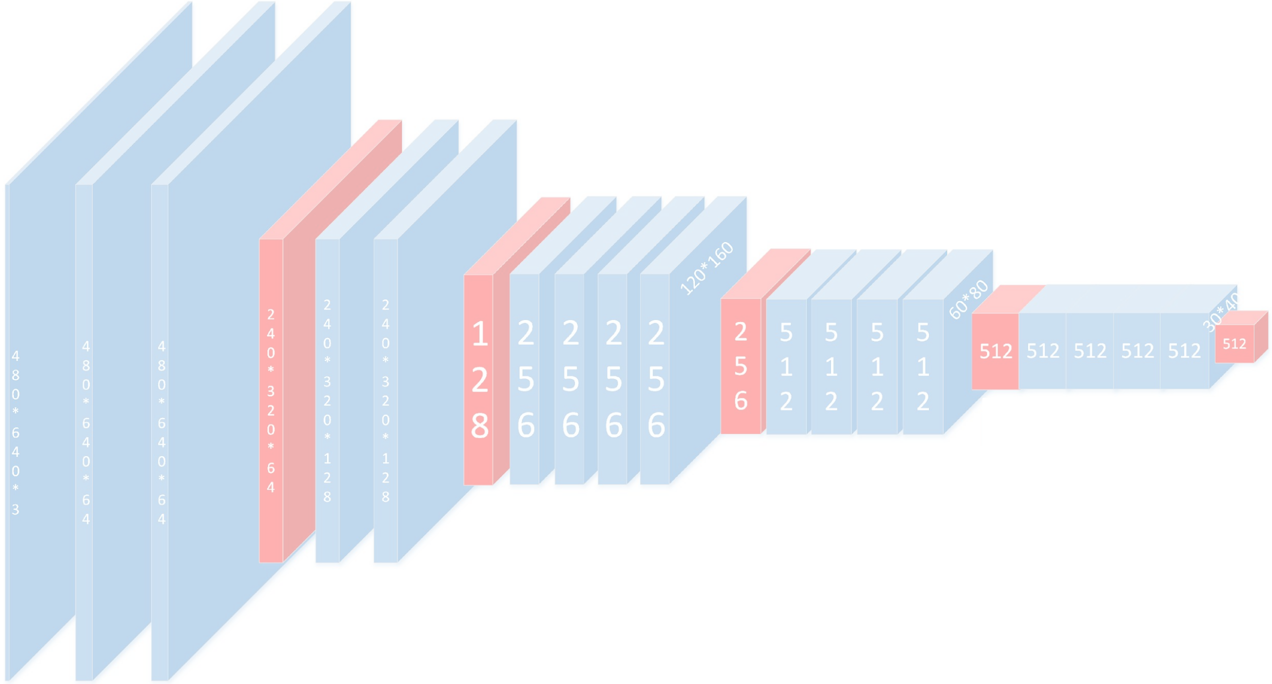
999. Ear

1000. Toilet paper

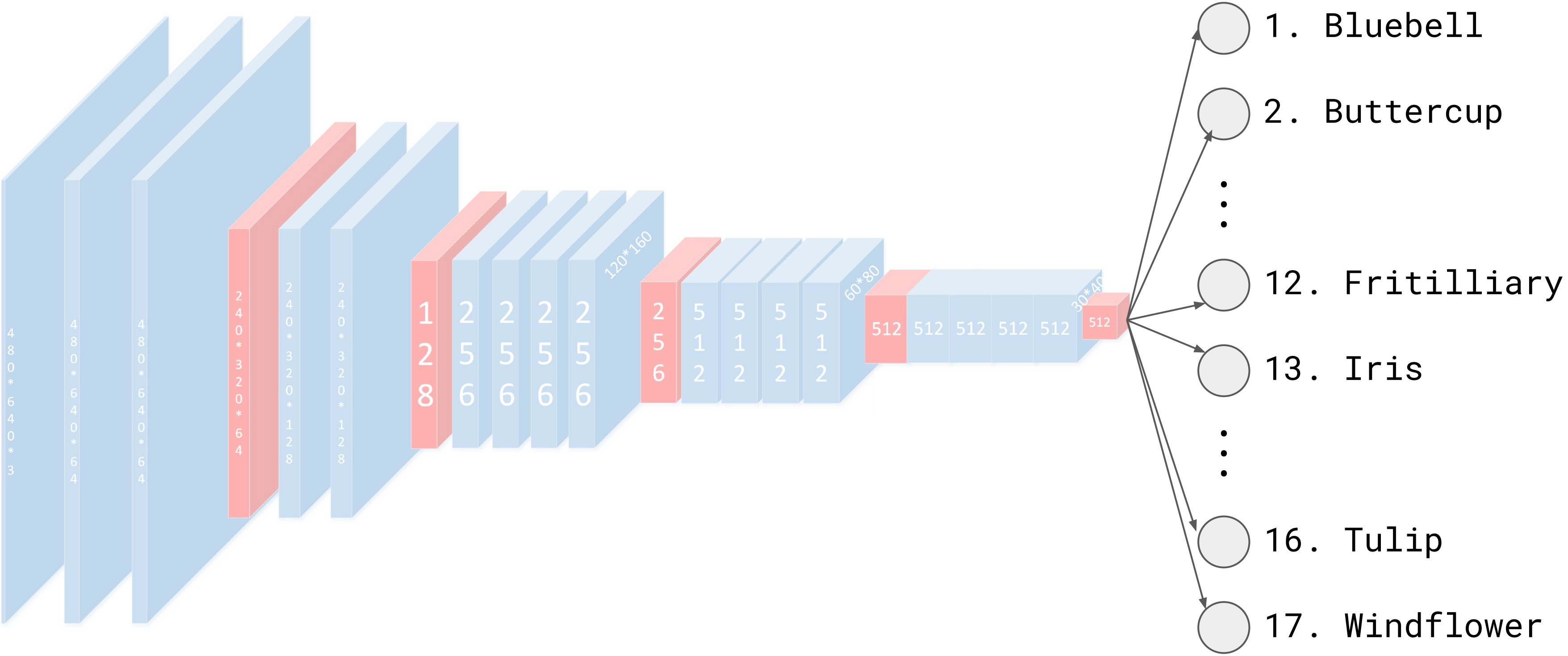
06. Transfer learning



06. Transfer learning



06. Transfer learning



1. Bluebell

2. Buttercup

⋮

12. Fritilliary

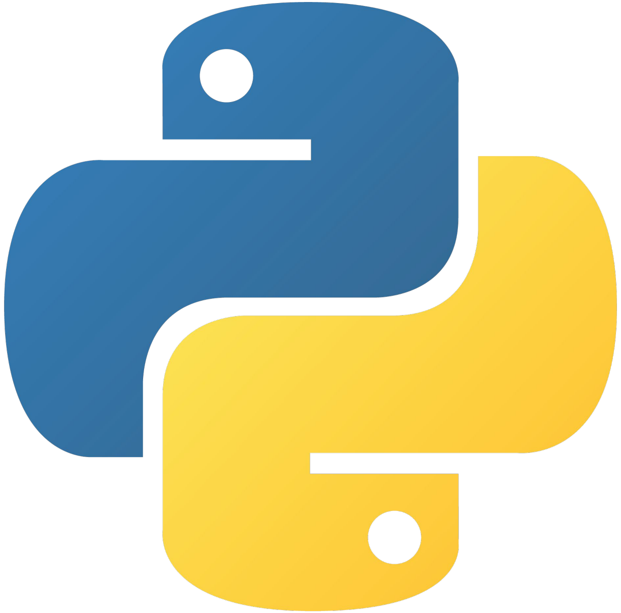
13. Iris

⋮

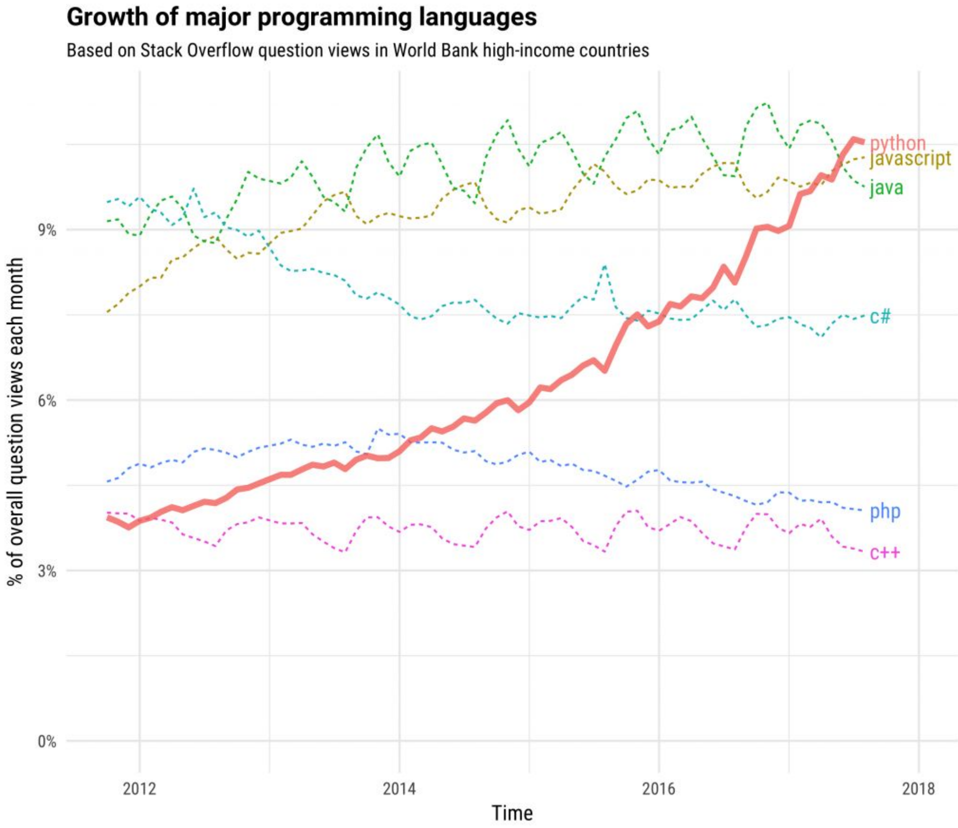
16. Tulip

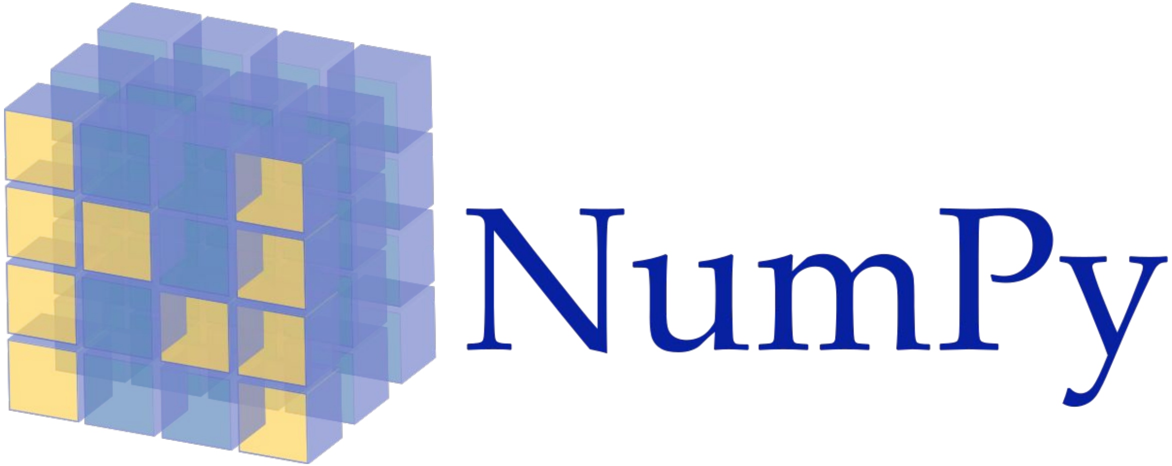
17. Windflower

07. Python

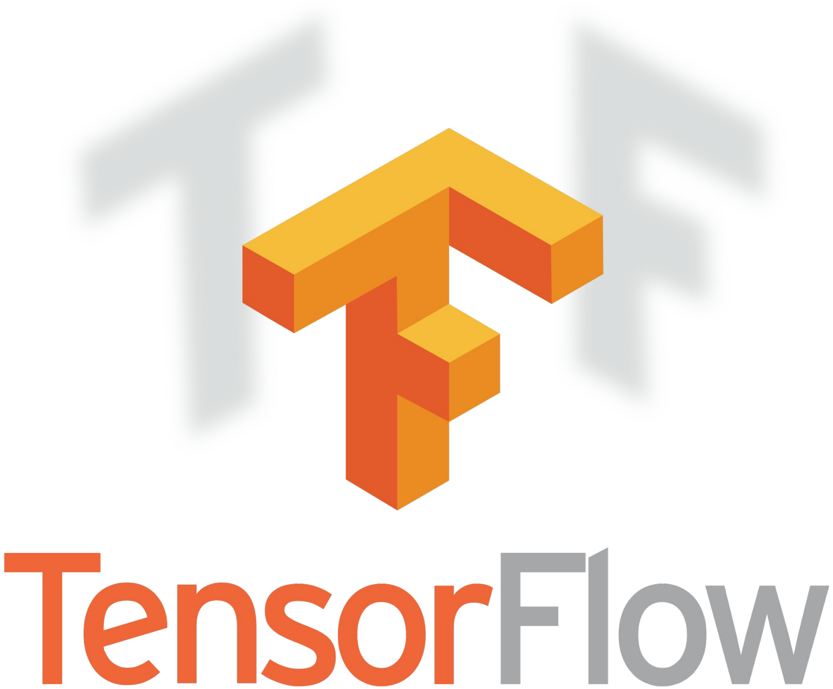


07. Python


Growth of major programming languages
Based on Stack Overflow question views in World Bank high-income countries
% of overall question views each month
0%
3%
6%
9%
2012
2014
2016
2018
Time
python
javascript
java
c#
php
c++


NumPy

07. Python

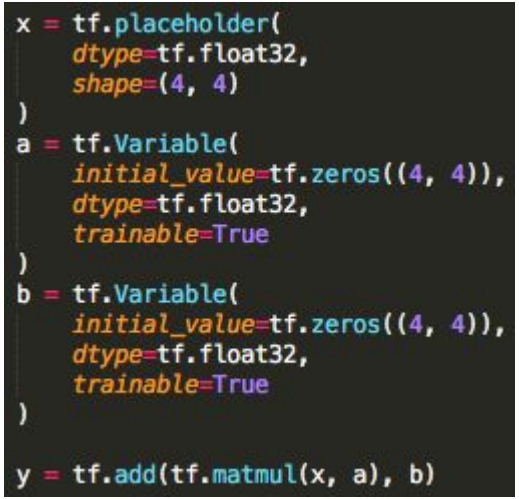


07. Python

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

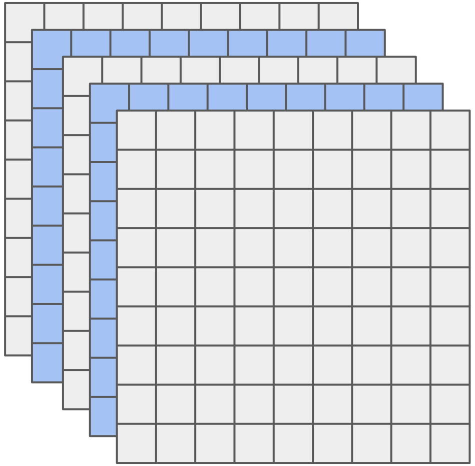
y = ax + b



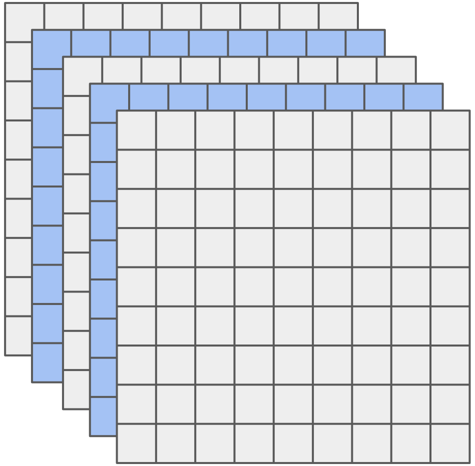
```python
x = tf.placeholder(
    dtype=tf.float32,
    shape=(4, 4)
)
a = tf.Variable(
    initial_value=tf.zeros((4, 4)),
    dtype=tf.float32,
    trainable=True
)
b = tf.Variable(
    initial_value=tf.zeros((4, 4)),
    dtype=tf.float32,
    trainable=True
)

y = tf.add(tf.matmul(x, a), b)
```

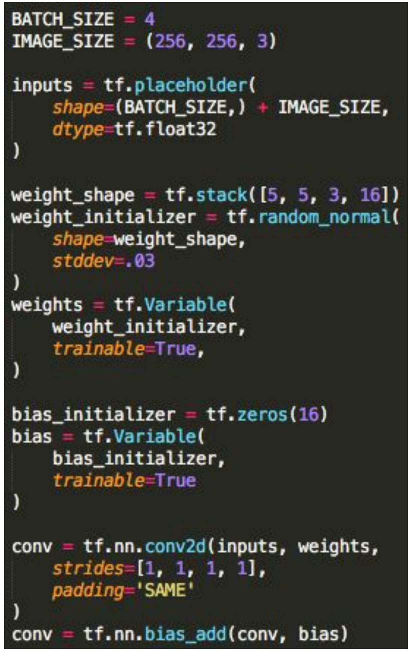
K
Keras

07. Python



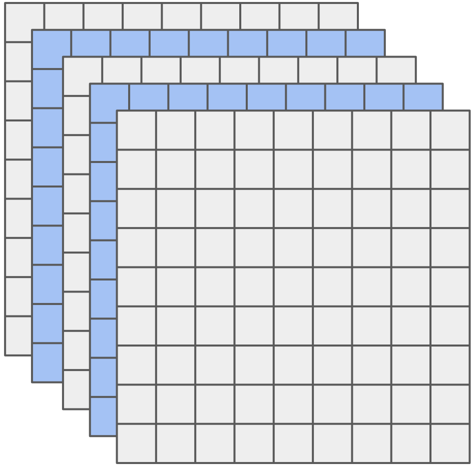
07. Python

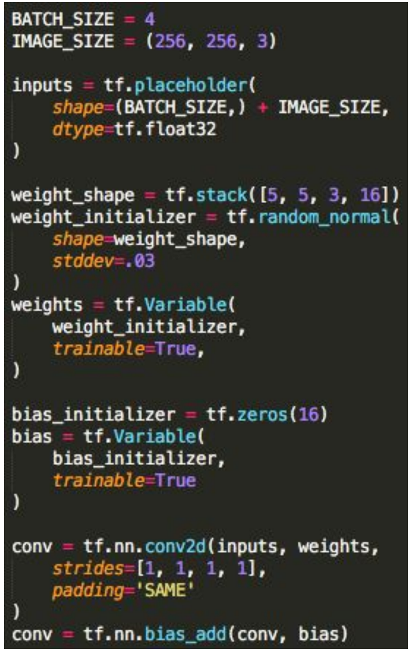




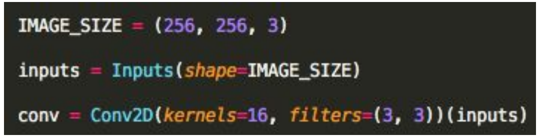
```
BATCH_SIZE = 4
IMAGE_SIZE = (256, 256, 3)

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



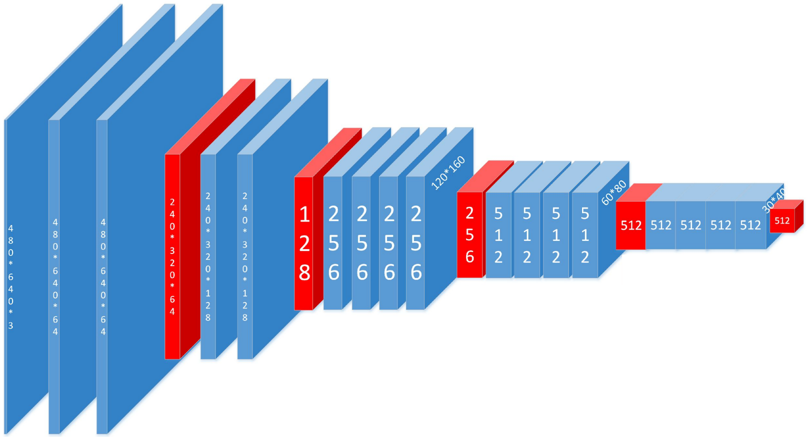


```python
BATCH_SIZE = 4
IMAGE_SIZE = (256, 256, 3)

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)
```

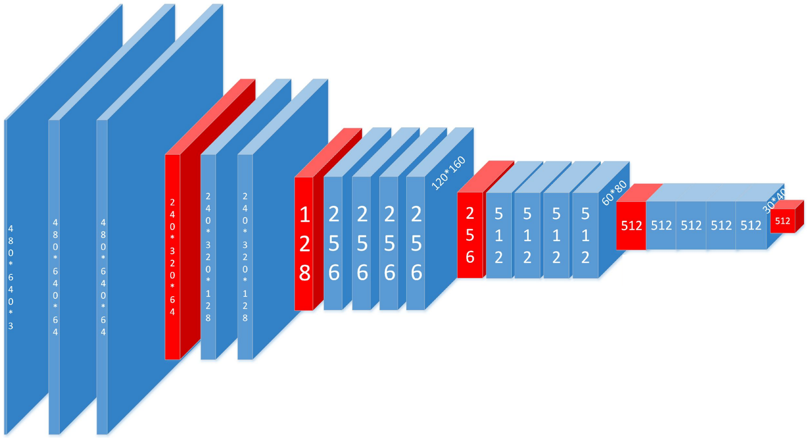


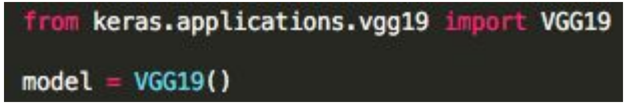
```python
IMAGE_SIZE = (256, 256, 3)

inputs = Inputs(shape=IMAGE_SIZE)

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

07. Python


480*640*3
480*640*64
480*640*64
240*320*64
240*320*128
240*320*128
128
256
256
256
256
120*160
256
512
512
512
512
60*80
512
512
512
512
512
30*40
512

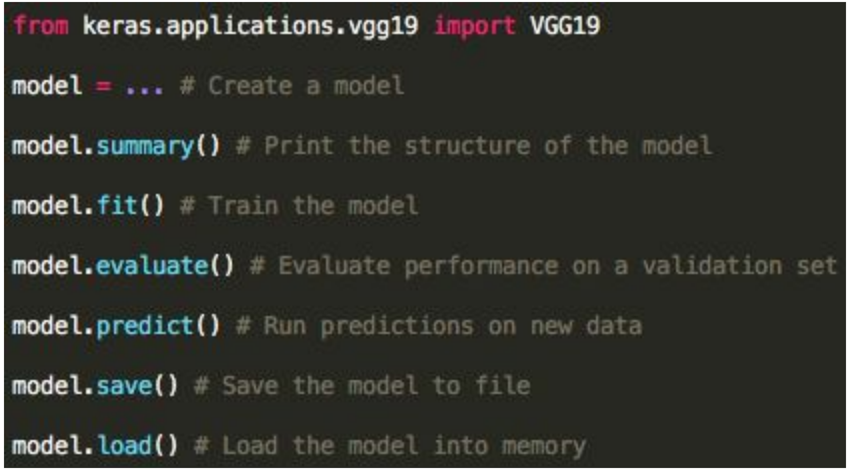
07. Python





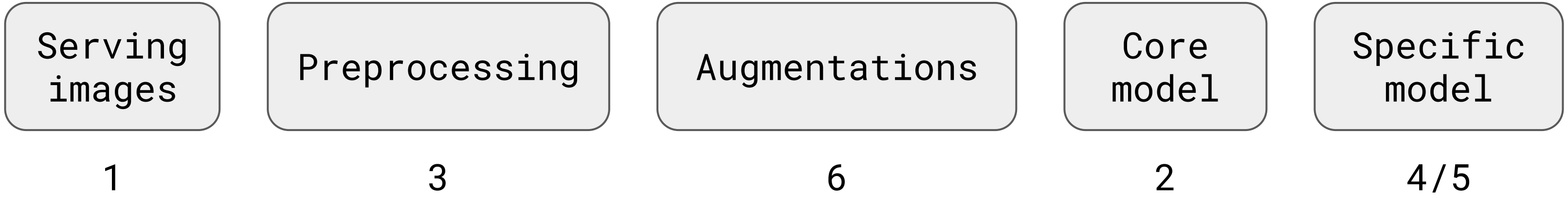
```python
from keras.applications.vgg19 import VGG19

model = VGG19()
```

07. Python



```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

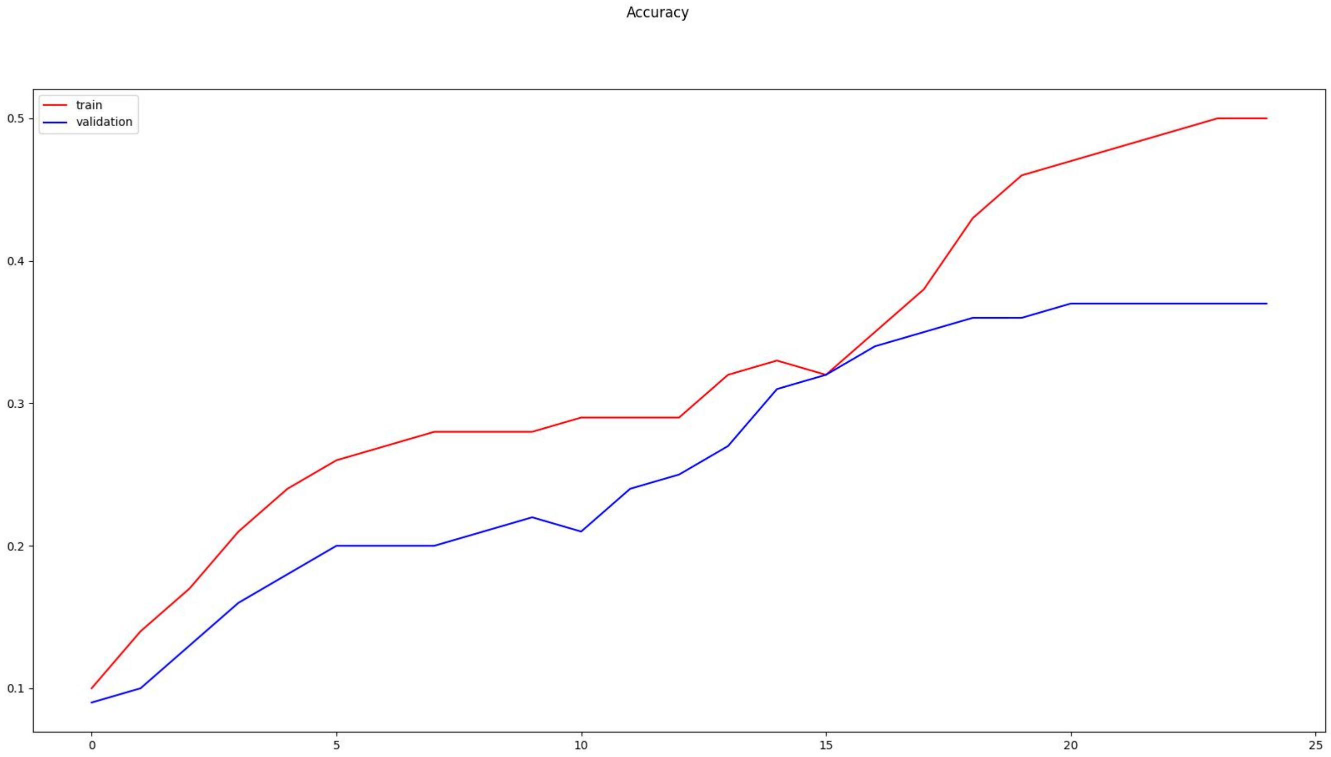


Serving images	Preprocessing	Augmentations	Core model	Specific model
1	3	6	2	4/5

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


Accuracy
train
validation