Image Classification

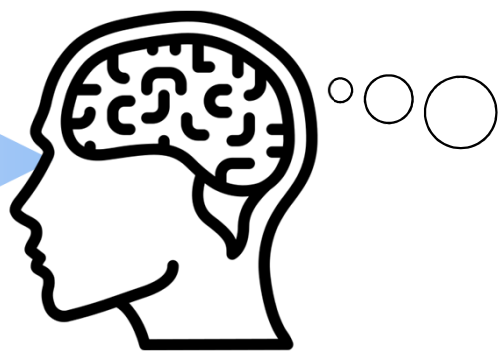
Notes based on
Machine Learning for the Web by YiNing at ITP, NYU,
How to build a Teachable Machine with TensorFlow.js by Nikhil Thorat from Google Brain, PAIR,
and Daniel Shiffman from the Coding Train

Classification: Basic Human task

Classification: Basic Human task

Input: image

Resolution:
800 x 600 x 3 (RGB)



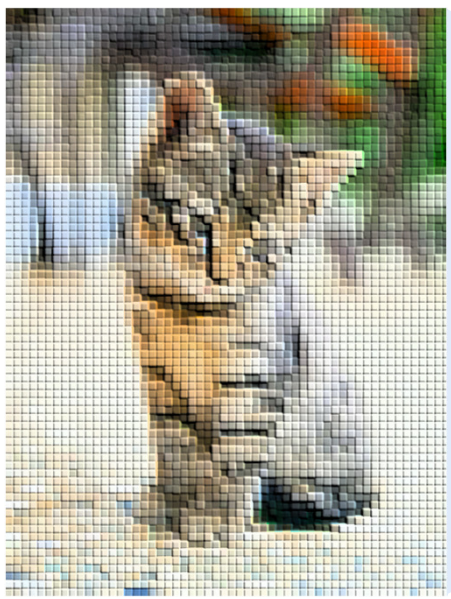
Output:

cat
bird
deer
dog
truck

All images from https://unsplash.com/s/photos/cat

Classification: Basic AI task

Input: image



Resolution:
800 x 600 x 3 (RGB)

AI Magic Box

Output:	Probability:
cat	0.82
bird	0.02
deer	0.04
dog	0.10
truck	0.02

All images from https://unsplash.com/s/photos/cat

AI Classification: Challenges

- Change view angle

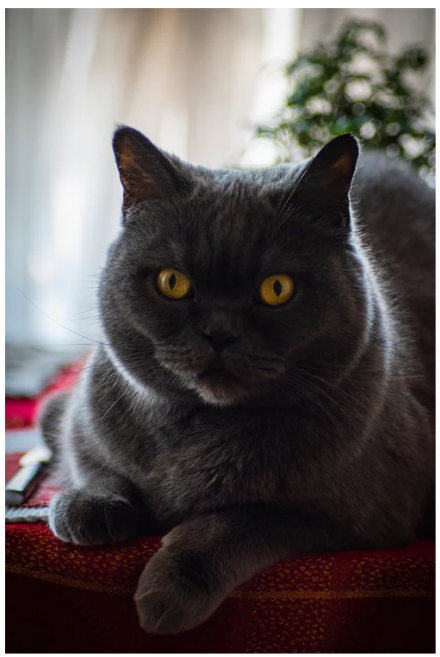
AI Classification: Challenges

- Variation within class

AI Classification: Challenges

- Sub-classes (different cat breeds)

Maine Coon

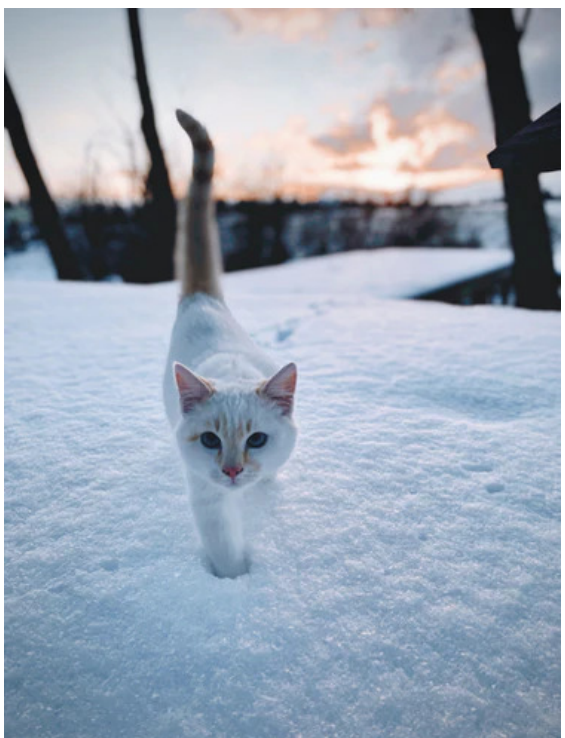


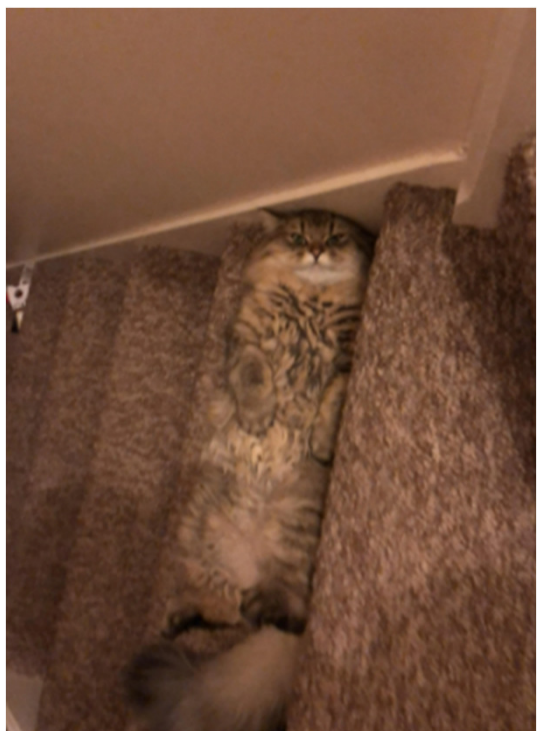
Bristish Shorthair

Grumpy Cat
(Mixed breed)
2012-2019

AI Classification: Challenges

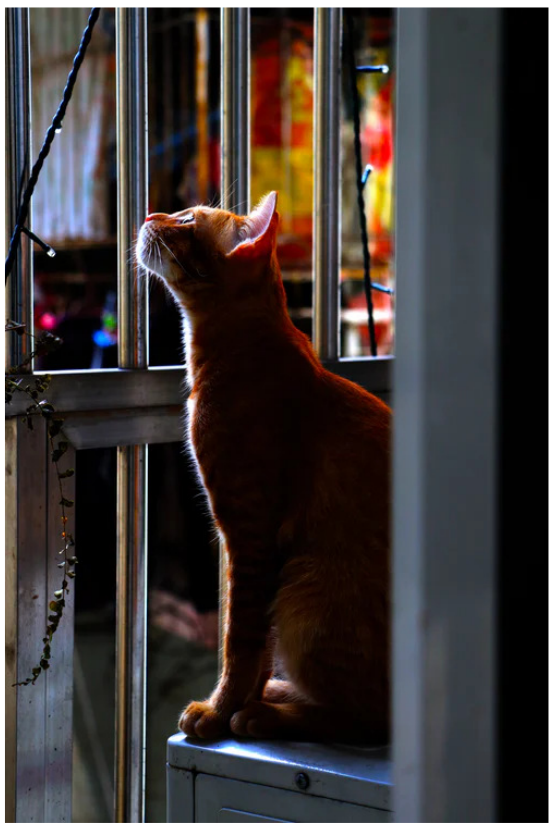
- Background blending / camouflage





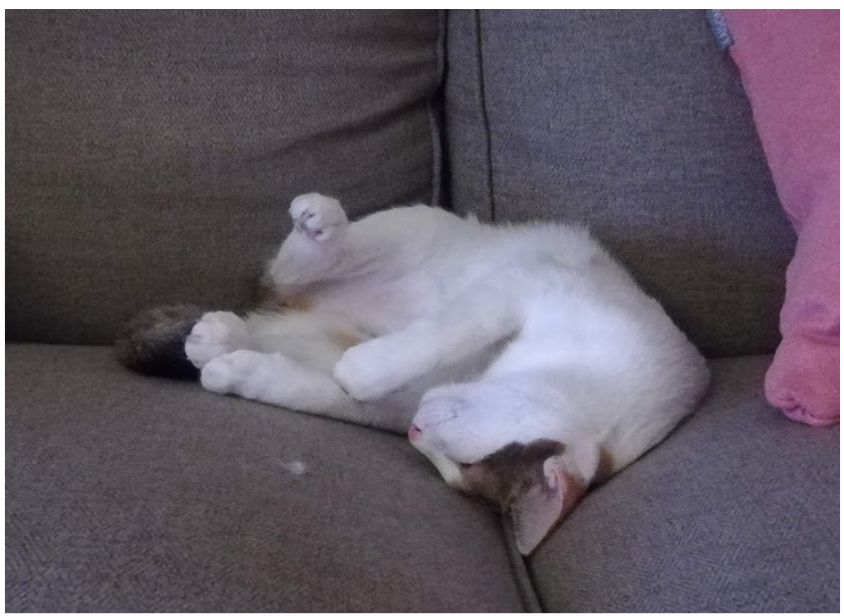
AI Classification: Challenges

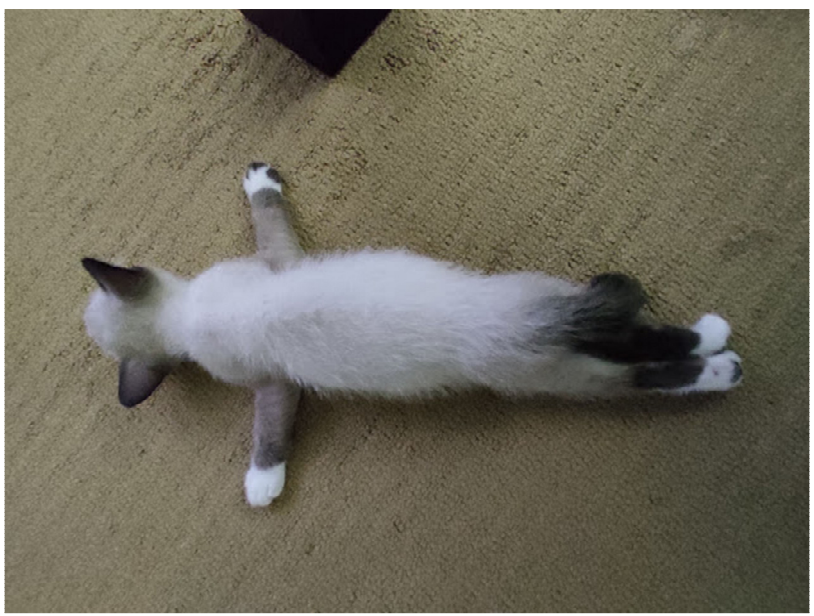
- Lighting change



AI Classification: Challenges

- Deformation





AI Classification: Challenges

- Occlusion

Image Classification: Building Blocks for other tasks.

- Object detection

Output: bounding boxes and class

Image Classification: Building Blocks for other tasks.

- Object segmentation

Output: segments and class

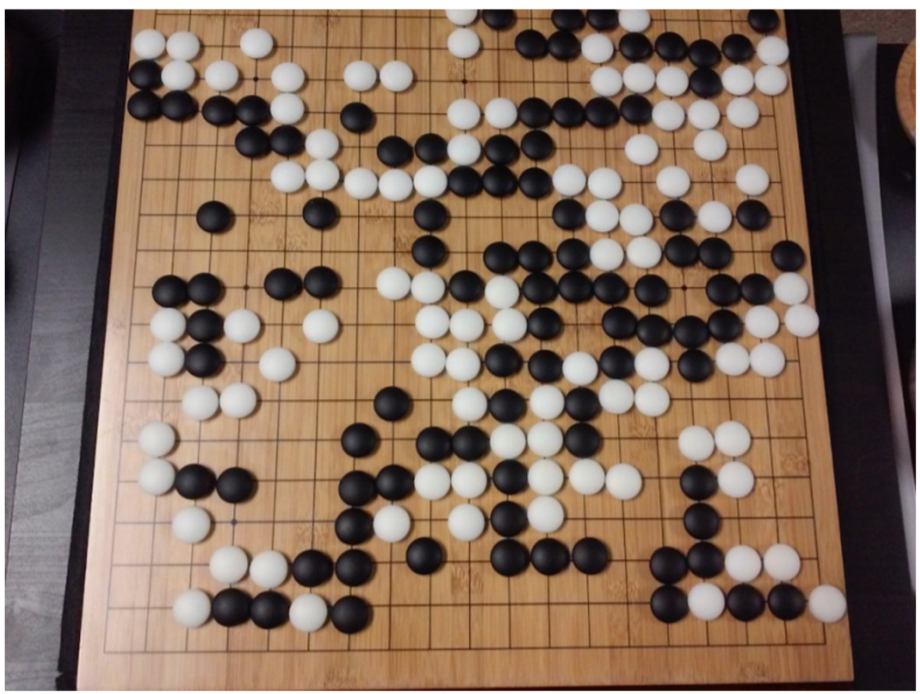
Image Classification: Building Blocks for other tasks.

- Image captioning

Output: description: There are two cars and a bike on the road

Image Classification: Building Blocks for other tasks.

- Playing go

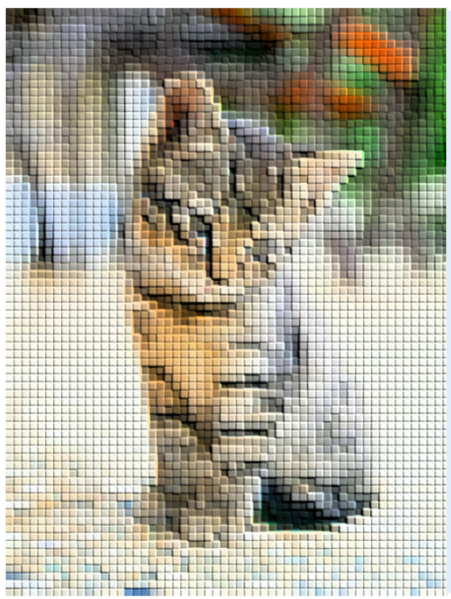


(1, 1)
(1, 2)
…
(1, 19)
…
(19, 19)

What's the next move?

Classification: Basic AI task

Input: image



Resolution:
800 x 600 x 3 (RGB)

AI Magic Box

Output:

Class:	Probability:
cat	0.82
bird	0.02
deer	0.04
dog	0.10
truck	0.02

All images from https://unsplash.com/s/photos/cat

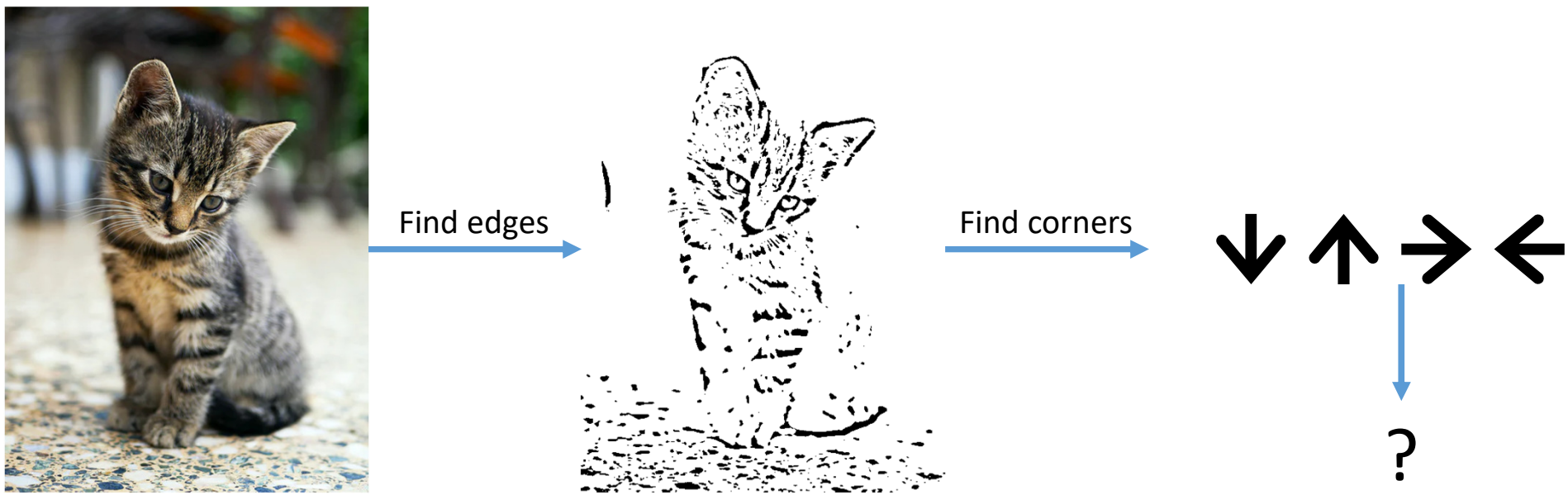
An Image Classifier

```
def classify_image(image):
    # some AI magic
    return class_label
```

Unlike e.g. sorting a list of numbers,

no obvious way to hard-code the algorithm for recognizing a cat, or other classes.

You can try edge detection…



Machine Learning: Data driven approach

1. Dataset – Collect a dataset of images with labels
2. Train – Use ML to train classifier
3. Test/deploy – evaluate classifer on new images

```
def train(images, labels):
  # Machine learning!
  return model

def predict(model, test_image):
  # use model to predict label
  return predicted_label
```

Example training set

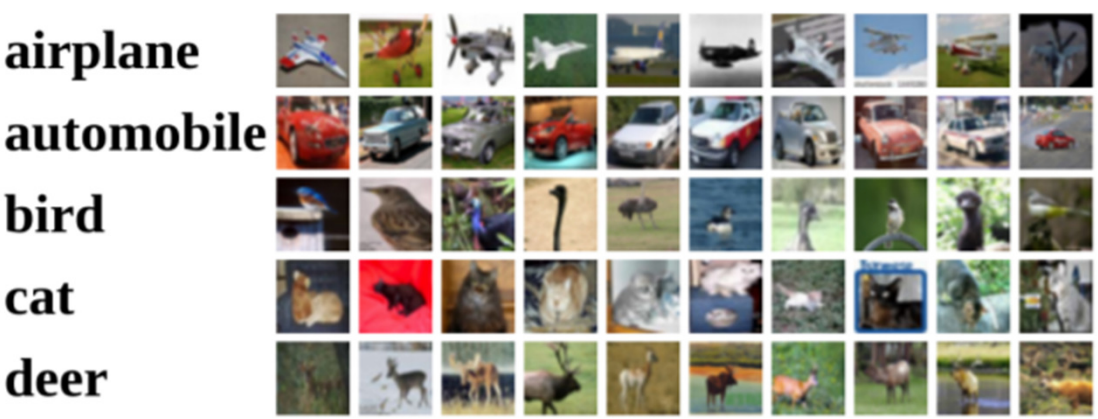
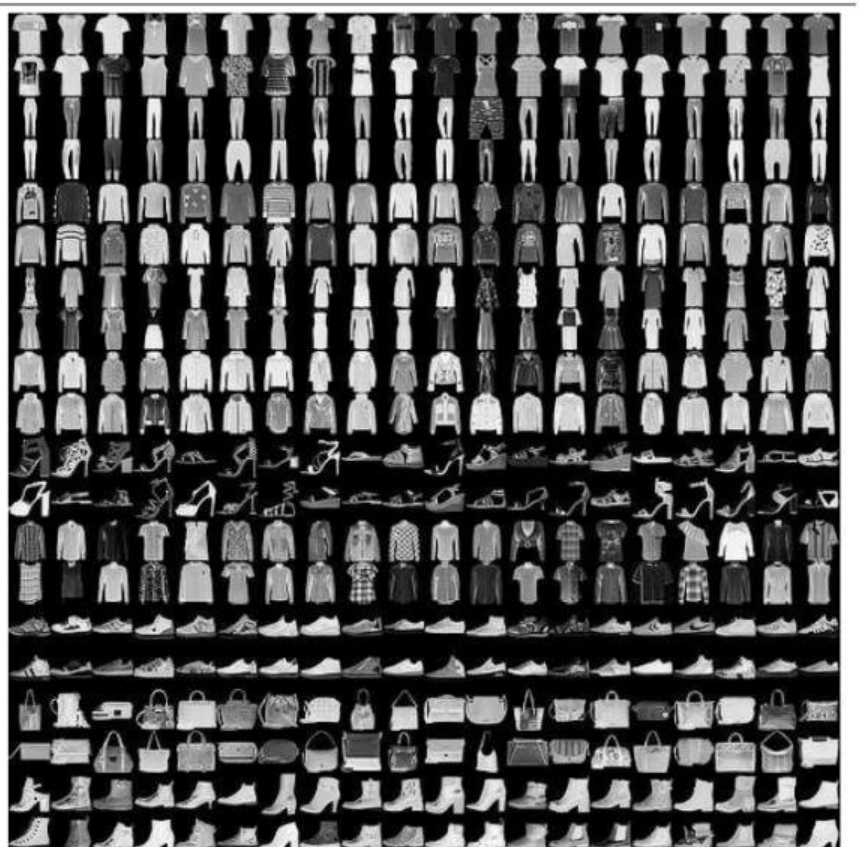


Image Classification Datasets: MNIST

10 classes: Digits 0 to 9
28x28 grayscale images
50k training images
10k test images

Results from MNIST often do not hold on more complex datasets!

Image Classification Datasets: Fashion MNIST

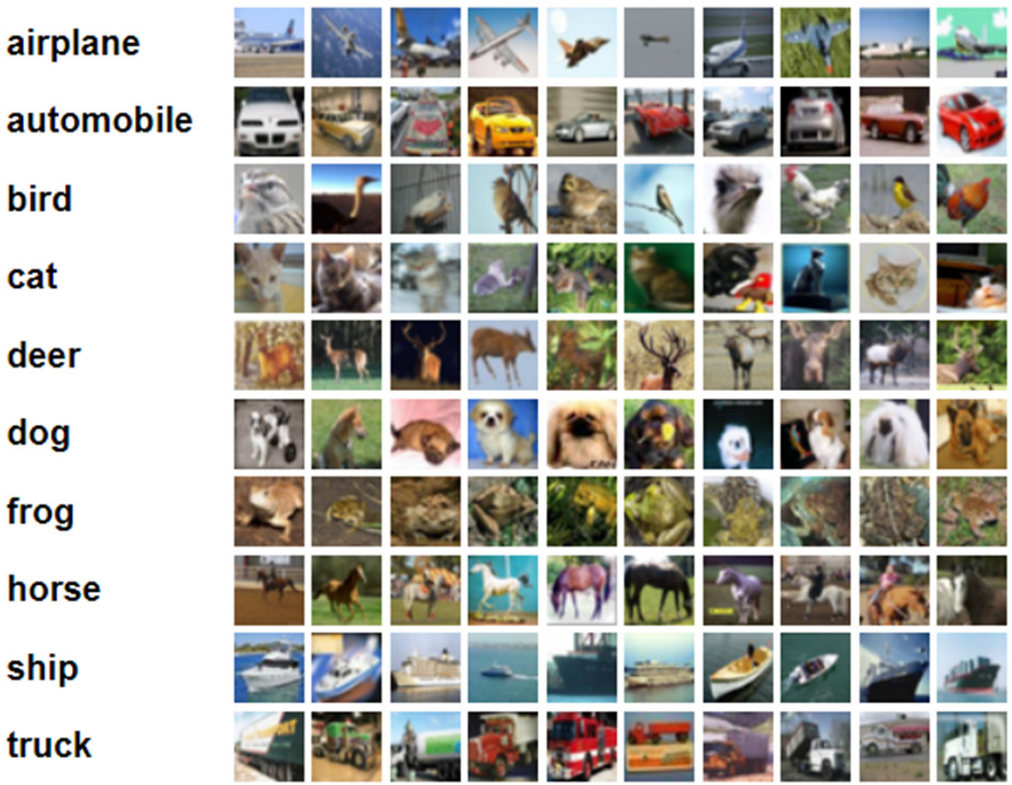
Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	

10 classes
28x28 grayscale images
50k training images
10k test images

https://github.com/zalandoresearch/fashion-mnist

Results from MNIST often do not hold on more complex datasets!

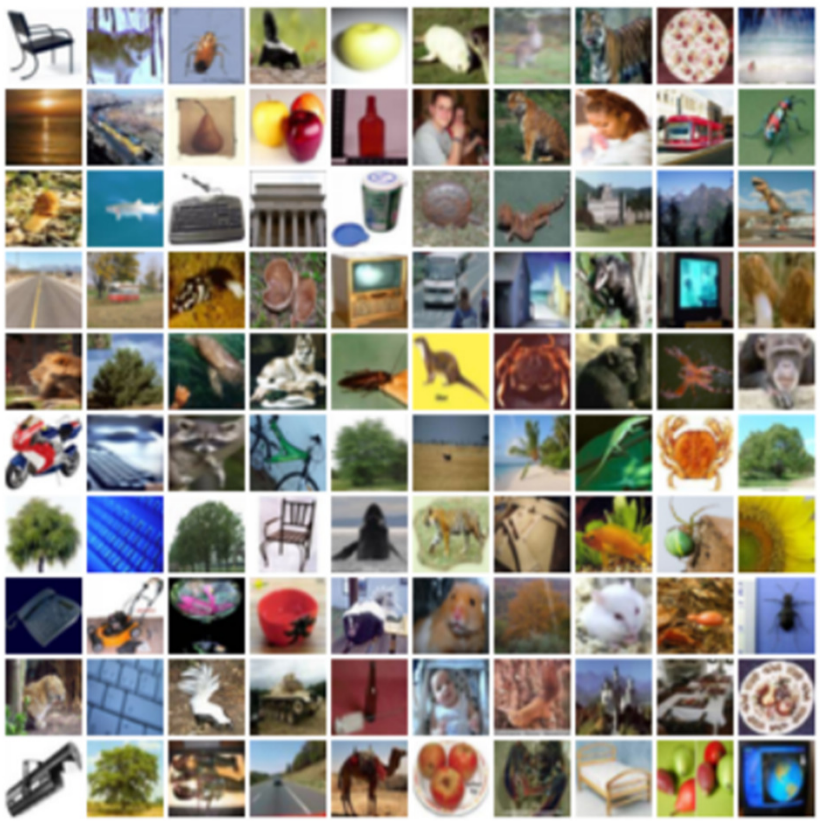
Image Classification Datasets: CIFAR10



10 classes
50k training images (5k per class)
10k testing images (1k per class)
32x32 RGB images

https://www.cs.toronto.edu/~kriz/cifar.html

Image Classification Datasets: CIFAR100



https://www.cs.toronto.edu/~kriz/cifar.html

100 classes
50k training images (500 per class)
10k testing images (100 per class)
32x32 RGB images

20 superclasses with 5 classes each:

Aquatic mammals:
beaver, dolphin, otter, seal, whale

Trees:
Maple, oak, palm, pine, willow

Image Classification Datasets: ImageNet

1000 classes

~1.3M training images (~1.3K per class)
50K validation images (50 per class)
100K test images (100 per class)

Performance metric: **Top 5 accuracy**

Algorithm predicts 5 labels for each image; one of them needs to be right

Image Classification Datasets: MIT Places

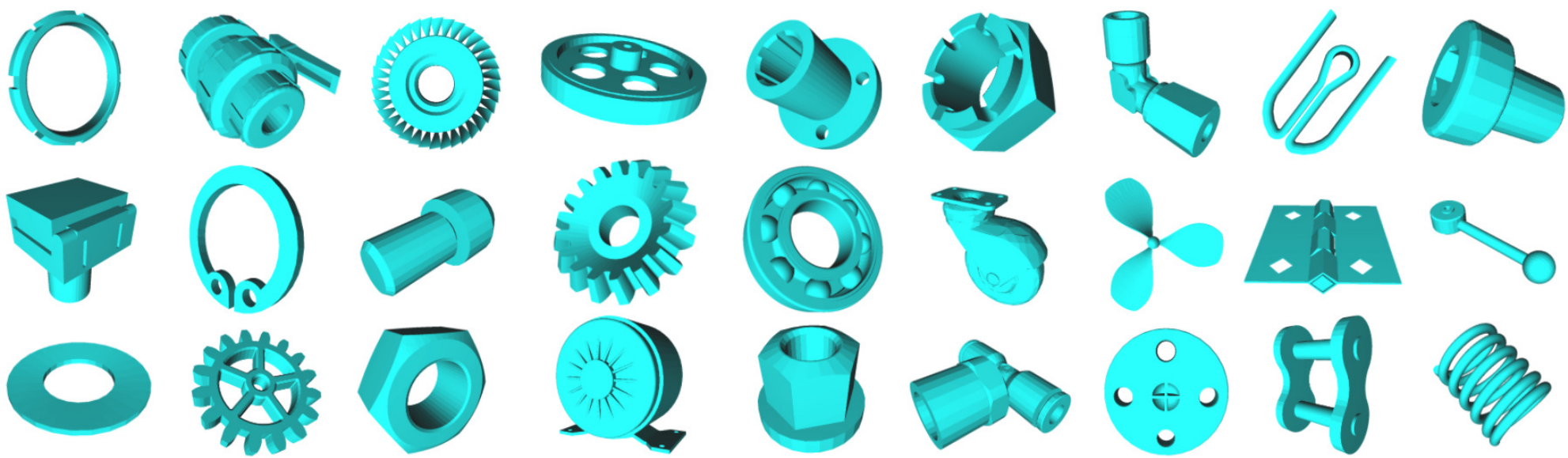
365 classes of different scene types

~8M training images
18.25K val images (50 per class)
328.5K test images (900 per class)

Images have variable size, often resized to **256x256** for training

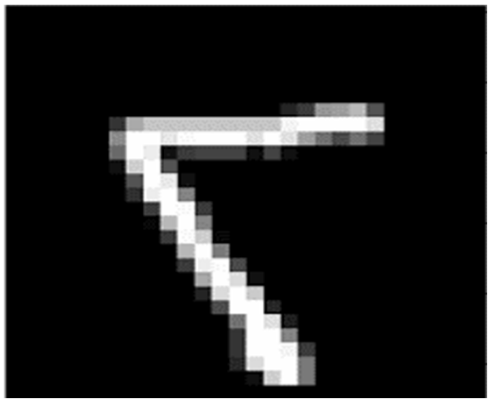
A Large-scale Mechanical Components Benchmark for Deep Neural Networks

- https://mechanical-components.herokuapp.com/
- https://www.youtube.com/watch?v=EaKo7ky15uA

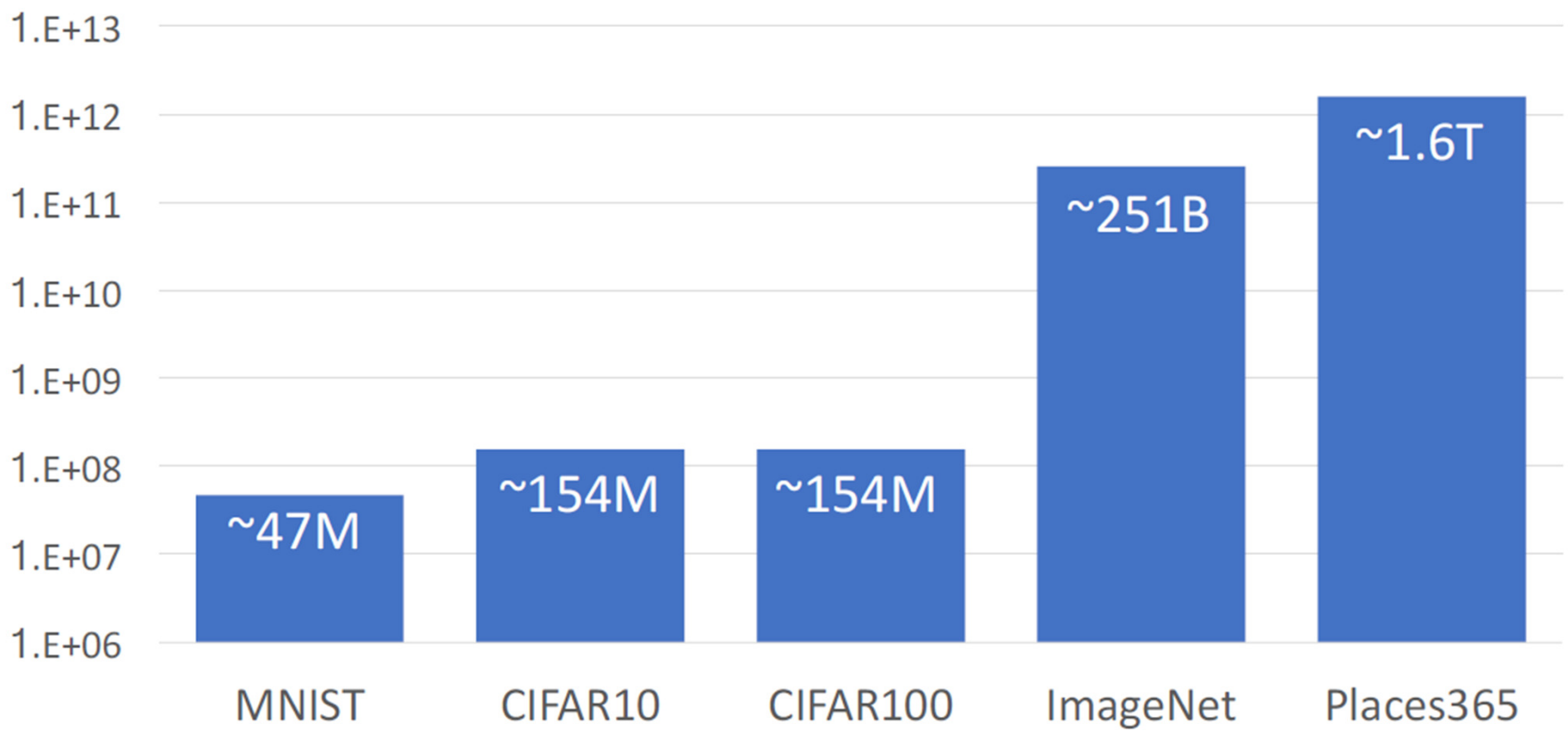


Datasets

1. MNIST
2. CIFAR10
3. CIFAR100
4. ImageNet
5. MIT Places



Datasets: Number of Training Pixels



Your first classifier: Nearest Neighbour

```
def train(images, labels):
  # Machine learning!
  return model
```

Memorize all data and labels

```
def predict(model, test_image):
  # use model to predict label
  return predicted_label
```

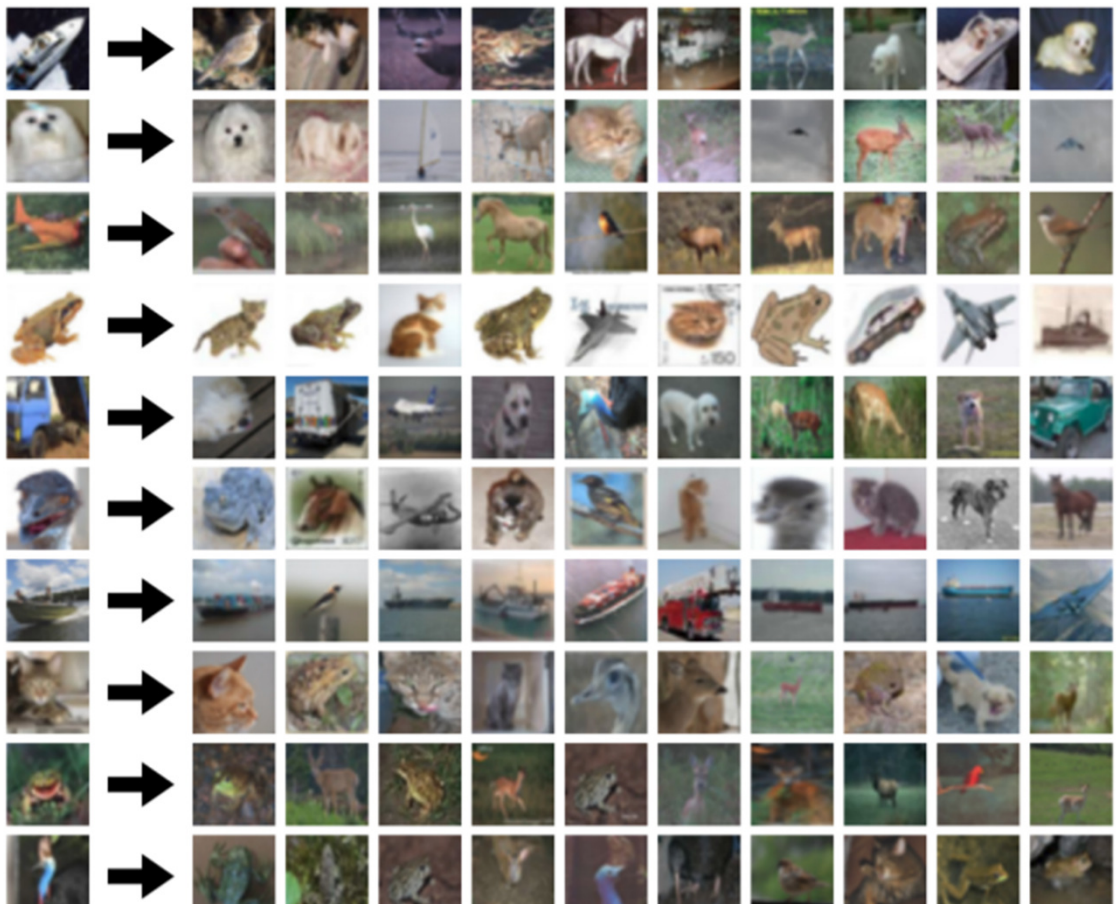
Predict the label of the most similar training image

Distance Metric to compare images

L1 distance: $$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

test image					training image					pixel-wise absolute value differences				
56	32	10	18	-	10	20	24	17	=	46	12	14	1	add → 456
90	23	128	133		8	10	89	100		82	13	39	33	
24	26	178	200		12	16	178	170		12	10	0	30	
2	0	255	220		4	32	233	112		2	32	22	108	

What does this look like?



How to find the nearest neighbour?

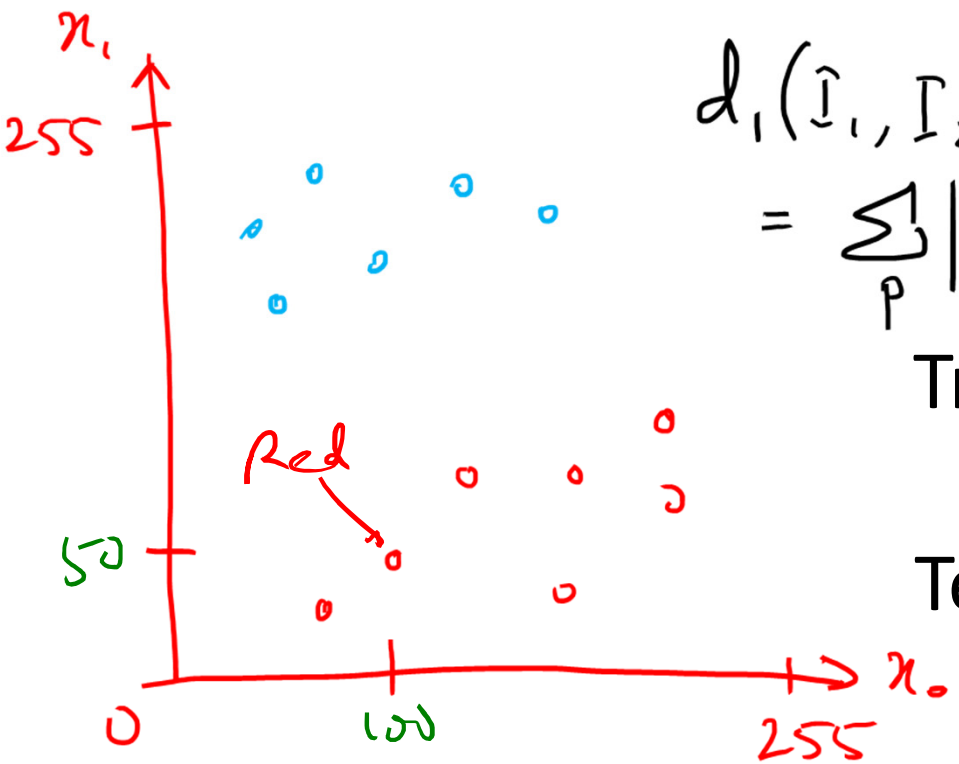
- Distance

Consider a 2-pixel image

x_0 100

x_1 50

pixel values : 0 - 255



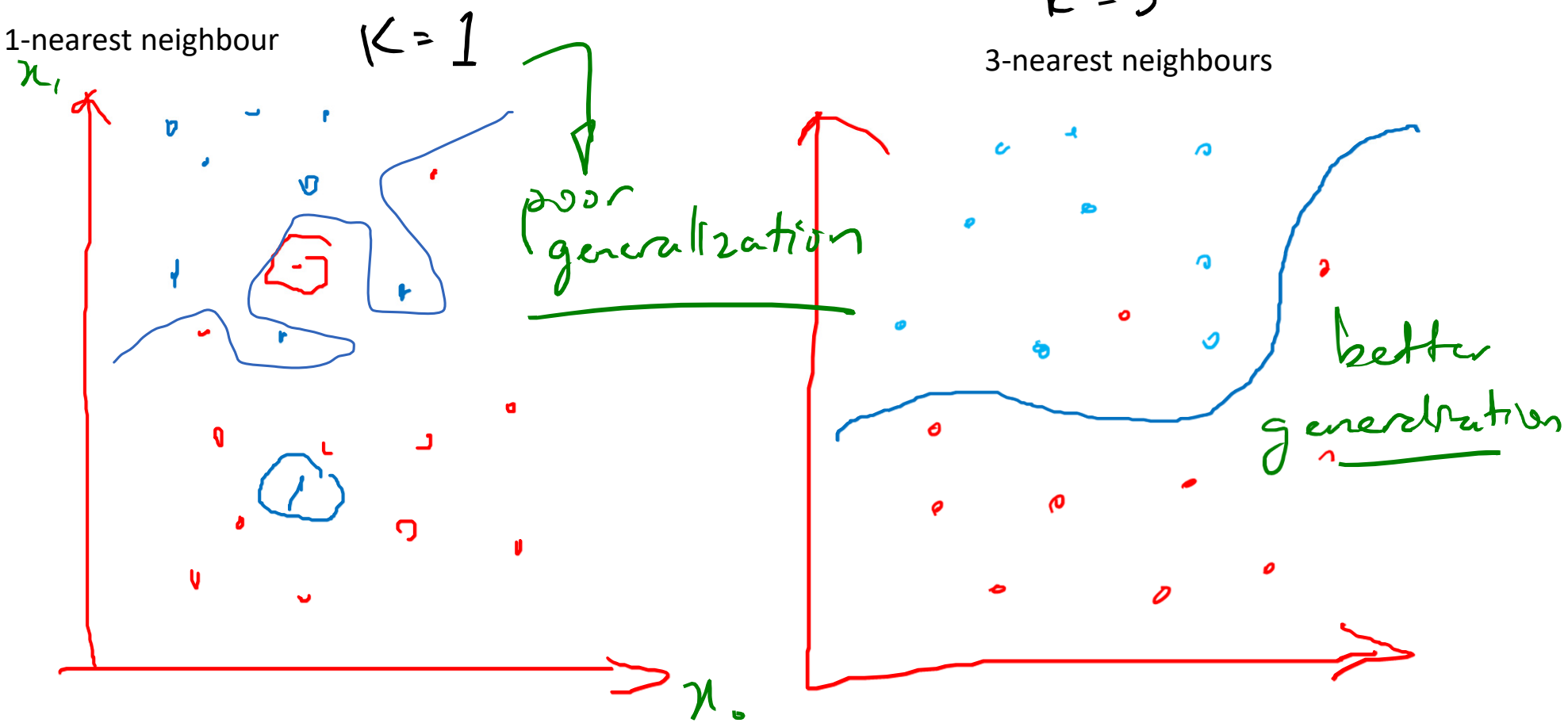
$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

Training Time = N

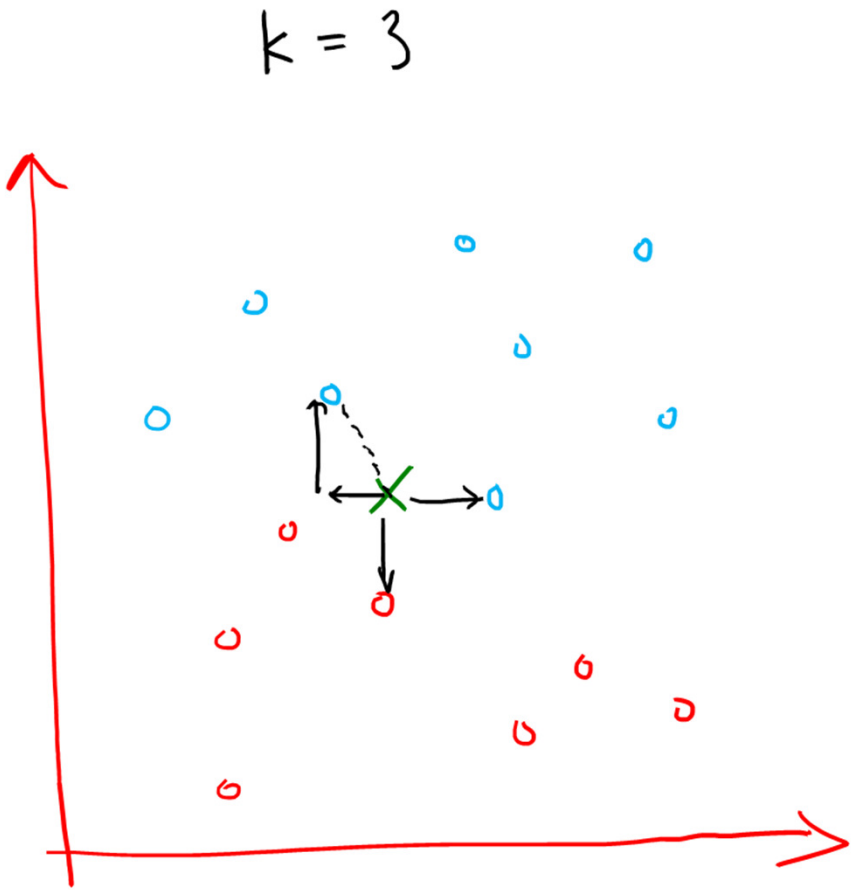
vs

Testing Time = N^2

Nearest neighbour Decision Boundaries



K-Nearest Neighbour



K-Nearest Neighbour – Distance Calculation

- Manhattan
- Euclidean

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)}$$

Hyperparameters

- What is the best value of **K** to use?
- What is the best **distance metric** to use?

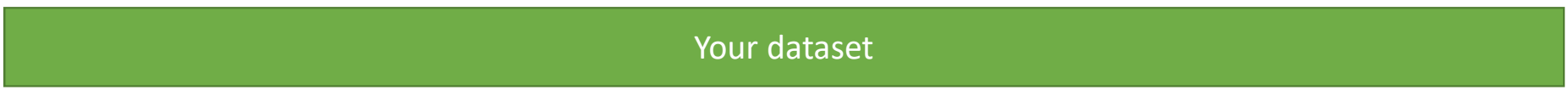
These are examples of **hyperparameters**: choices about our learning algorithm that we don't learn from the training data; instead we set them at the start of the learning process

Very problem-dependent. In general need to try them all and see what works best for our data / task.

Setting Hyperparameters

Idea #1: Choose hyperparameters that work best on the data

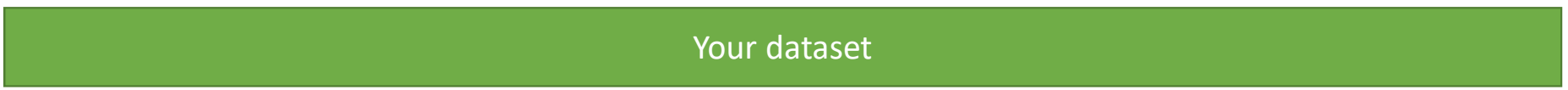
BAD: K = 1 always works perfectly on training data



Setting Hyperparameters

Idea #1: Choose hyperparameters that work best on the data

BAD: K = 1 always works perfectly on training data



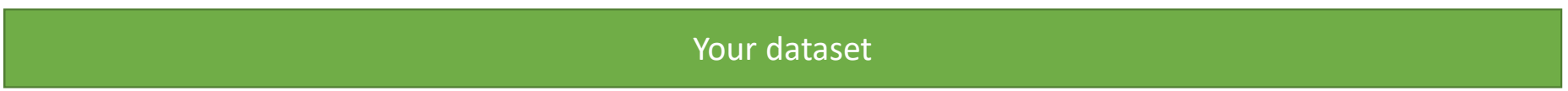
Idea #2: Split data into **train** and **test**, choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data

Setting Hyperparameters

Idea #1: Choose hyperparameters that work best on the data

BAD: K = 1 always works perfectly on training data



Idea #2: Split data into **train** and **test**, choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data

Idea #3: Split data into **train**, **val**, and **test**; choose hyperparameters on val and evaluate on test

Better!

K-Nearest Neighbour

- Seldom used at test time
- Distance metrics on pixels are not informative

- However, K-Nearest Neighbour combined with ConvNet (feature extractor) works very well

MobileNet

- Use ml5.js on p5.js web editor
- Loaded MobileNet model
- Classified the input image:

  cock

What if we want to define our own classes?

Projects related to Teachable Machine

- Pacman
- Pong game
- Objectifier Spatial Programming
- Project Euphonia
- Teachable Snake
- Teachable Sorter
- Teachable Arcade
- Eyeo Festival 2019 - Coding train, code

How does teachable machine work?

Image from Webcam → MobileNet(Second to last / last layer) → KNN Classifier → Output

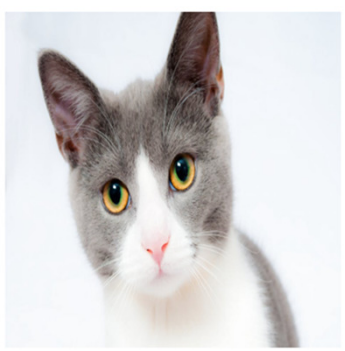
High-level semantic features of the image

MobileNet Model

- **MobileNet**: **Efficient** Convolutional Neural Networks for Mobile Vision Applications(paper).

Model Loaded

The MobileNet model labeled this as robin, American robin, Turdus migratorius, with a confidence of 0.99.



class name	probability	imagenet class id
Egyptian cat	0.478	285
tabby, tabby cat	0.300	281
tiger cat	0.167	282
remote control, remote	0.016	761
Siamese cat, Siamese	0.008	284

MobileNet Model

- **Small**, **fast, accurate**. A model that is trained on **ImageNet**.

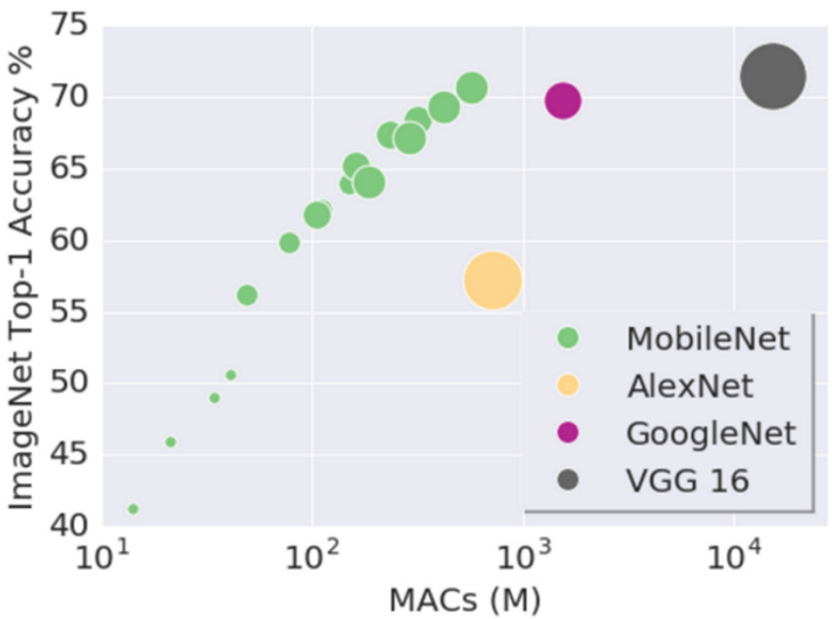


Table 9. Smaller MobileNet Comparison to Popular Models

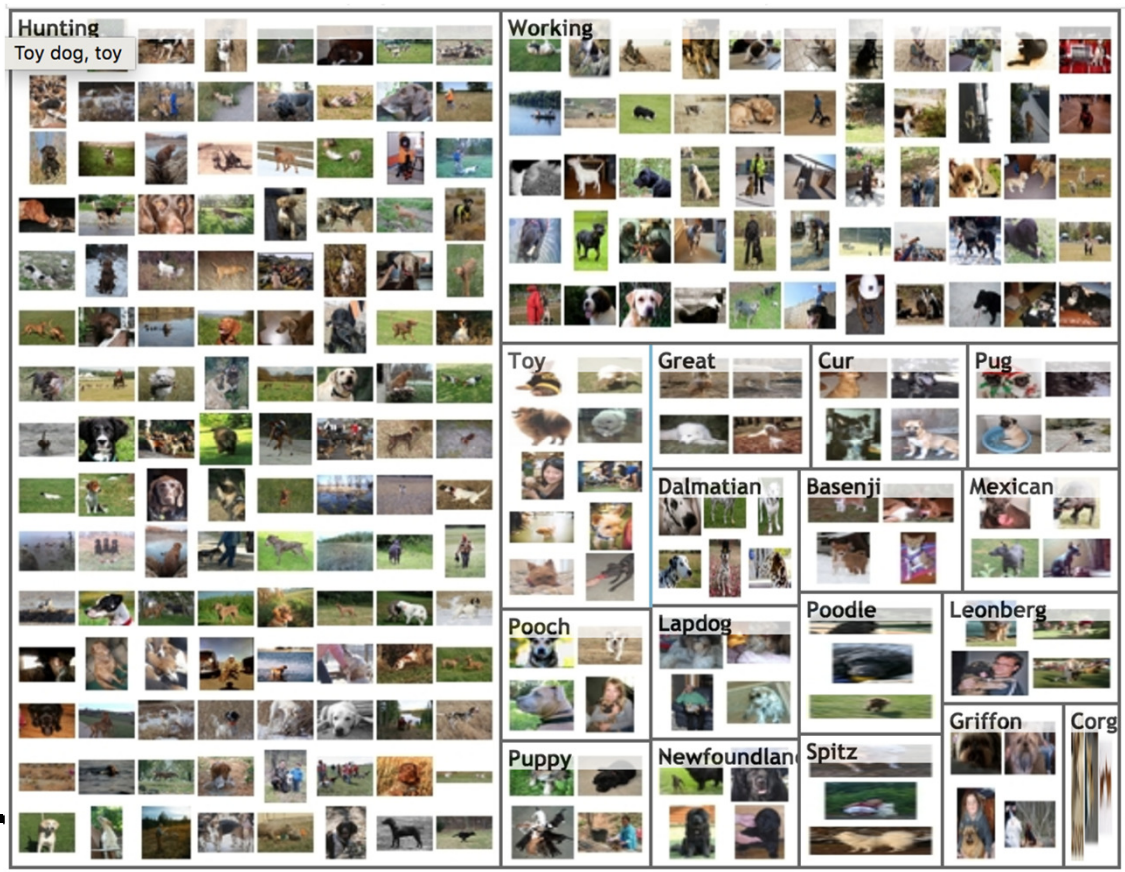
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
0.50 MobileNet-160	60.2%	76	1.32
Squeezenet	57.5%	1700	1.25
AlexNet	57.2%	720	60

* Multiply-Accumulates (MACs) measures the number of fused Multiplication and Addition operations

ImageNet Dataset

ImageNet: a **dataset** of **millions** of images with labels for **1000** different classes of **objects**, like dogs, cats, and fruits.

- ImageNet 2011 Fall Release (32326)
  - plant, flora, plant life (4486)
  - geological formation, formation (175)
  - natural object (1112)
  - sport, athletics (176)
  - artifact, artefact (10504)
  - fungus (308)
  - person, individual, someone, somebody, mortal, soul (6978)
  - animal, animate being, beast, brute, creature, fauna (3998)
  - Misc (20400)



ImageNet Dataset

List of 1000 different classes of objects

0: 'tench, Tinca tinca',
1: 'goldfish, Carassius auratus',
2: 'great white shark, white shark, man-eater, man-eating shark, ' + 'Carcharodon carcharias',
3: 'tiger shark, Galeocerdo cuvieri',
4: 'hammerhead, hammerhead shark',
5: 'electric ray, crampfish, numbfish, torpedo',
6: 'stingray',
7: 'cock',
8: 'hen',
9: 'ostrich, Struthio camelus',
10: 'brambling, Fringilla montifringilla',

Transfer Learning

Use the **knowledge** gained while solving **one problem** and **applying** it to a **different** but related problem.

Start with a **pre-trained model** that are good at one task, lets you train far more **quickly** and with **less data** than if you were to train from scratch.

Transfer Learning

Use the knowledge gained while recognizing the **Imagenet classes** could apply when trying to recognize **user customized classes**.

Benefits:

- **Short** training time
- Needs very **little data**
- All in the **browser**.

Image from Webcam → MobileNet(Second to last layer) → KNN Classifier → Output

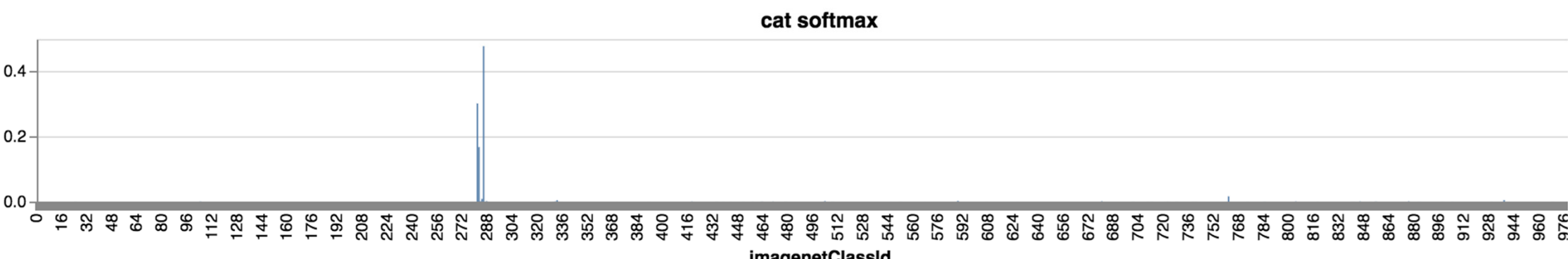
(High-level semantic features of the image)

Output from MobileNet

Output from last layer of MobileNet:

- X axis: 0 - 999, 1000 ImageNet class name id
- Y axis: the probability (0 - 1) of the class

class name	probability	imagenet class id
Egyptian cat	0.478	285
tabby, tabby cat	0.300	281
tiger cat	0.167	282
remote control, remote	0.016	761
Siamese cat, Siamese	0.008	284



Source: [How to build a Teachable Machine with TensorFlow.js](#) by @nsthorat

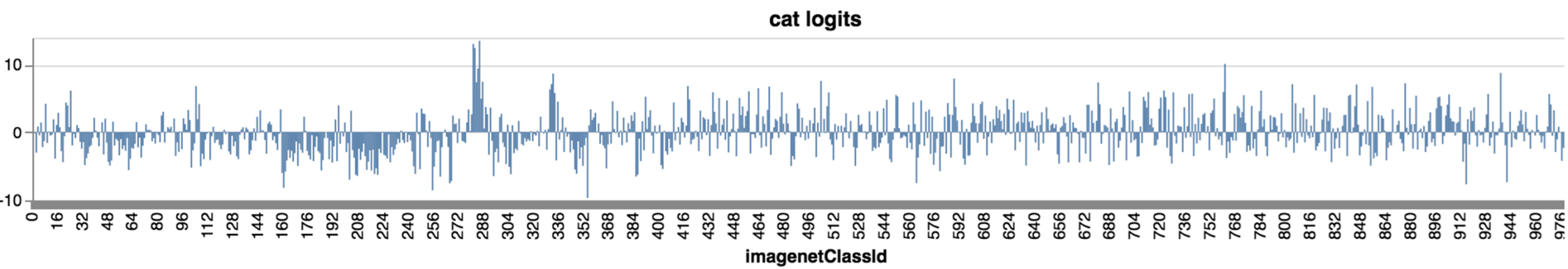
Logits Activation from MobileNet

Activations: **output** of each layer

Softmax: calculates a **probability** for every possible class

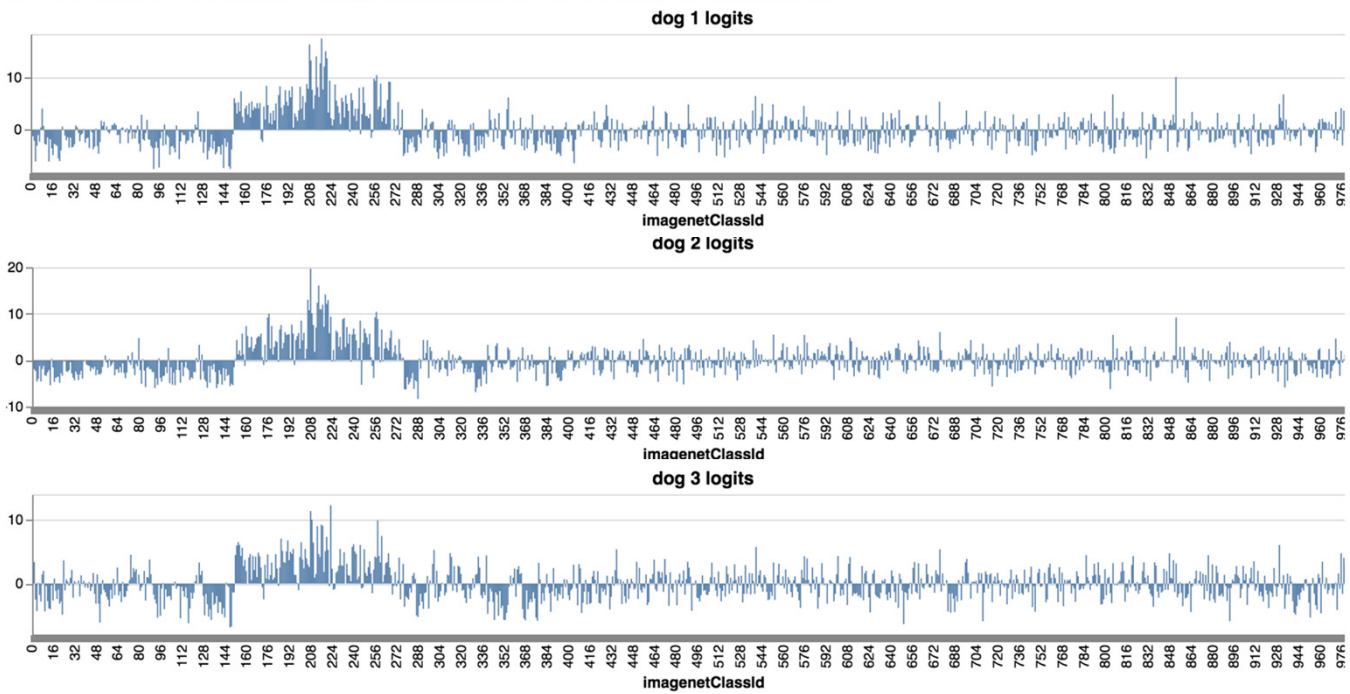
Logits: **unnormalized** predictions vector

class name	probability	imagenet class id
Egyptian cat	0.478	285
tabby, tabby cat	0.300	281
tiger cat	0.167	282
remote control, remote	0.016	761
Siamese cat, Siamese	0.008	284



Source: How to build a Teachable Machine with TensorFlow.js by @nsthorat

More Logits Activation examples from MobileNet

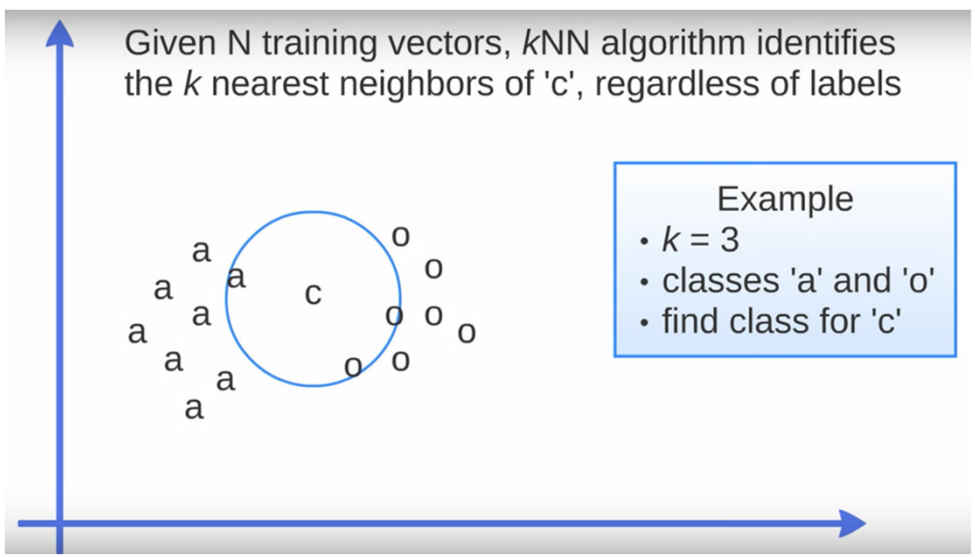


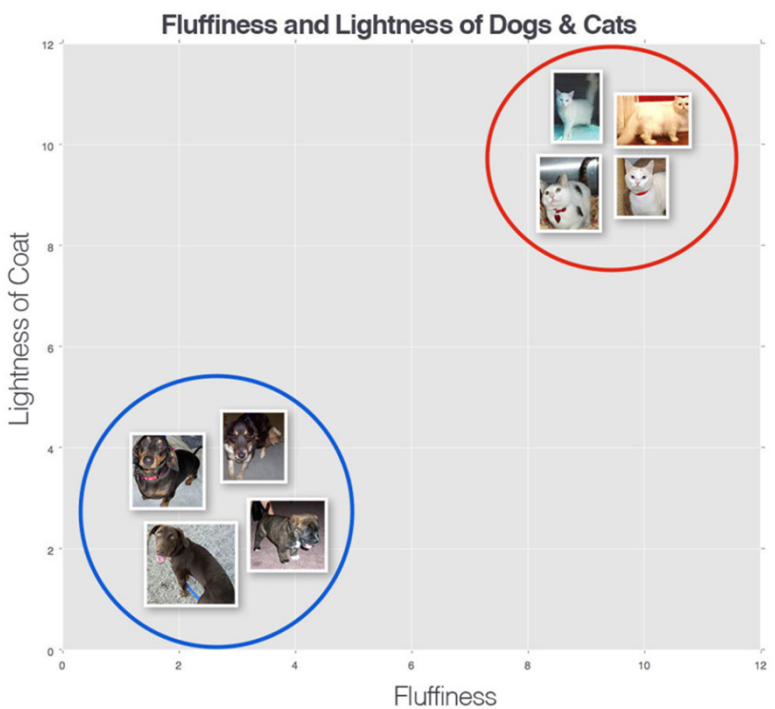
Source: [How to build a Teachable Machine with TensorFlow.js](#) by @nsthorat

What is KNN Classifier?

K Nearest Neighbors, A classify algorithm, play with this interactive demo here.

"Tell me who your neighbors are, and I'll tell you who you are"





Source: How kNN algorithm works, K-NN classifier for image classification

Build a KNN Image classifier with ml5

1. Data collection:

   mobileNet.infer() → knnClassifier.addExample()

2. Prediction

   mobileNet.infer() → knnClassifier.classify()

Image from Webcam → MobileNet(Second to last layer) → KNN Classifier → Output

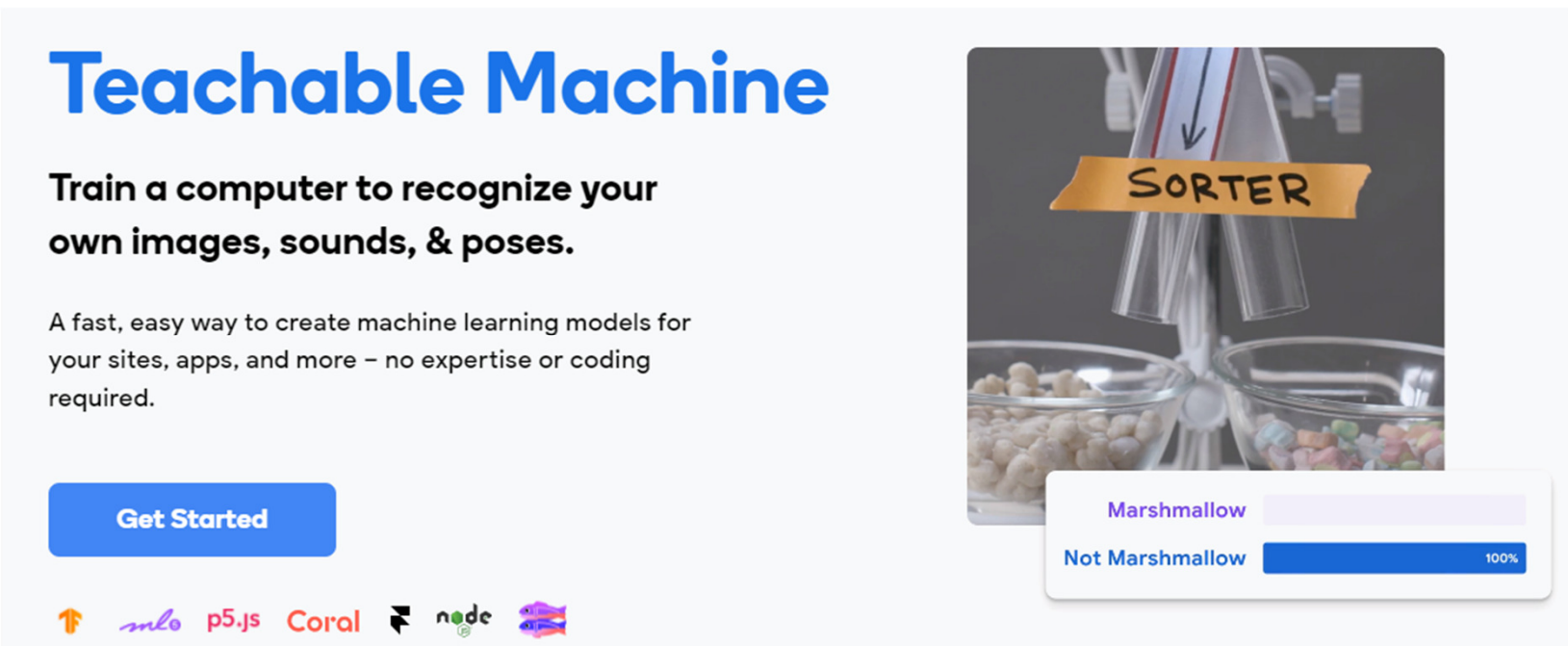
Coding Session

- Teachable machine (image)
- KNN Image Classifier
- Regression with feature extractor

Code: https://editor.p5js.org/ml5/sketches/KNNClassification_Video

Teachable Machine (/CNN Image Classifier)

Customizable image classifier on webcam images



Projects related to Teachable Machine (Image Recognition)

- Teachable Machine 2.0: Making AI easier for everyone
- Objectifier Spatial Programming
- Pacman
- Fruit quality detection
- Garbage detection
- Flower recognizer
- Teachable Snake
- Teachable Sorter
- Teachable Arcade

Main Takeaways

1. What is MobileNet
2. What is KNN Classification?
3. How does Teachable Machine work?
4. What is Transfer Learning?