# Impacts of AI: COMP3800-03
# Computer Vision
# Wentworth Institute of Technology

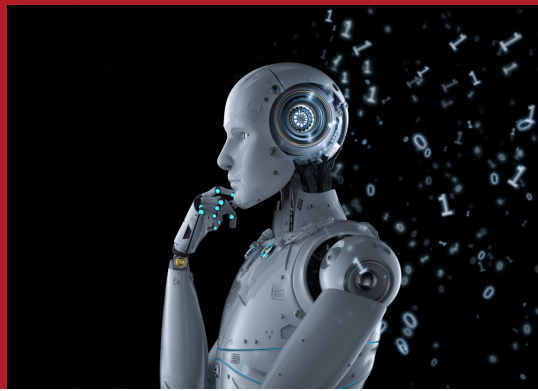
# Computer Vision: Explained

**Training computer vision models is done in much the same way as teaching children about objects visually.**

Instead of a person being shown physical items and having them identified, however, the computer vision algorithms are provided many examples of images that have been tagged with their contents.

In addition to these positive examples, negative examples are also added to the training.

*For example, if we're training for images of cars, we may also include negative examples of airplanes, trucks, and even boats.*



# Image Classification and Tagging

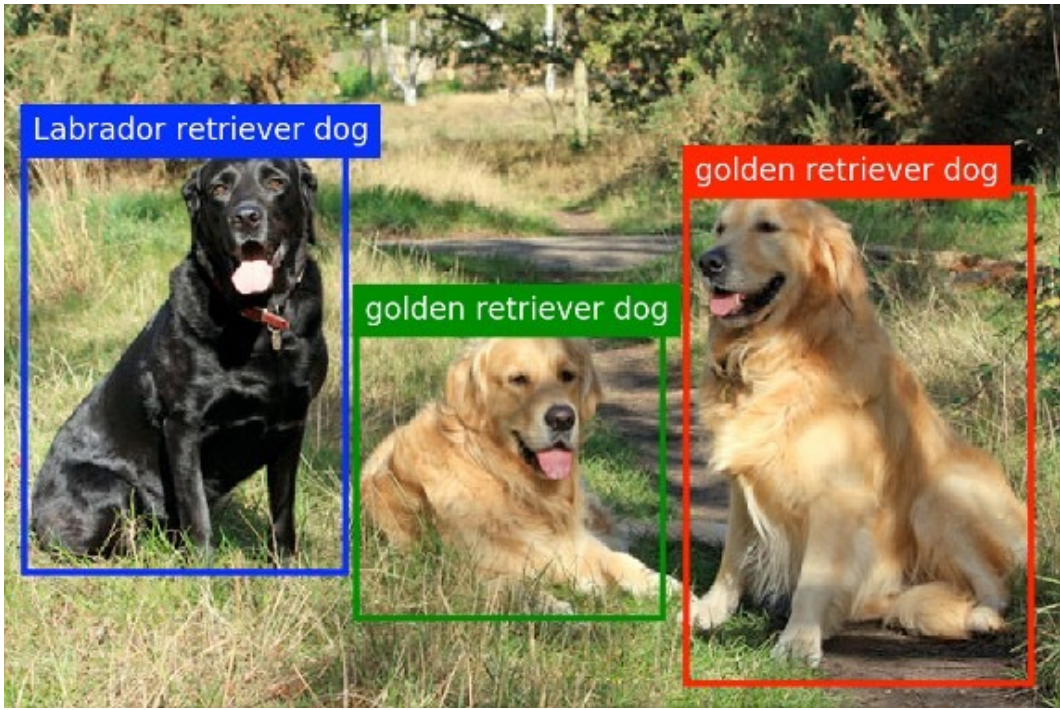
Computer vision's most core functionality, *general image tagging and classification*, allows users to understand the content of an image.


*For example*
You may need to find images with contents of "male playing soccer outside" or organize images into visual themes such as cars, sports, or fruits.

# Object Localization

Sometimes your application's requirements will include not just classifying what is in the image

But also understanding the position of the particular object in relation to everything else

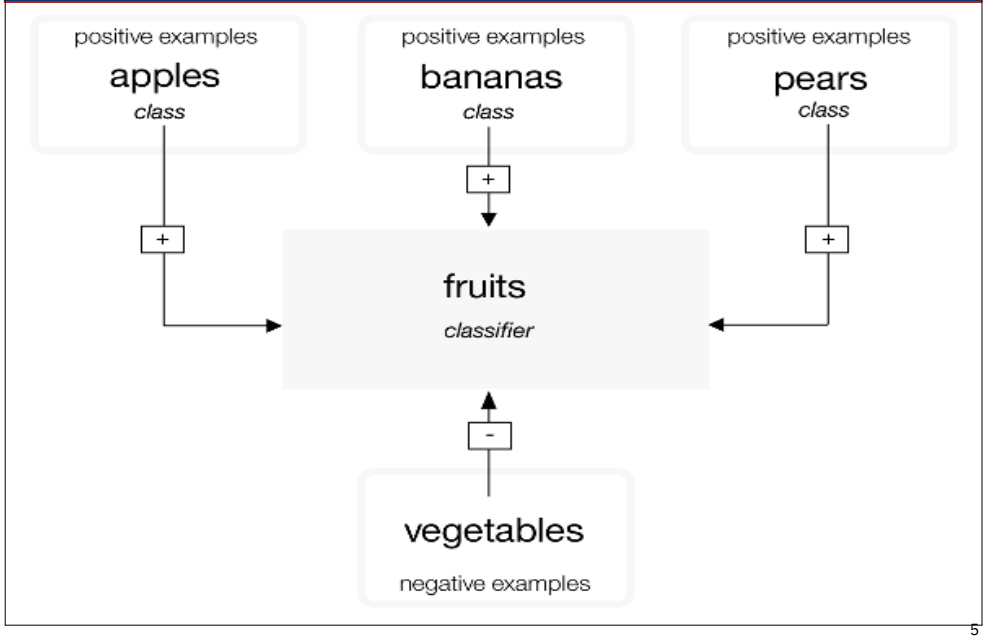


# Custom Classifiers

If you're looking to identify or classify only a small set of objects, custom classifiers could be the right tool.

Most of the large third-party platforms provide some mechanism for building custom visual classifiers, allowing you to train the computer vision algorithms to recognize specific content within your images

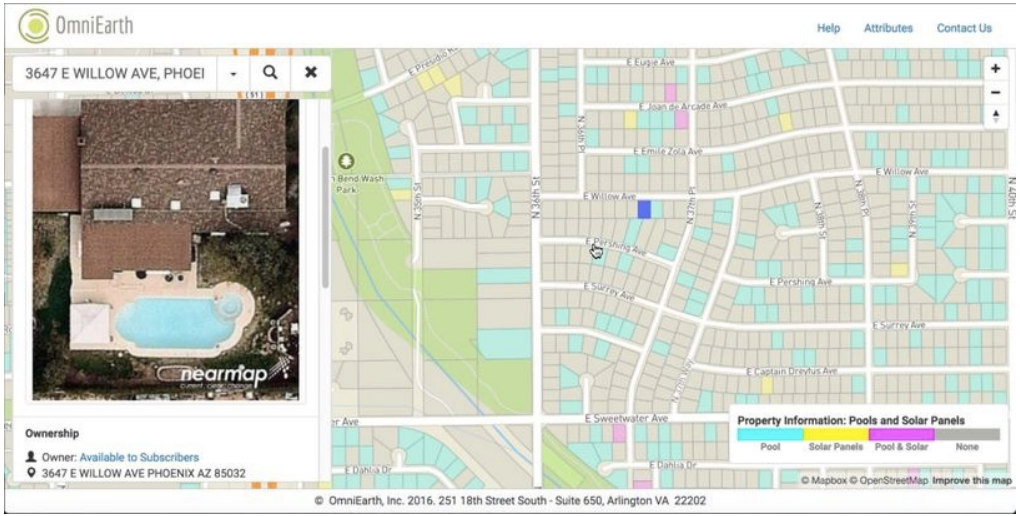
*For example,*
If you were training a custom classifier on fruits, you'd want to have positive training images of apples, bananas, and pears. For negative examples, you could have pictures of vegetables, cheese, or meat.



# Use Cases: Satellite Imaging

California has been in a drought, making water an extraordinarily precious resource—one that Californian residents and governments are eager to protect.

OmniEarth's program uses satellite and aerial imagery to keep track of water use. Given a picture, so the machine can 'see' what's in the picture, whether it's an animal, car, toy, etc.



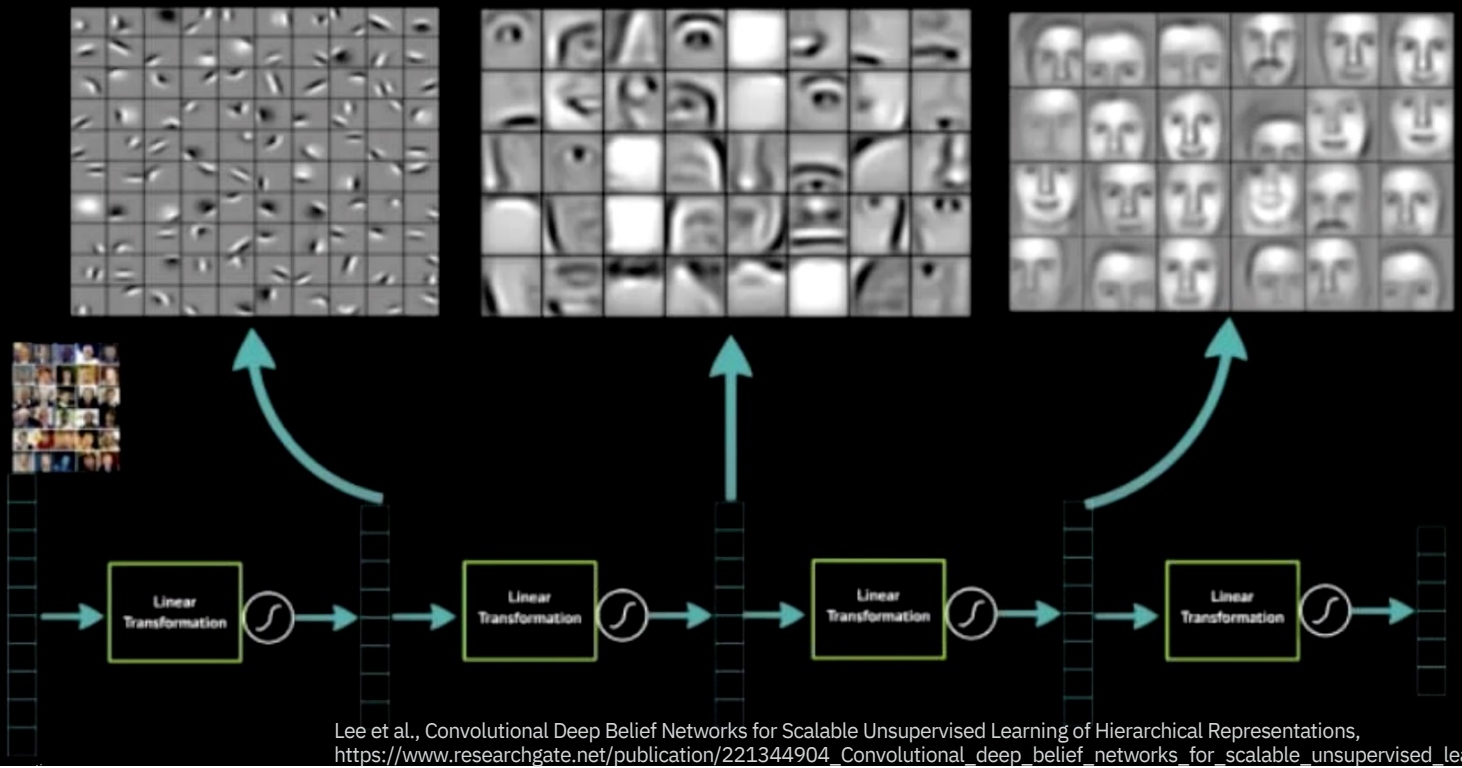
# Use Cases: Video Search in Surveillance and Entertainment

The proliferation of cameras in recent years has led to an explosion in video data

In home surveillance videos, the only viable solution is still human monitoring 24/7, watching screens and raising an alert when something happens.

BlueChasm's product, VideoRecon, can watch and listen to videos, identifying key objects, themes, or events within the footage. It will then tag and timestamp those events, and then return the metadata to the end user

# Deep learning learns layers of features




Lee et al., Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations, https://www.researchgate.net/publication/221344904_Convolutional_deep_belief_networks_for_scalable_unsupervised_learning_of_hierarchical_representations

# Let's Build-up a Neural Network

Node (neuron)

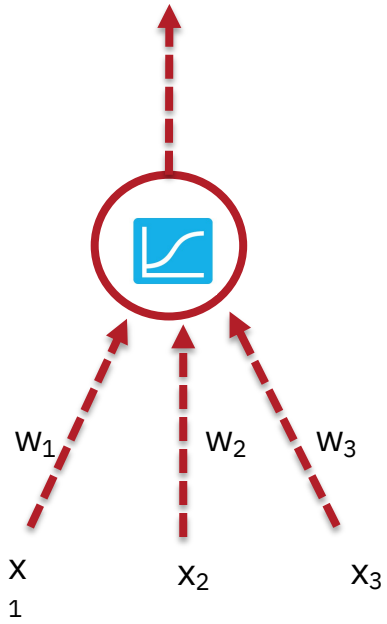
Connection (Synapse)

$x_1, x_2, x_3, \ldots x_n$ Input

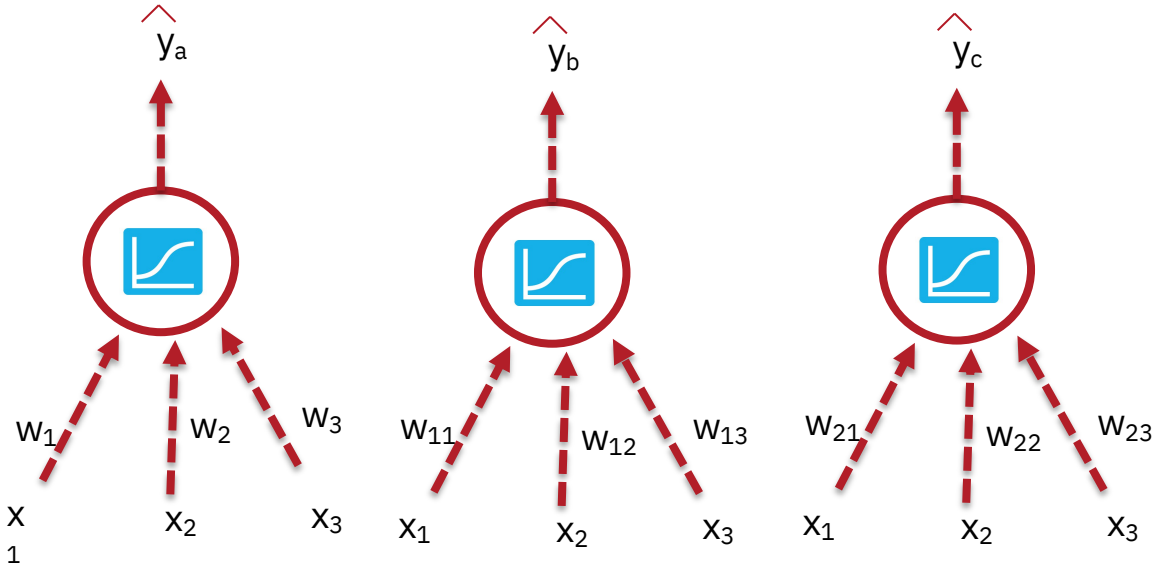
$w_1, w_2, w_3, \ldots w_n$ Weights

$\hat{y}$ Predicted output

$\hat{Y} = \text{function}(x_1, x_2, x_3, w_1, w_2, w_3)$

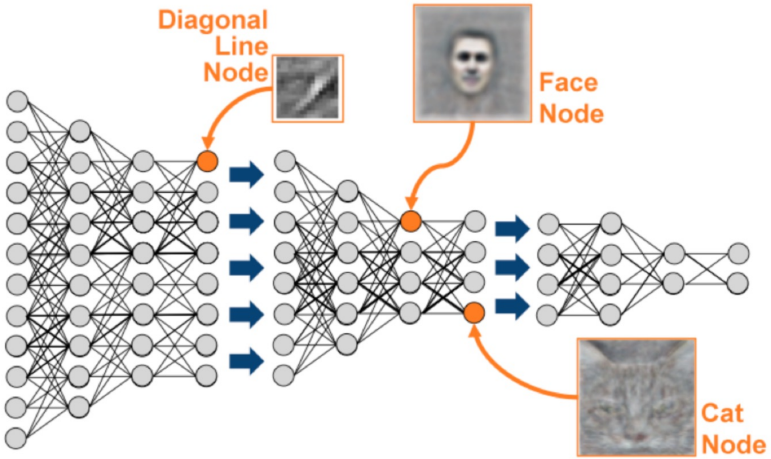


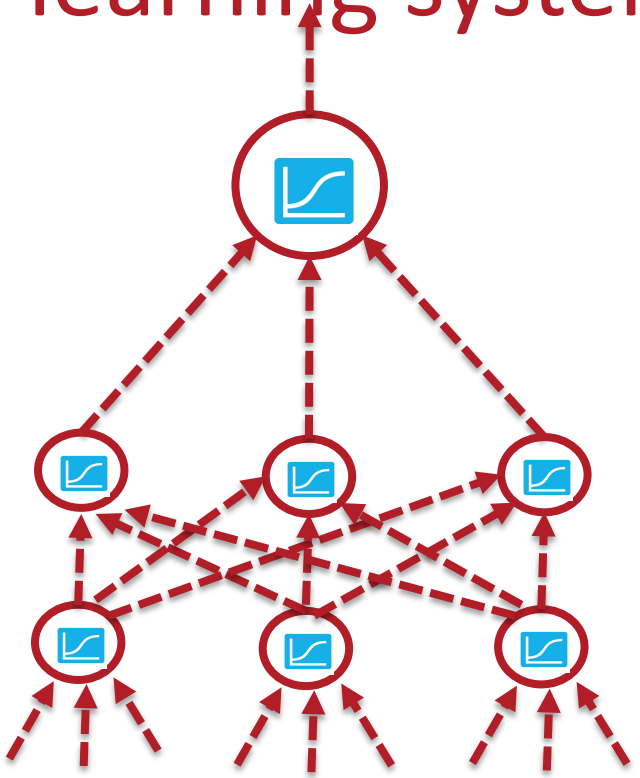
# A Layer is a group of nodes organized at the same level



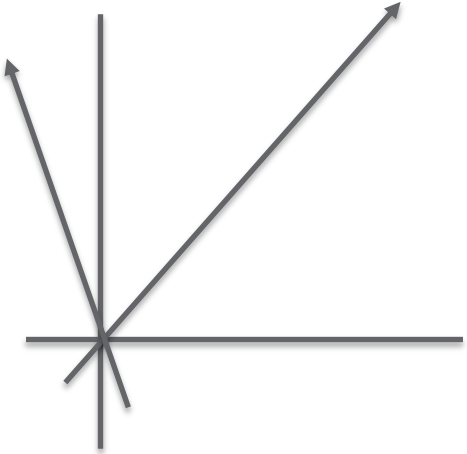
# Multiple layers are known as a multiperceptron or deep learning systems

- A network that uses binary classification the output will be two nodes.
- The bias used in the nodes can be a sigmoid operation, parabolic tangent or a Rectify linear unit (ReLU)

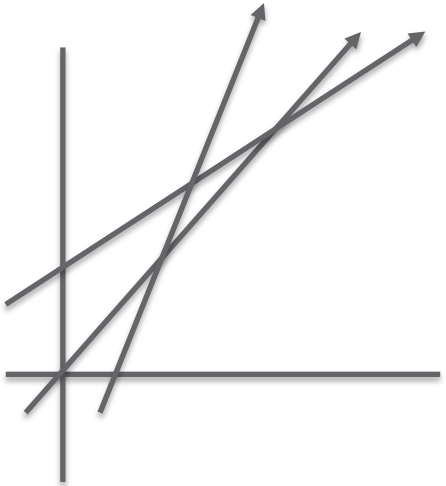


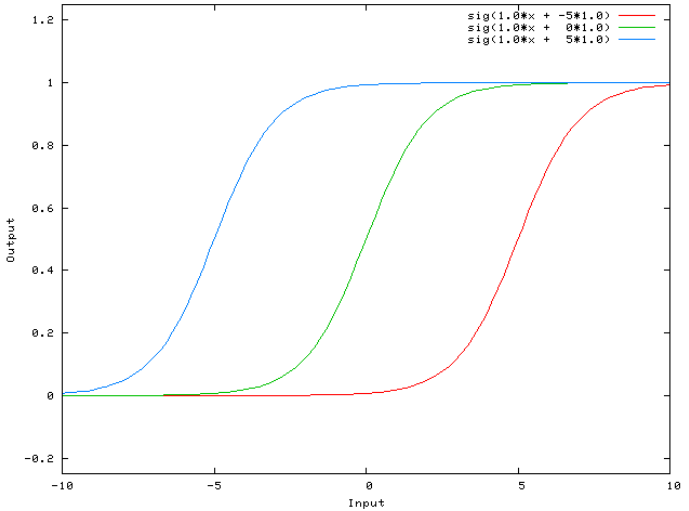
# Why weights and why bias?



y= (f)x



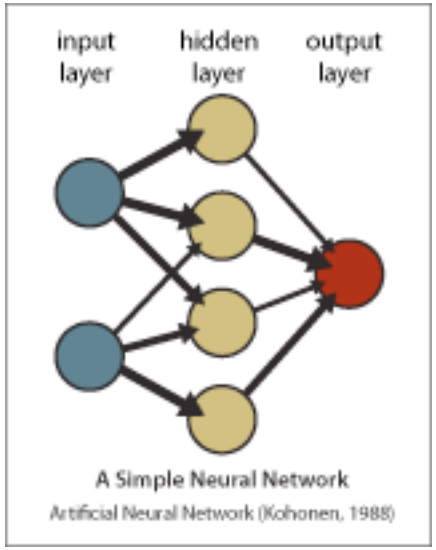
y = (f)x + c

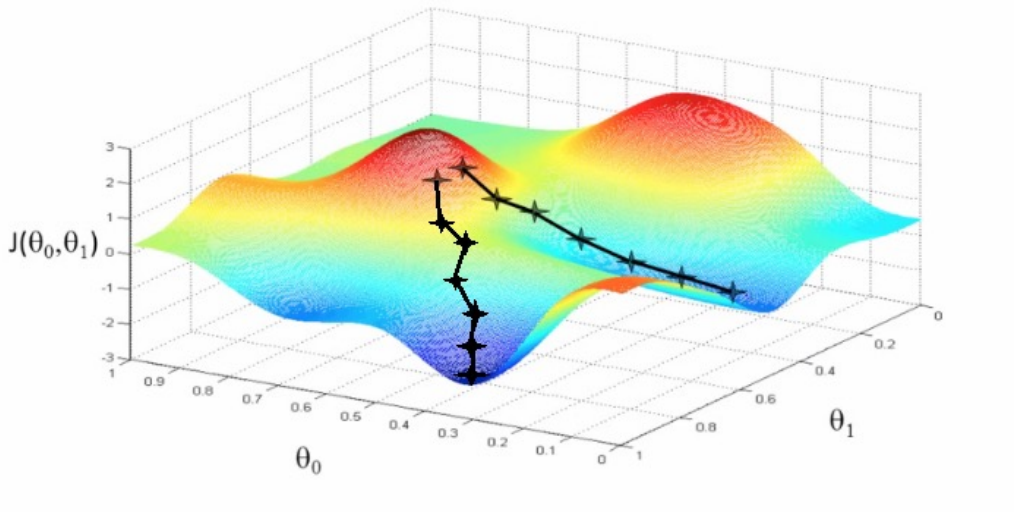



Three sigmoid curves—
the same input data, but with different biases

The bias value (c) allows the activations function to be shifted to the left or to the right, to better fit the data; hence, the changes to the weight alter the steepness of the vector or curve. The bias, on the other hand, shifts the entire vector or curve so it fits better.

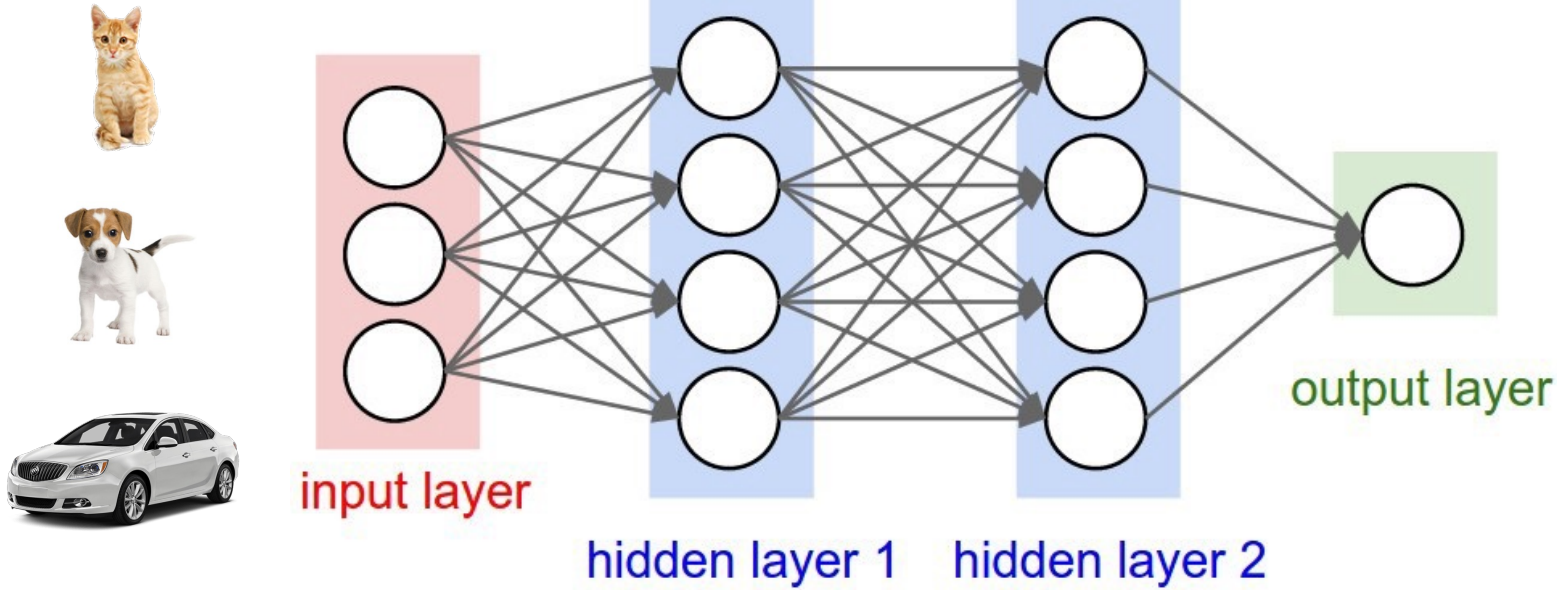
# Descent to the lowest point: Global Minimum



# How do pixels and features turn into vectors? 1-hot encoding

**Labeled input → supervised learning**



# What if your data comprise images, tweets, videos?

[ x,x,x]

[ 1,0,0]

[ x,x,x]

[ 0,1,0]

[ x,x,x]

[ 0,0,1]

**Index or element in a vector**

# 1-hot encoding allows you to encode categorical data into numbers

**1-Hot Encoding:**

- Create a matrix of 0's and 1's
- Make each category a column in a table
- Warning: *you must encode (n-1) categories you have in the variable*
  - Otherwise, you will have *perfect multicollinearity* and encounter mathematical issues

| Sample | Species |
|---|---|
| 1 | Human |
| 2 | Human |
| 3 | Penguin |
| 4 | Octopus |

**Cons:**

- This makes the data very large and sparse
- Better methods for text data
- Does not scale well to big data

**Solutions:**

- Bag of words / TF-IDF for text data
- Advanced algorithms such as neural networks with embedding layers

| Sample | Human | Penguin | Octopus |
|---|---|---|---|
| 1 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 |
| 3 | 0 | 1 | 0 |
| 4 | 0 | 0 | 1 |