place zoom
headshot
window here

CS249r

Intro to (tiny) Machine Learning Part 1

Brian Plancher 9/9/2020

Disclaimer

There are many online resources for learning about Machine Learning -- we will **try to summarize the key points** that you need to understand to dive into TinyML in the next two classes.

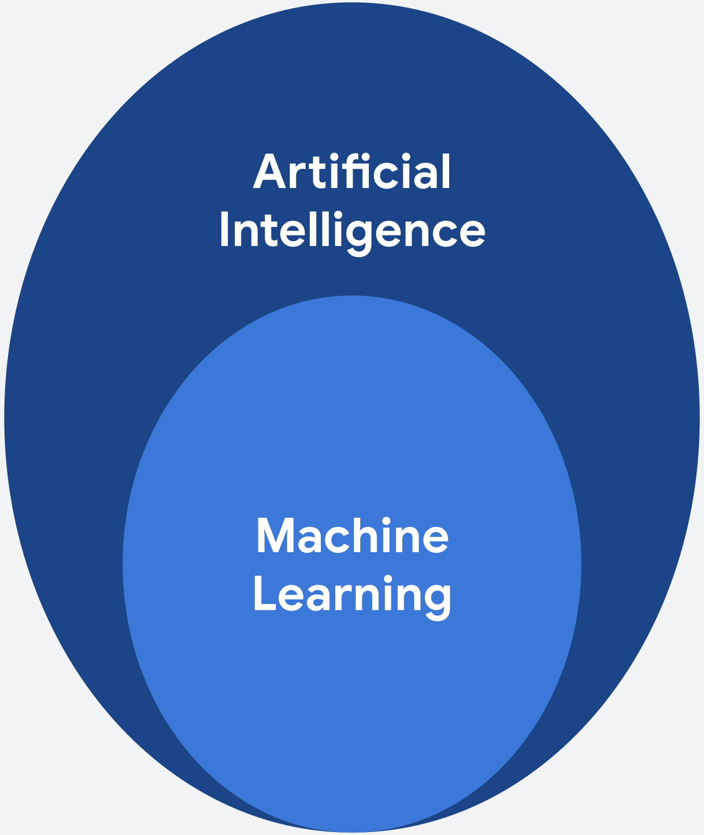
Some helpful resources for further study (and inspiration for lecture):

- The Machine Learning for Humans Blog
- The Machine Learning is Fun Blog
- The Machine Learning Glossary
- Justin Markham's SciKitLearn Course
- Andrew Ng's ML Coursera Course
- Google's ML Crash Course
- CalTech's ML Video Library
- MIT's Intro to Deep Learning
- The History of Deep Learning

What is Machine Learning?

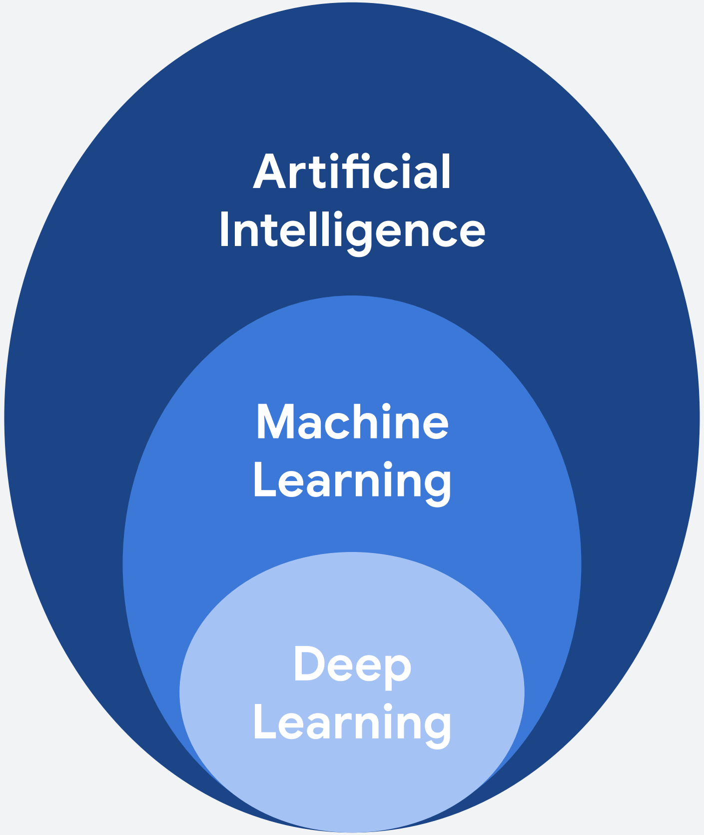
What is (Deep) Machine Learning?

1. **Machine Learning** is a subfield of **Artificial Intelligence** focused on developing algorithms that learn to **solve problems by analyzing data for patterns**



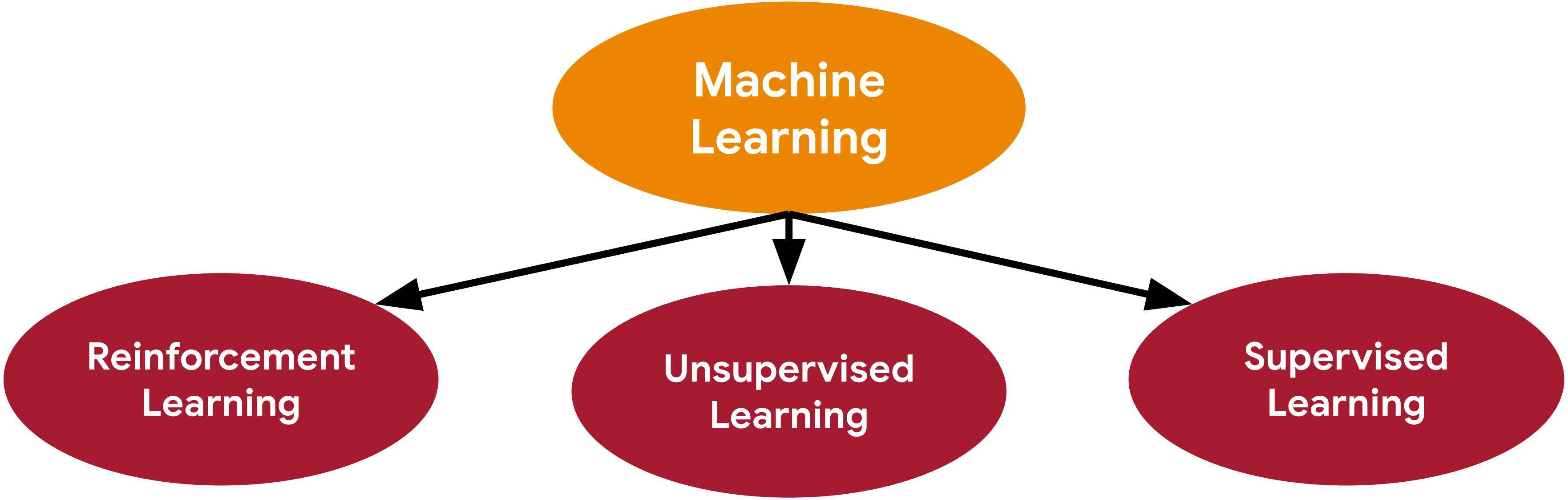
What is (Deep) Machine Learning?

1. Machine Learning is a subfield of Artificial Intelligence focused on developing algorithms that learn to solve problems by analyzing data for patterns
2. **Deep Learning** is a type of Machine Learning that leverages **Neural Networks** and **Big Data**

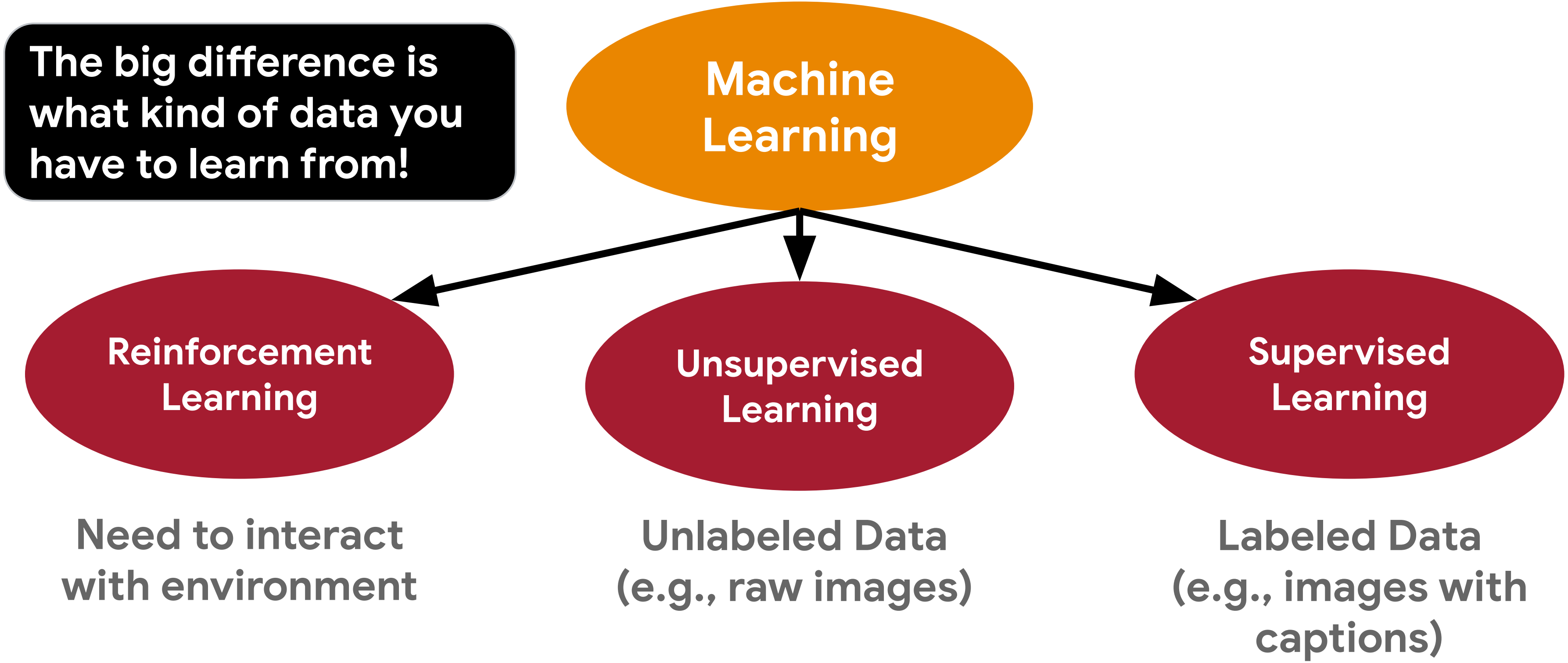


(A) ML Taxonomy

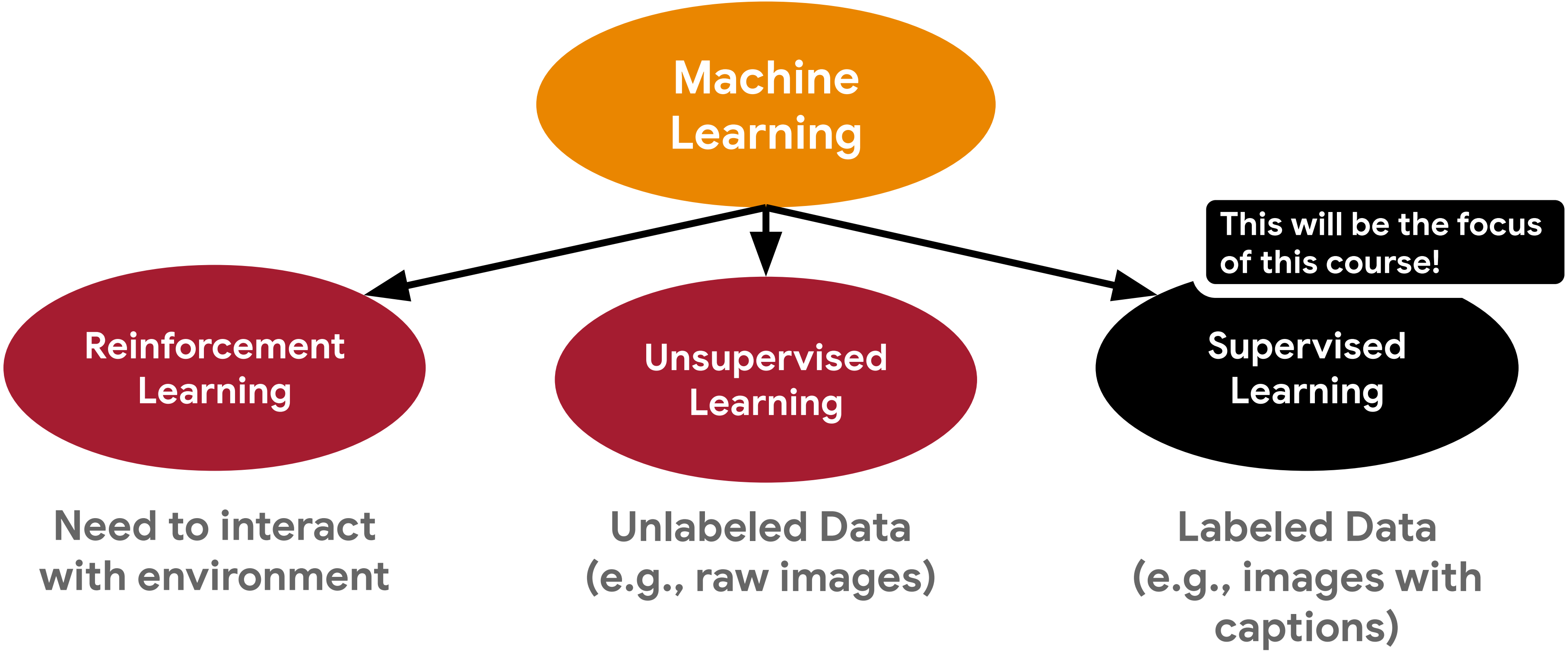
(A) ML Taxonomy



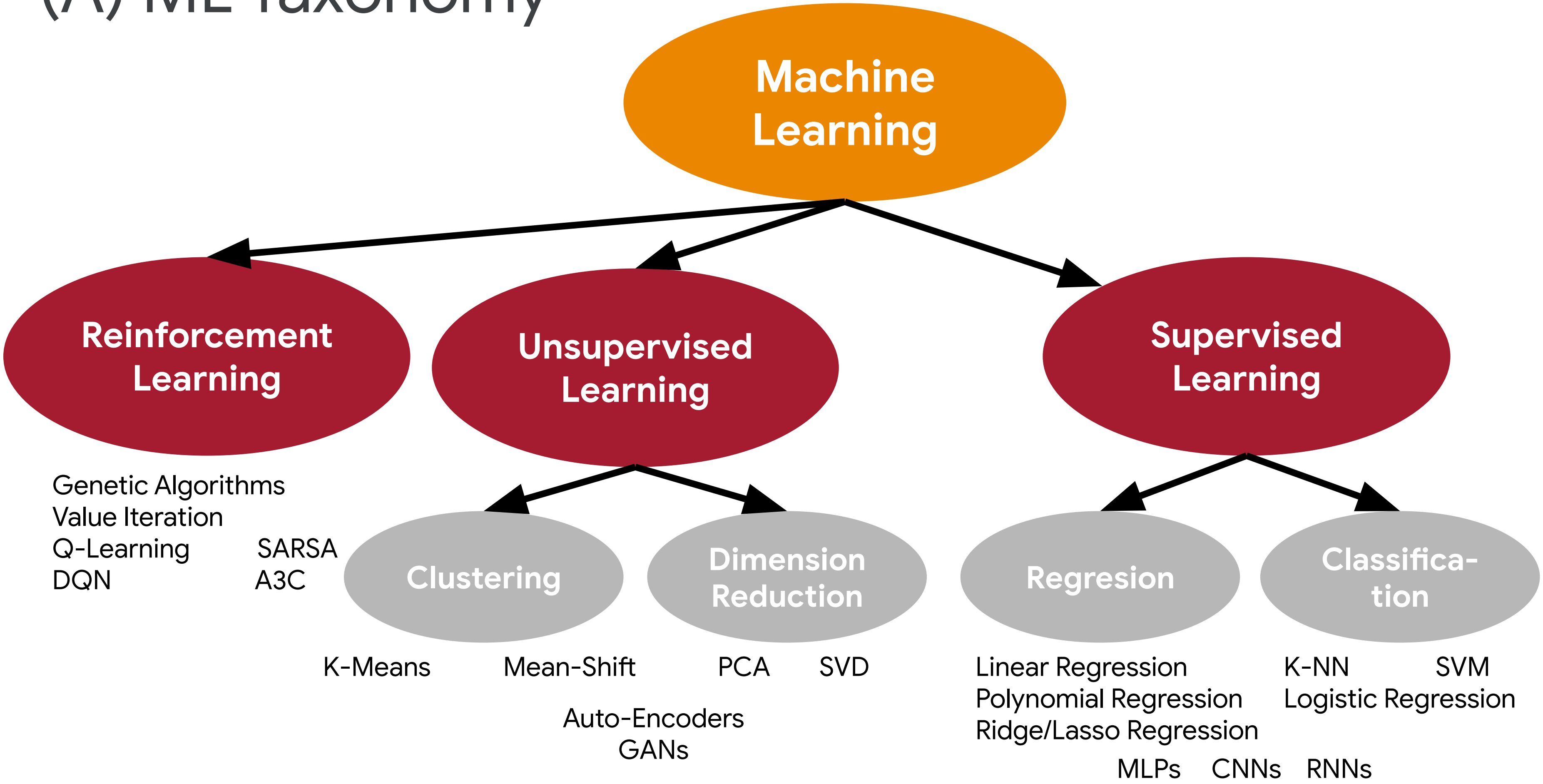
(A) ML Taxonomy



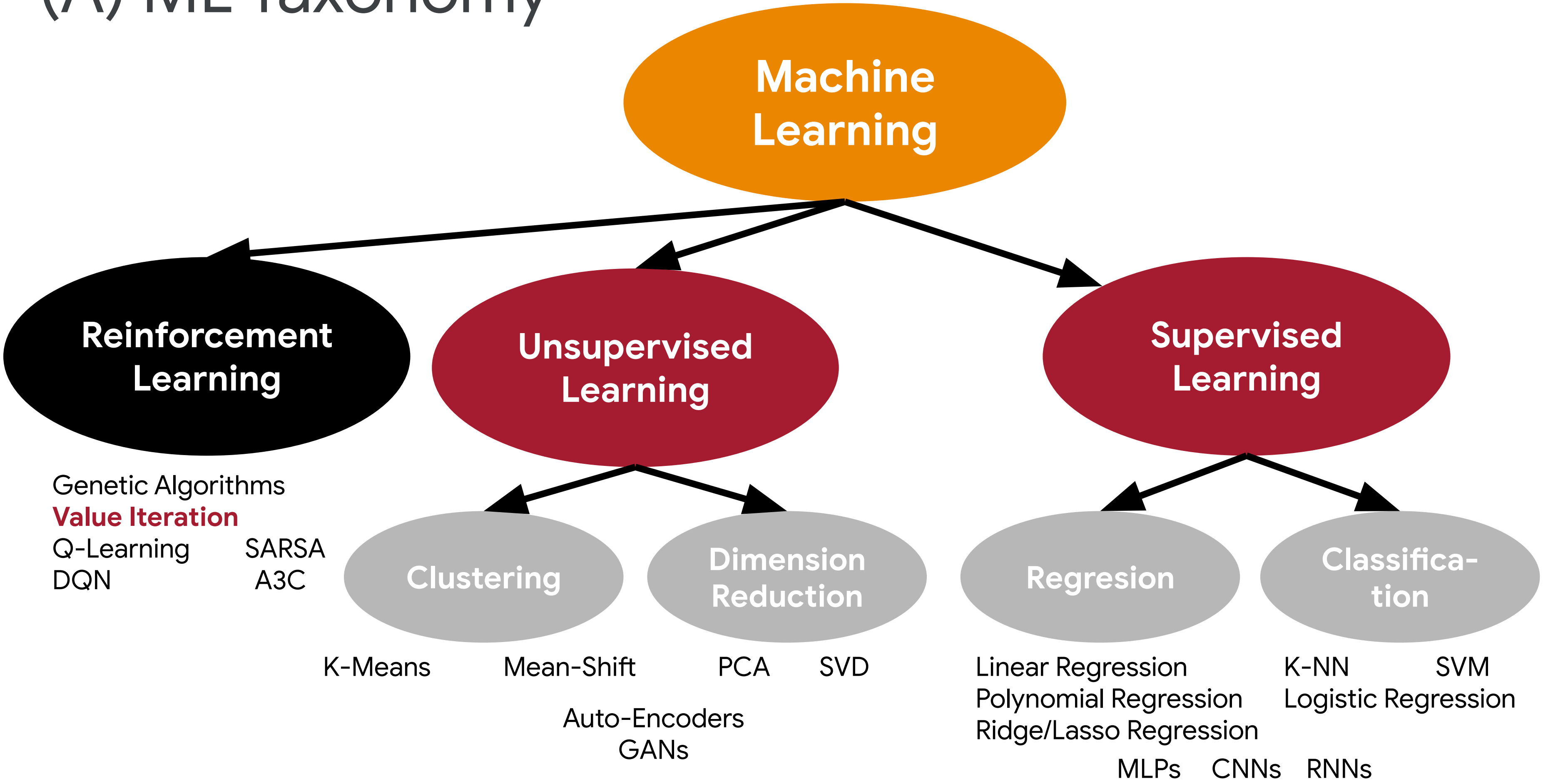
(A) ML Taxonomy



(A) ML Taxonomy

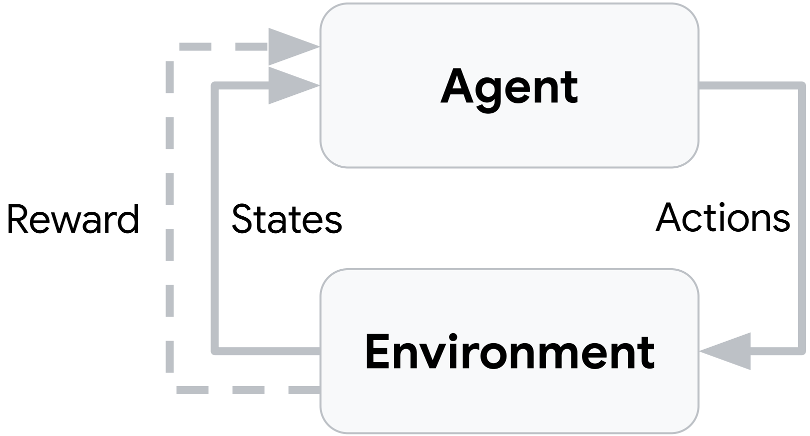


(A) ML Taxonomy



Reinforcement Learning with Value Iteration

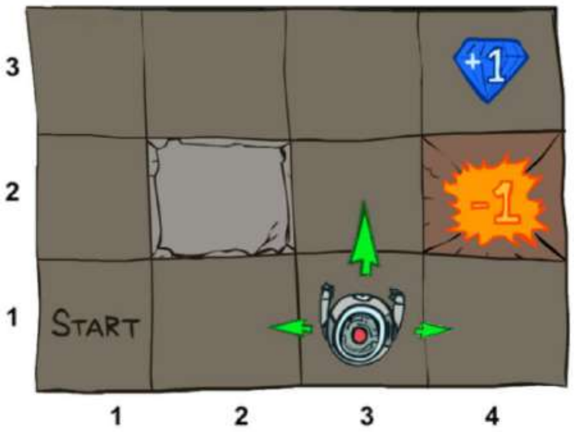
Reinforcement Learning is used when there is no data BUT an **agent** can interact with an **environment** (through actions resulting in new states) and after a series of actions the agent receives a **reward.** This is a common model used in **Robotics**.



https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c

Reinforcement Learning with Value Iteration

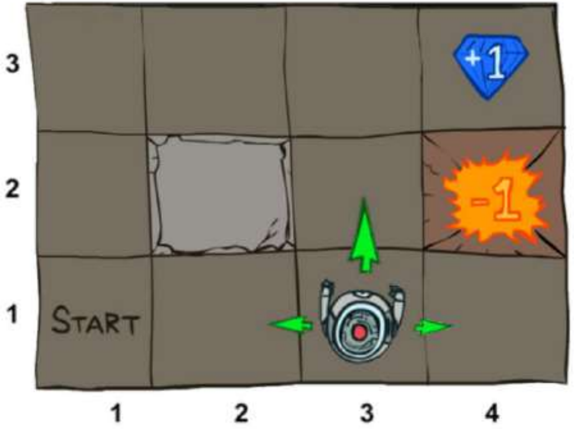
Value Iteration builds a **recursive brute force estimate** of the **value** of being in each **gridded** world state

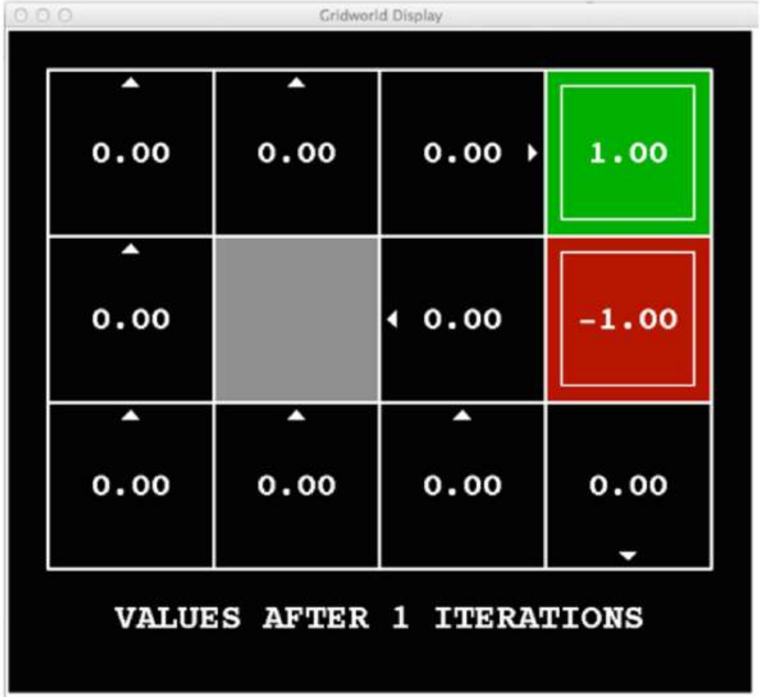


http://ai.berkeley.edu/home.html

Reinforcement Learning with Value Iteration

Value Iteration builds a **recursive brute force estimate** of the **value** of being in each **gridded** world state

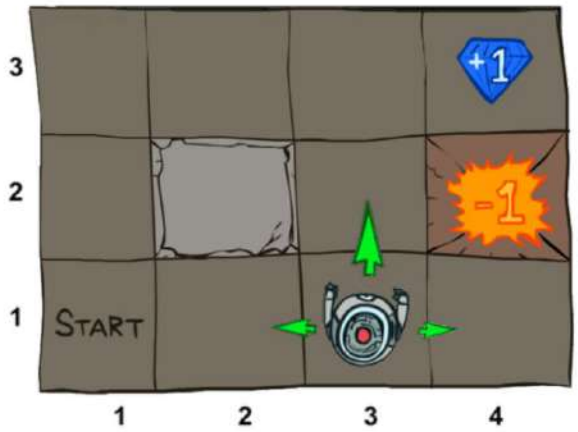


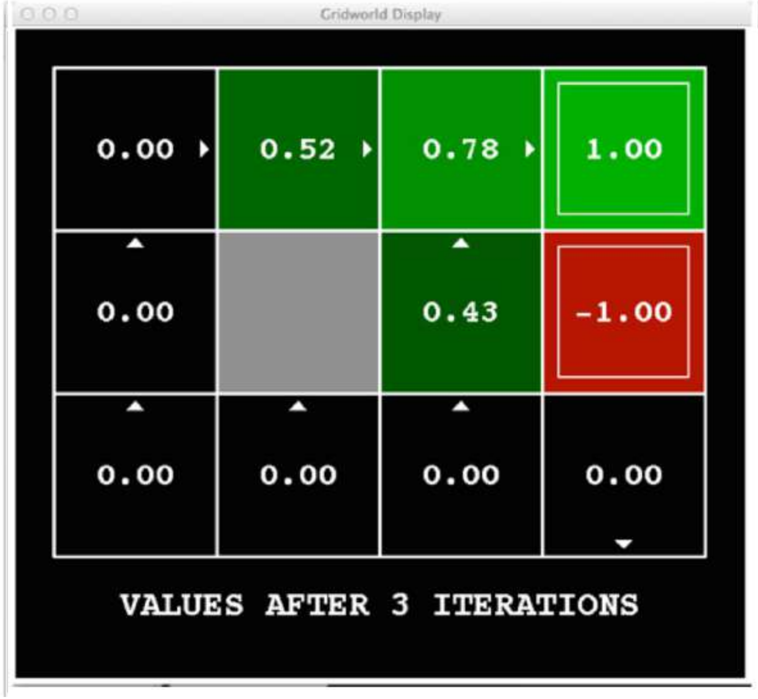


http://ai.berkeley.edu/home.html

Reinforcement Learning with Value Iteration

Value Iteration builds a **recursive brute force estimate** of the **value** of being in each **gridded** world state

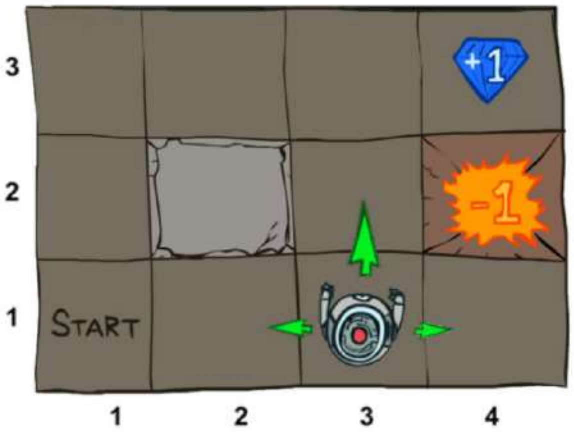


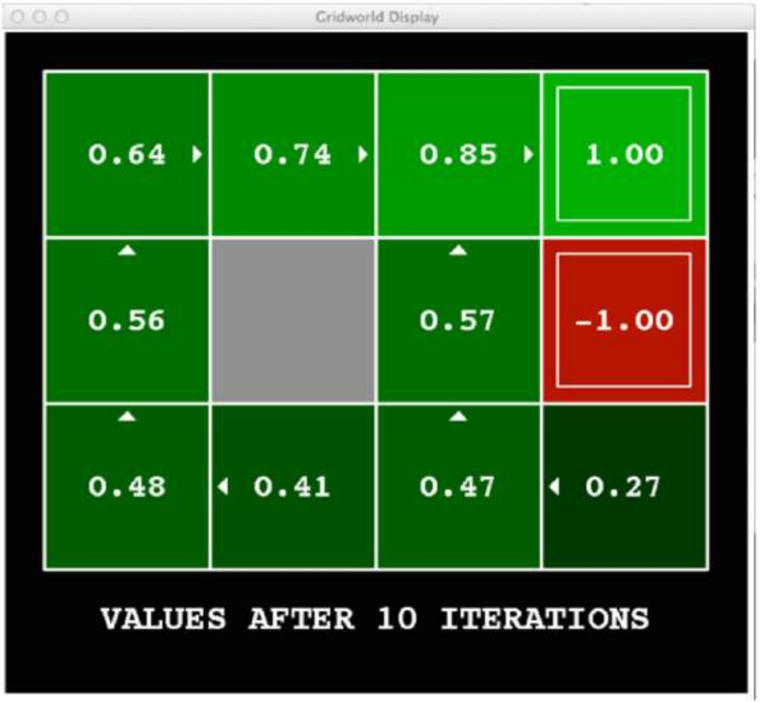


http://ai.berkeley.edu/home.html

Reinforcement Learning with Value Iteration

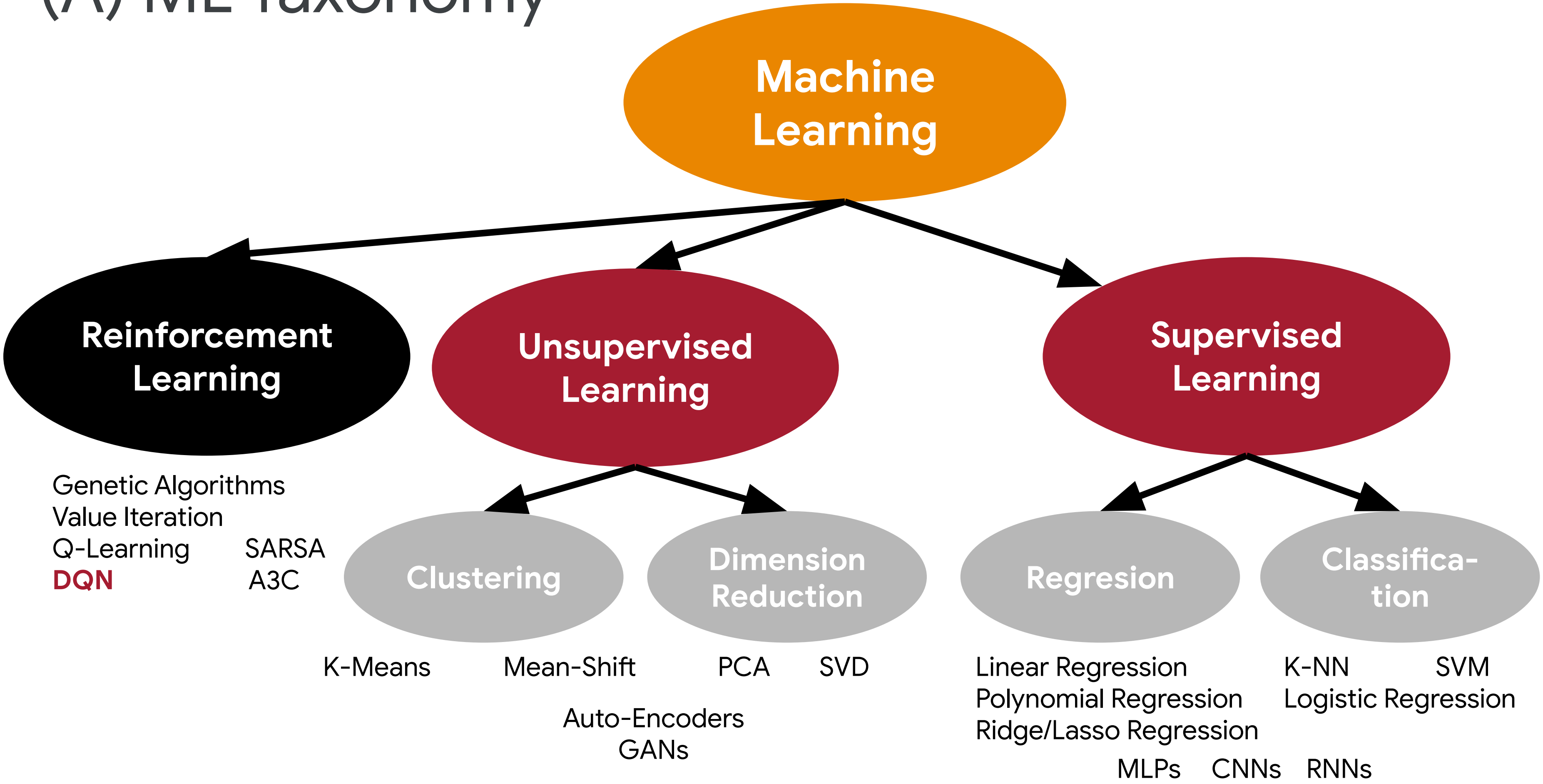
Value Iteration builds a **recursive brute force estimate** of the **value** of being in each **gridded** world state





http://ai.berkeley.edu/home.html

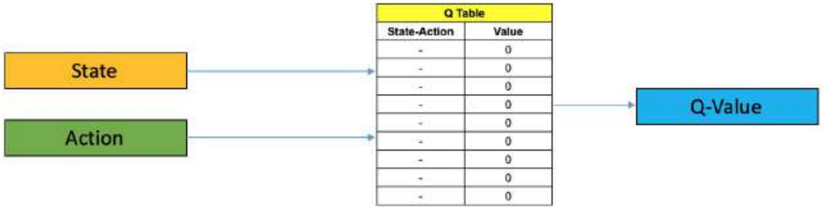
(A) ML Taxonomy



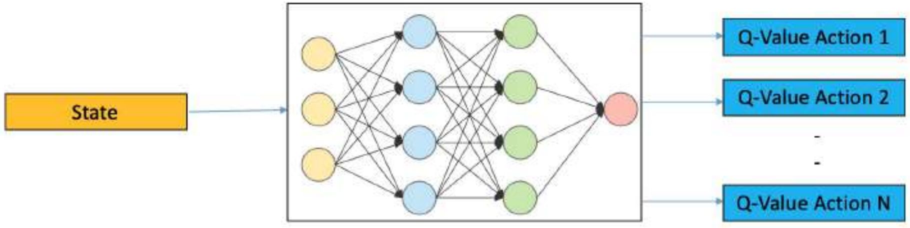
Reinforcement Learning with DQN

We can take this a step further and **generalize** things more by using a **neural network to map images directly to actions**

More on this later!



Q Learning



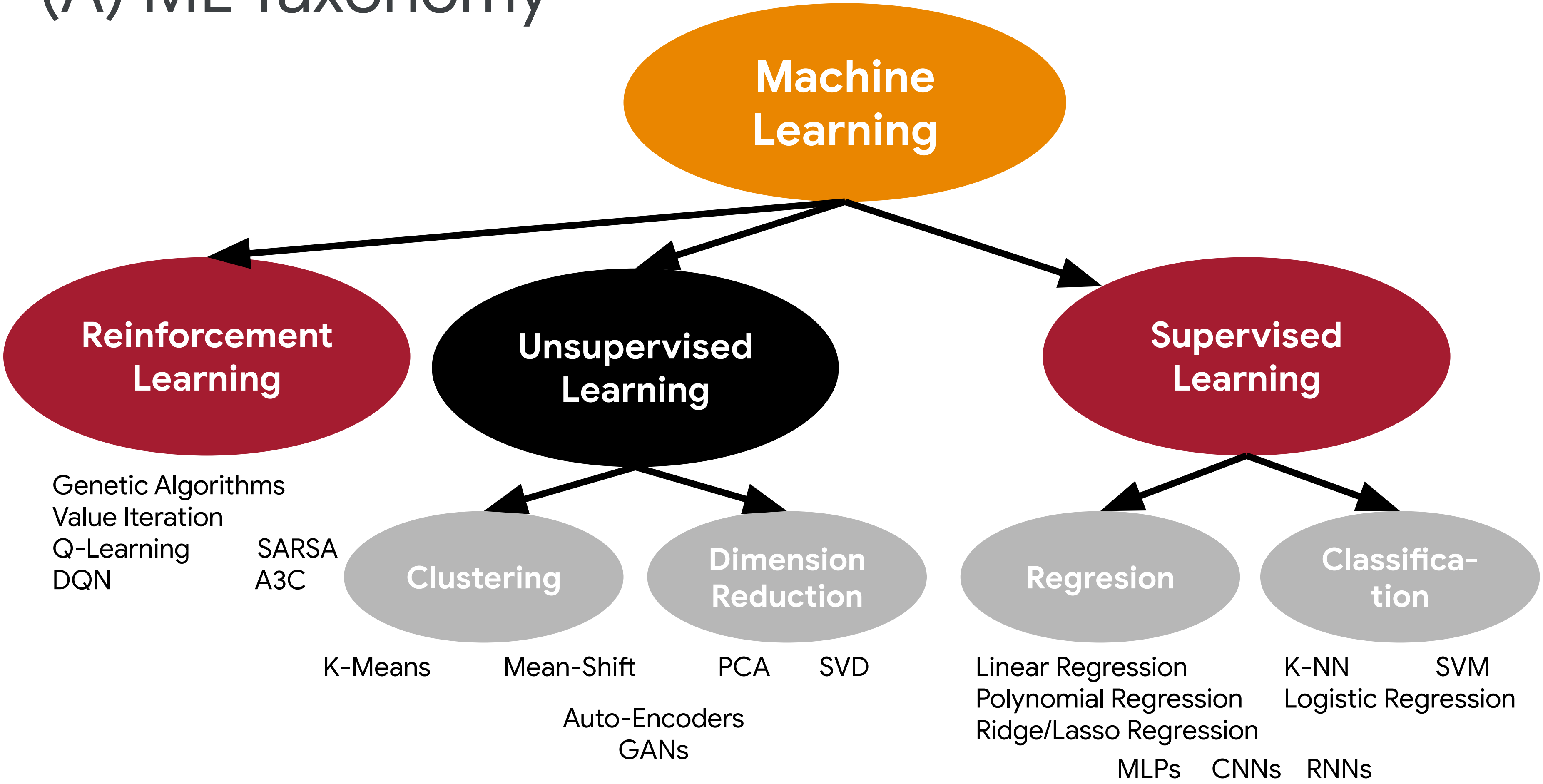
Deep Q Learning

https://blogs.oracle.com/datascience/reinforcement-learning-deep-q-networks

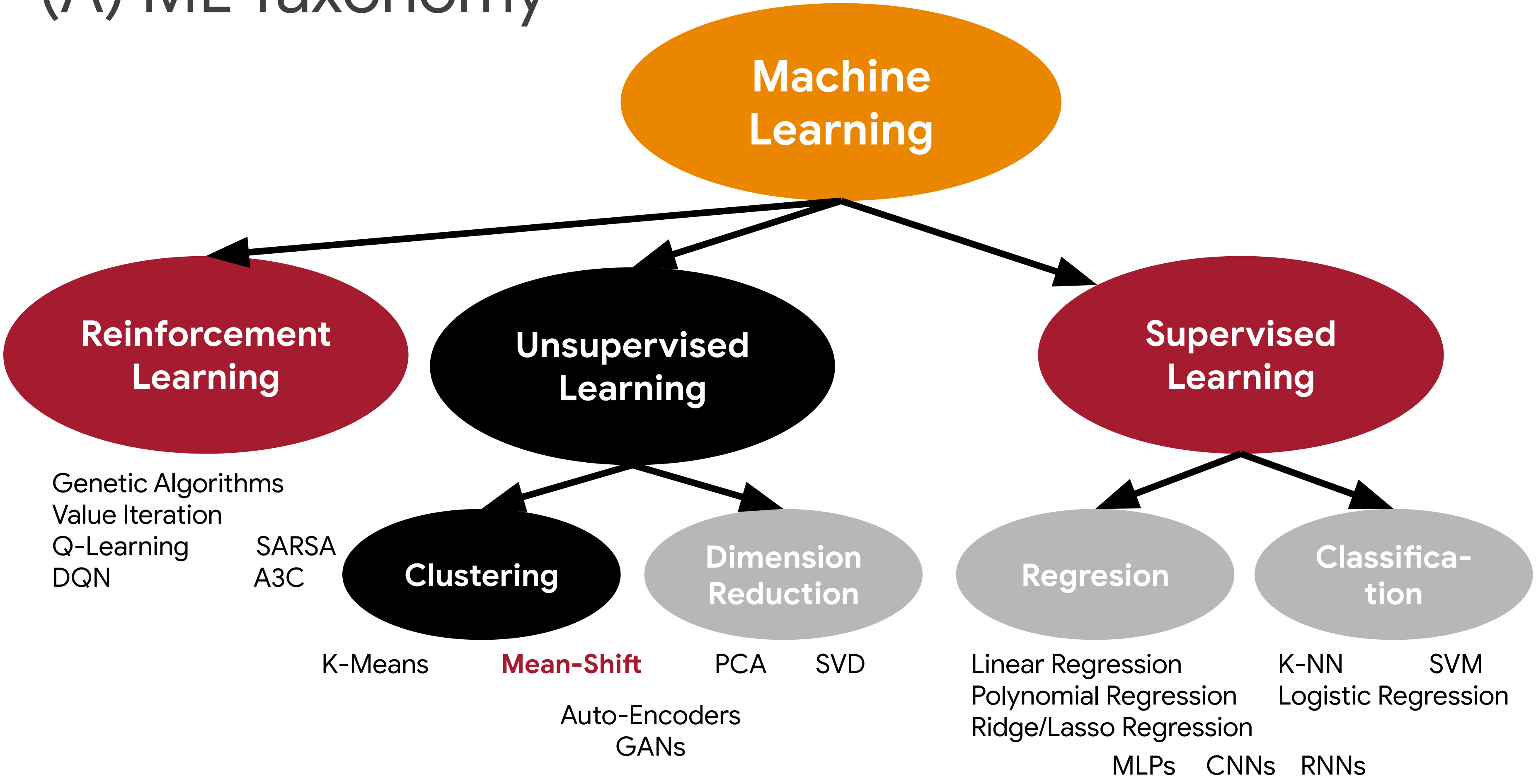
Reinforcement Learning with DQN

https://ai.googleblog.com/2018/06/scalable-deep-reinforcement-learning.html

(A) ML Taxonomy



(A) ML Taxonomy



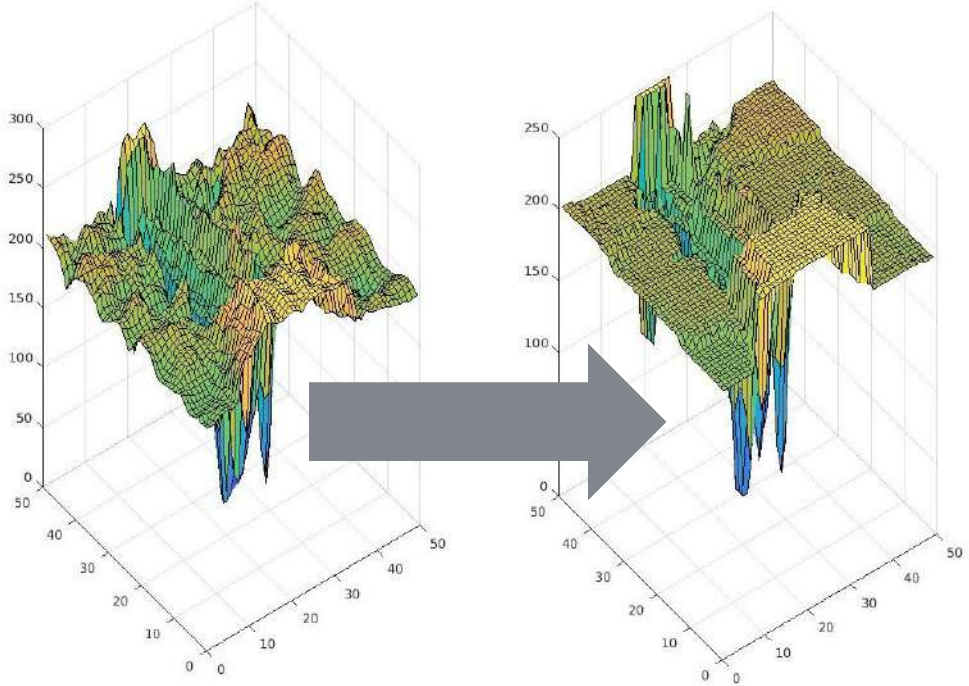
Unsupervised Learning Clustering with Mean Shift

Unsupervised Learning takes in **unlabeled** data and tries to determine some kind of relationships present in the data

Unsupervised Learning
Clustering with Mean Shift

Unsupervised Learning takes in **unlabeled** data and tries to determine some kind of relationships present in the data

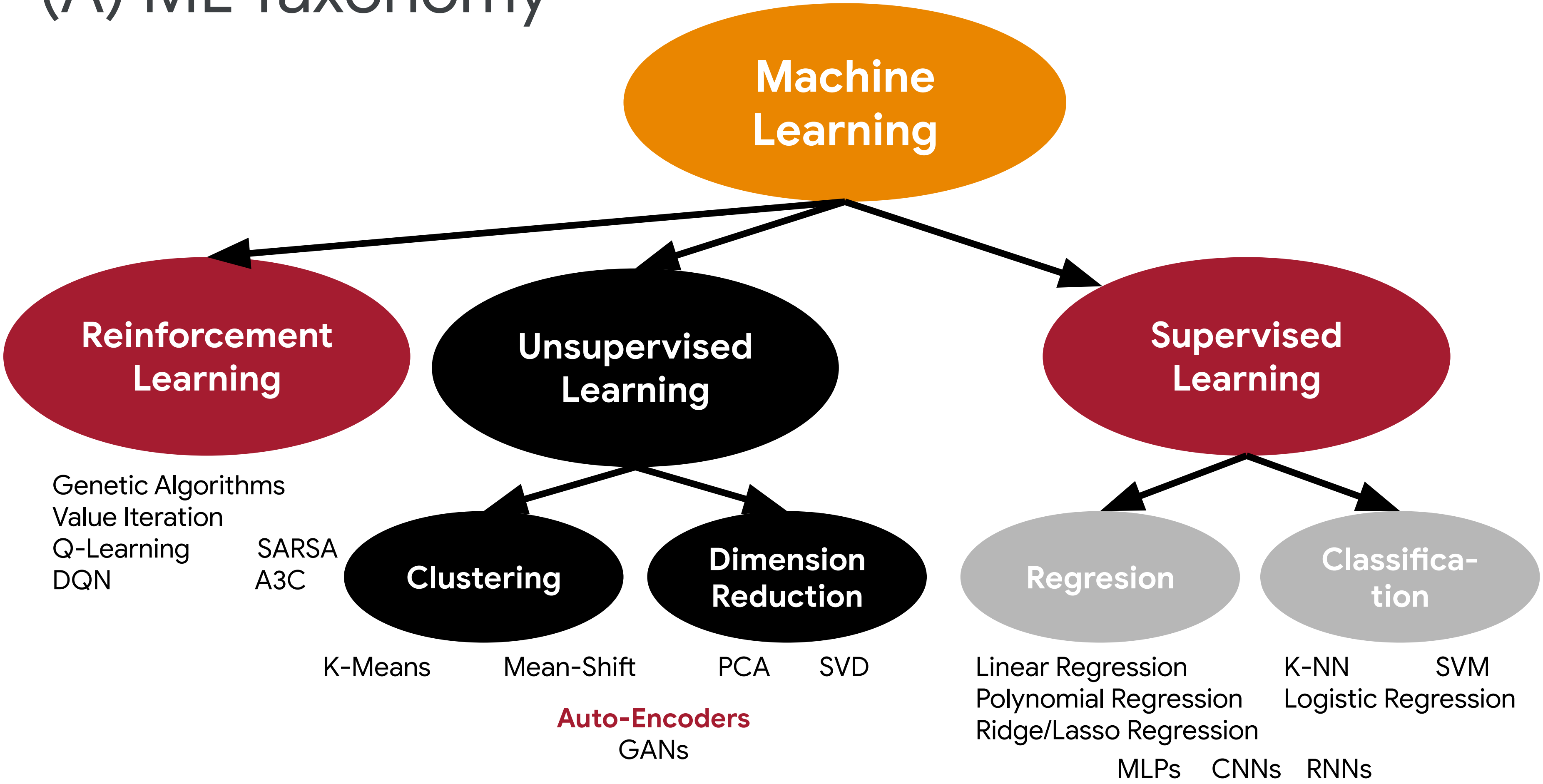
The **mean shift** algorithm finds **clusters** of data by **spatially** averaging values across an image to determine clusters



Unsupervised Learning
Clustering with Mean Shift

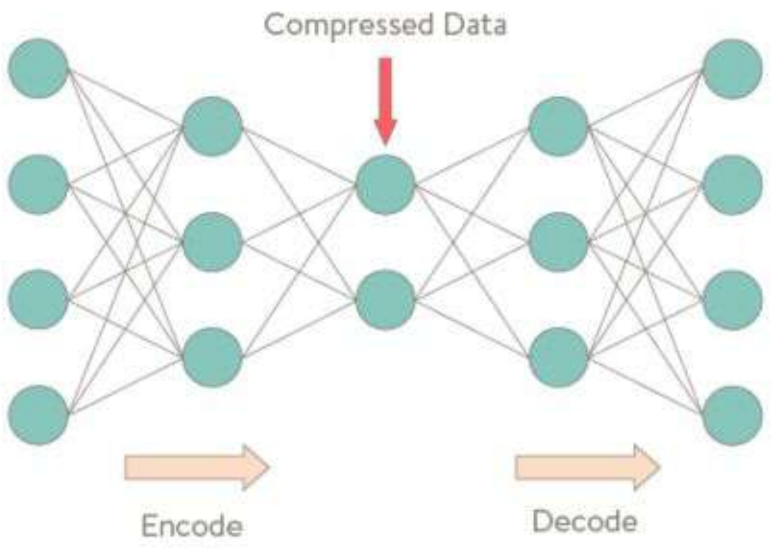
Unsupervised Learning
Clustering with Mean Shift

(A) ML Taxonomy



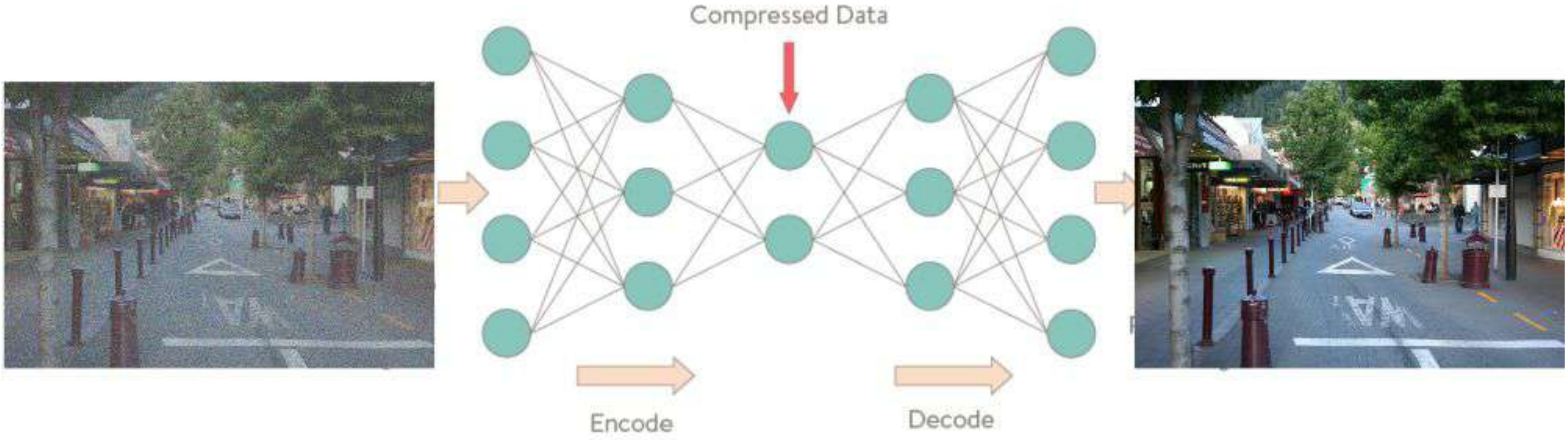
Unsupervised Learning
Autoencoders

Autoencoders are Neural Networks with a **bottleneck** in the middle which is used to get a **latent (lower dimensional) representation** of the input data



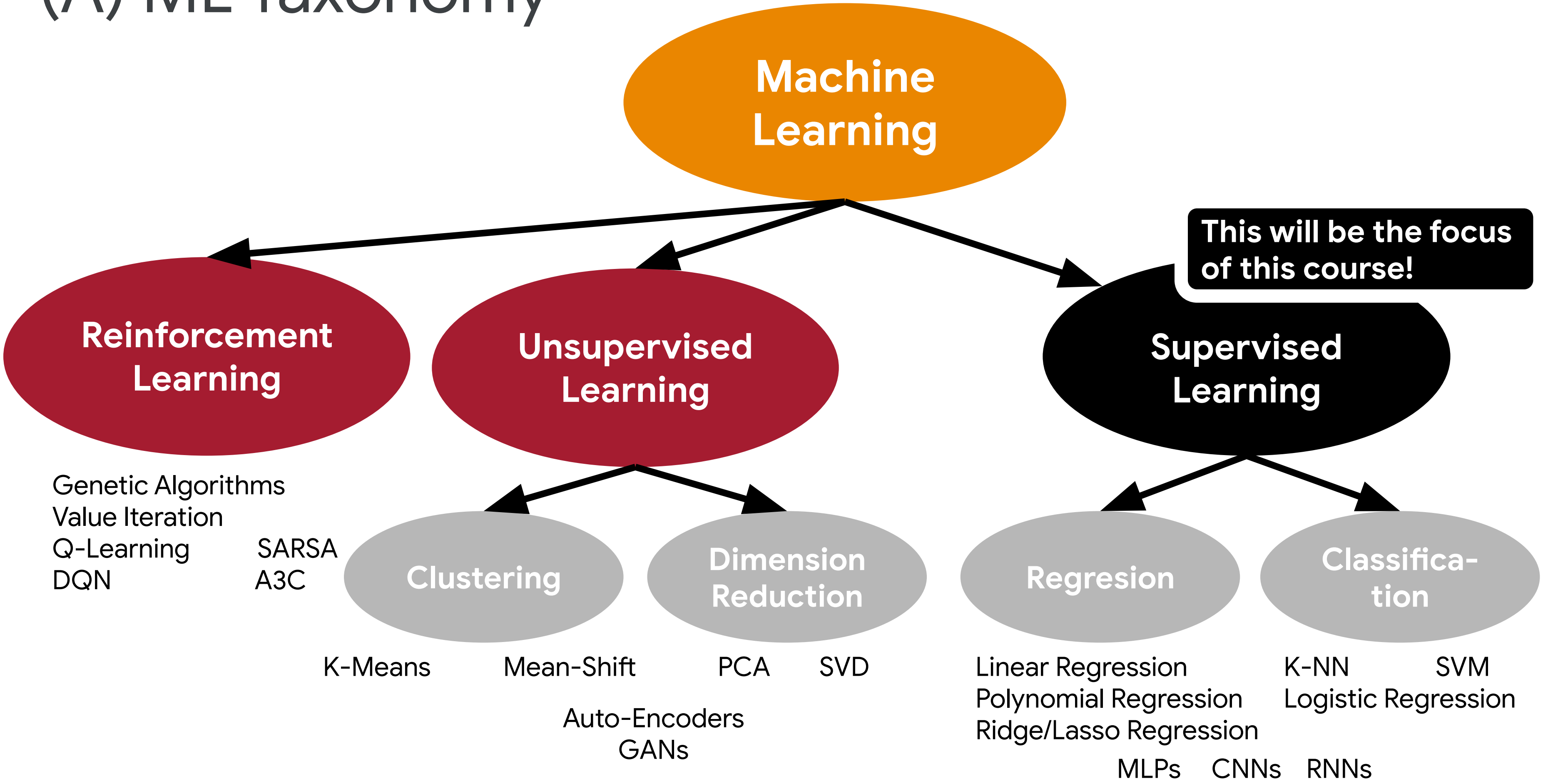
https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726

Unsupervised Learning
Autoencoders



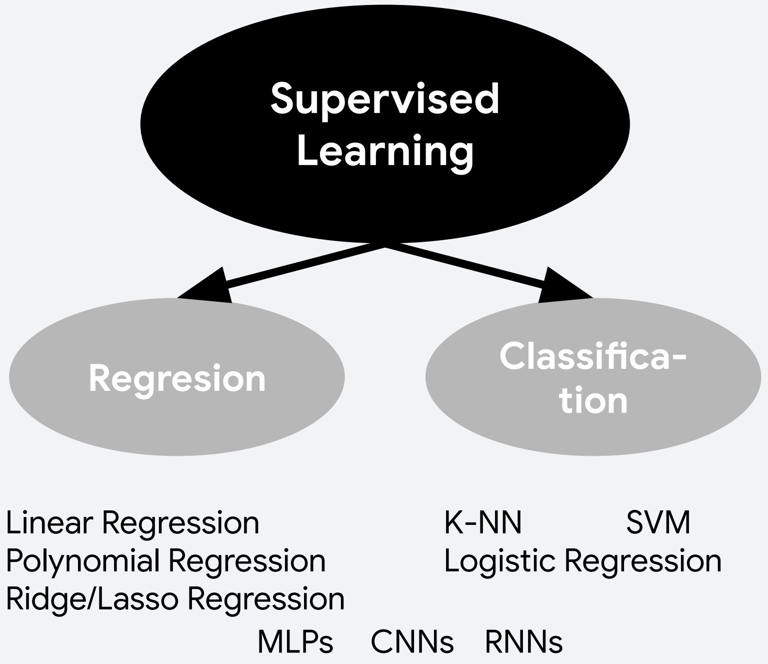
https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726

(A) ML Taxonomy



Supervised Learning

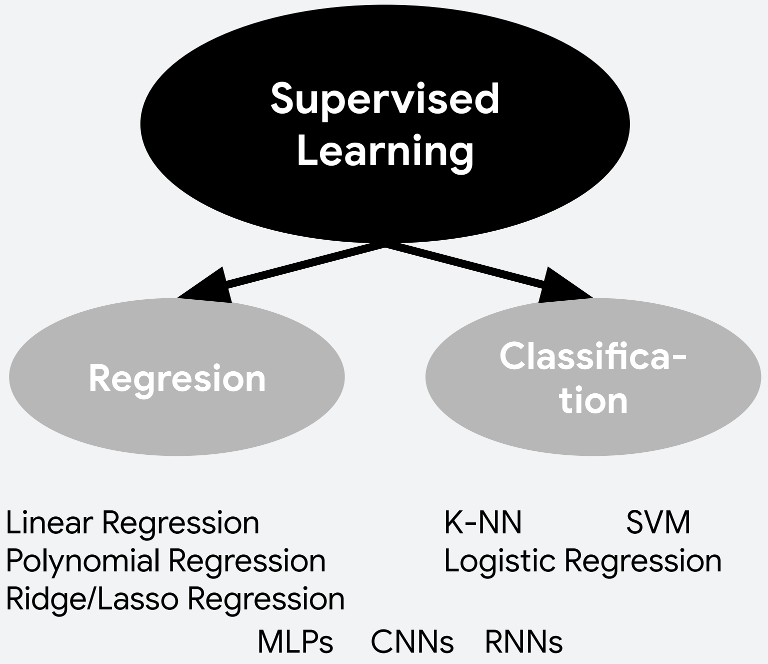
Classification is when the output is designed to be used as a **class** (e.g., which animal is in a picture).



Supervised Learning

Classification is when the output is designed to be used as a **class** (e.g., which animal is in a picture).

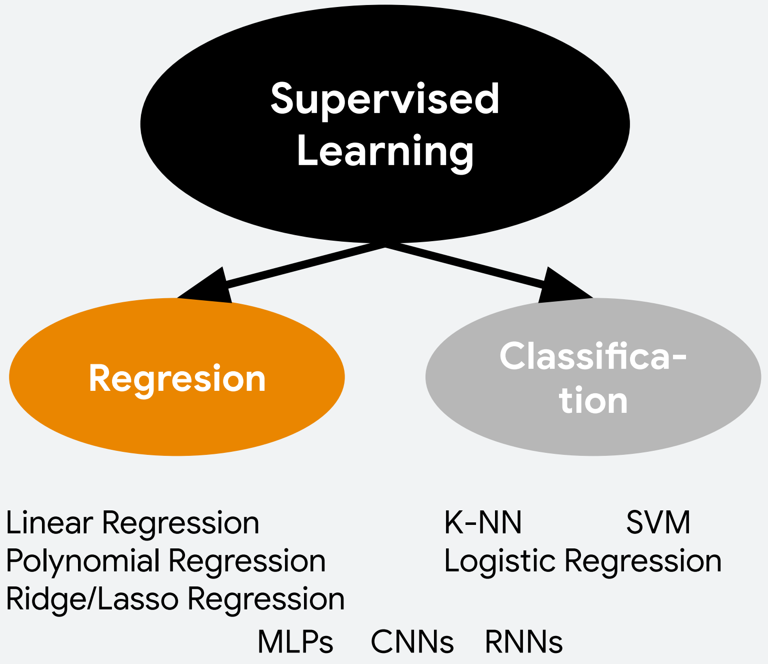
Regression is when the output is designed to be used as a **value** (e.g., the percentage of the vote a candidate will receive)



Supervised Learning

Classification is when the output is designed to be used as a **class** (e.g., which animal is in a picture).

Regression is when the output is designed to be used as a **value** (e.g., the percentage of the vote a candidate will receive)



Classical Supervised Learning:
Regression

Regression

Regression is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

$$f_\theta(x) \rightarrow \hat{y}$$

Regression

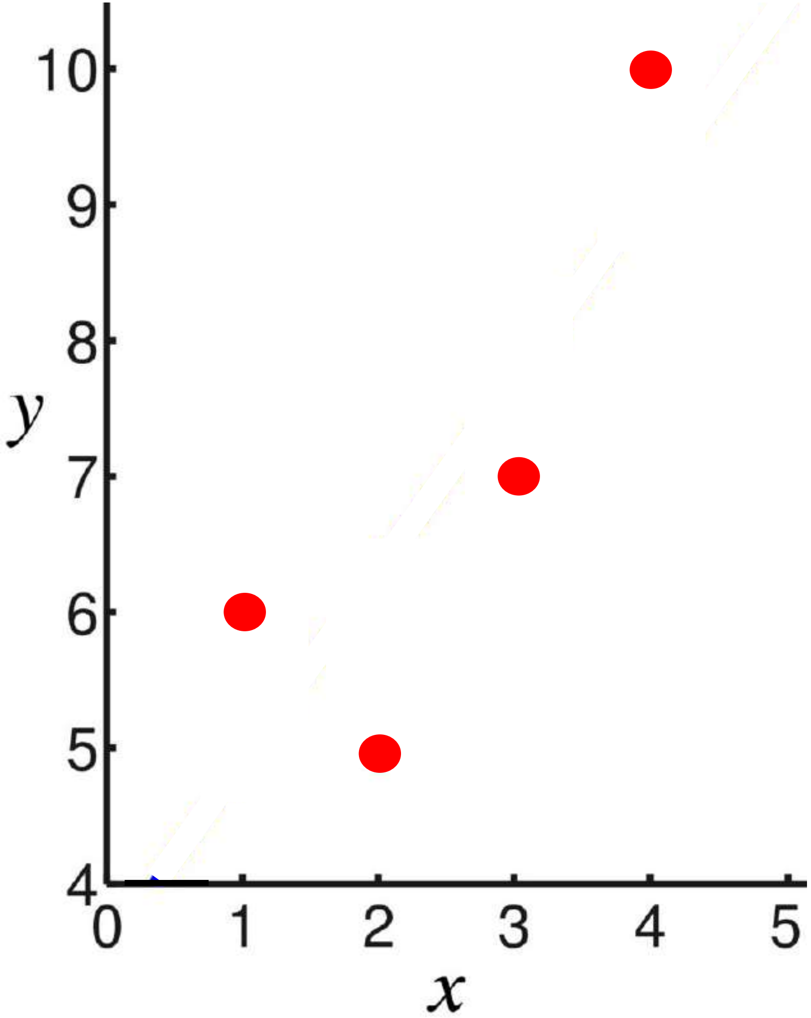
Regression is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

$$f_\theta(x) \rightarrow \hat{y}$$

Let's start by exploring 2D
Linear Regression

Linear Regression

Data



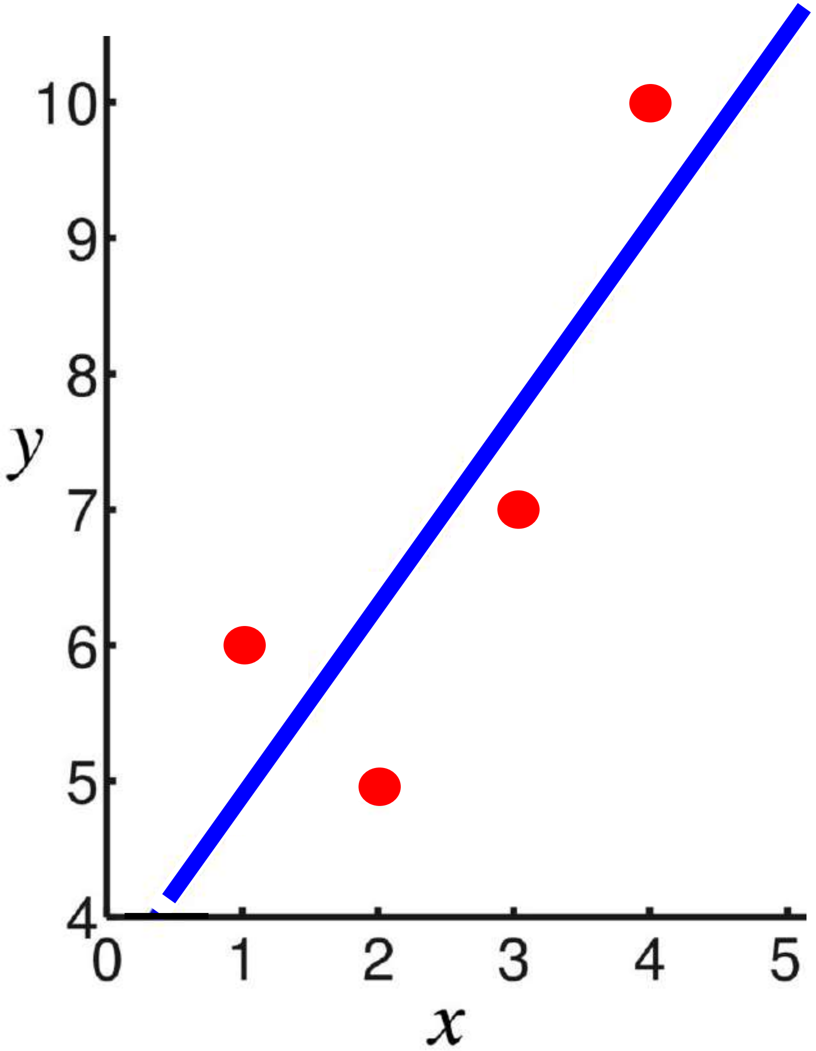
Linear Regression

Data

Model

$$f_\theta : \hat{y} = \theta_1 x + \theta_0$$

Slope Intercept



Linear Regression

Data

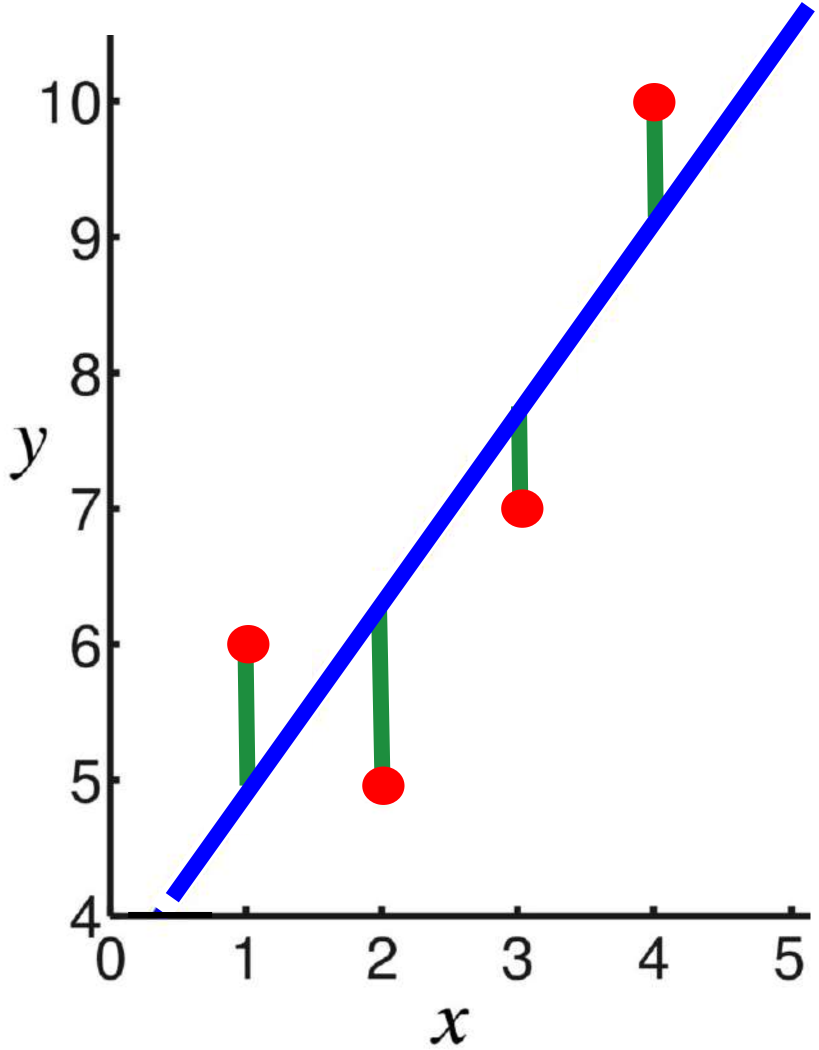
Model

$$f_\theta : \hat{y} = \theta_1 x + \theta_0$$

Slope Intercept

Error

$$e_i : y_i - (\theta_1 x_i + \theta_0)$$



Optimizing the Model

The model's parameters can be optimized through the use of a **loss function**:

$$\min_\theta \ell(\vec{e})$$

Optimizing the Model

The model's parameters can be optimized through the use of a **loss function**:

$$\min_\theta \ell(\vec{e})$$

A common loss function is the **Mean Squared Error (MSE)**

$$\min_\theta \sum_{i=0}^{N} (e_i)^2$$

Optimizing the Model

The model's parameters can be optimized through the use of a **loss function**:

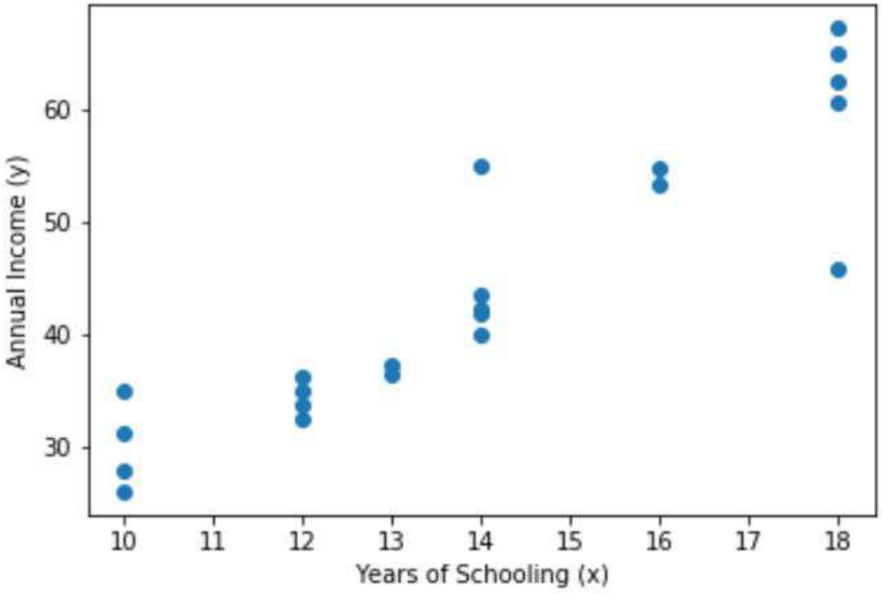
$$\min_\theta \ell(\vec{e})$$

A common loss function is the **Mean Squared Error (MSE)**

$$\min_\theta \sum_{i=0}^{N} (e_i)^2$$

Lets walk through an example of optimizing a model using MSE for some example data!

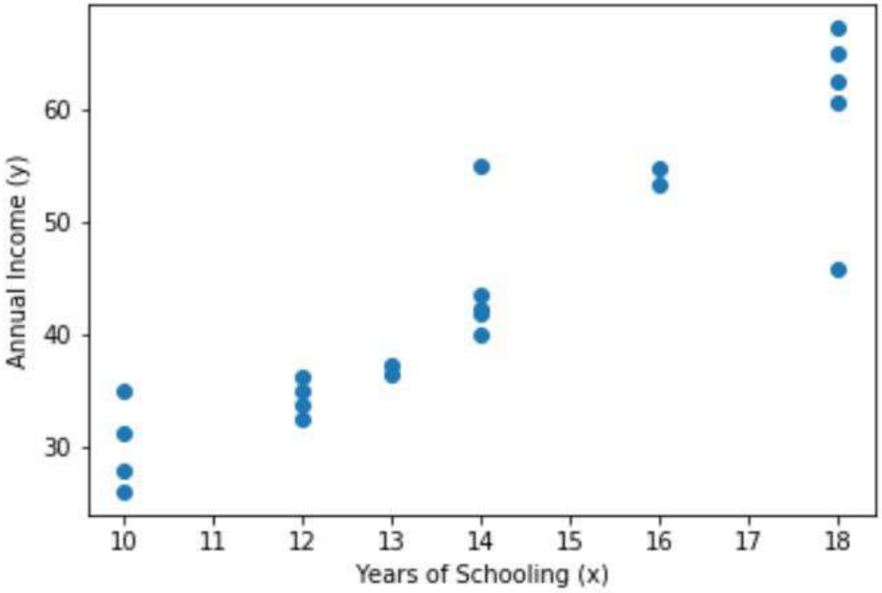
Stay in School!



This is based off of **REAL DATA!**

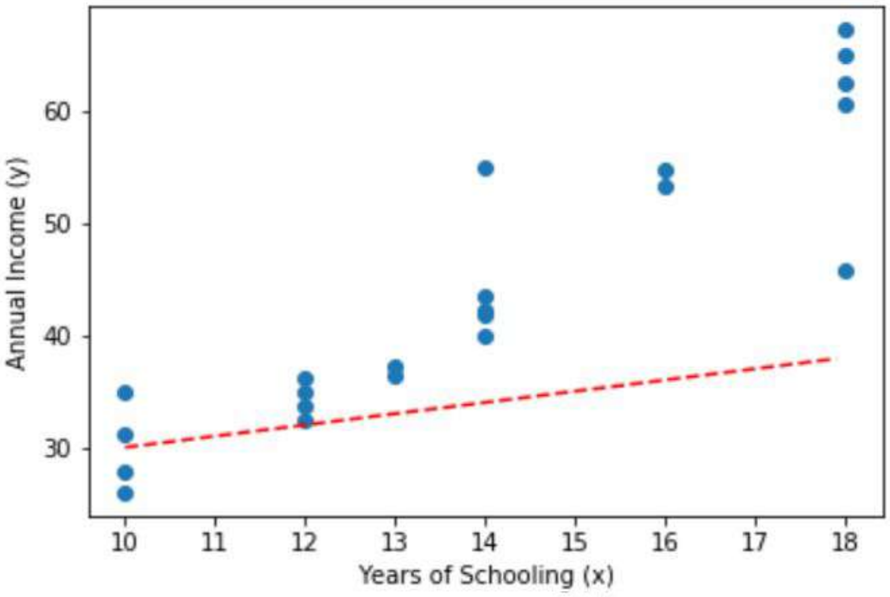
https://nces.ed.gov/programs/coe/indicator_cba.asp

Stay in School!

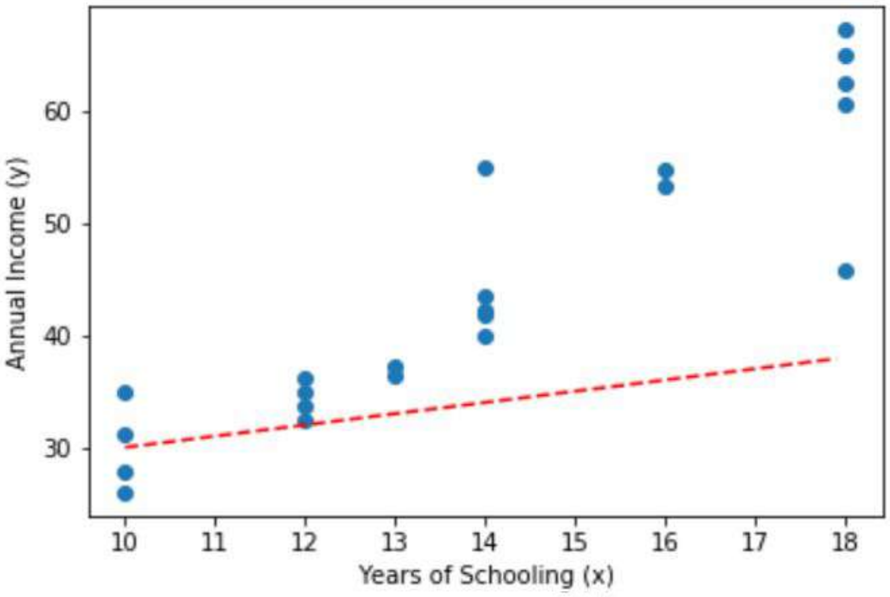


First let's take an initial guess of what the best fine line might be!

First let’s take an initial guess!

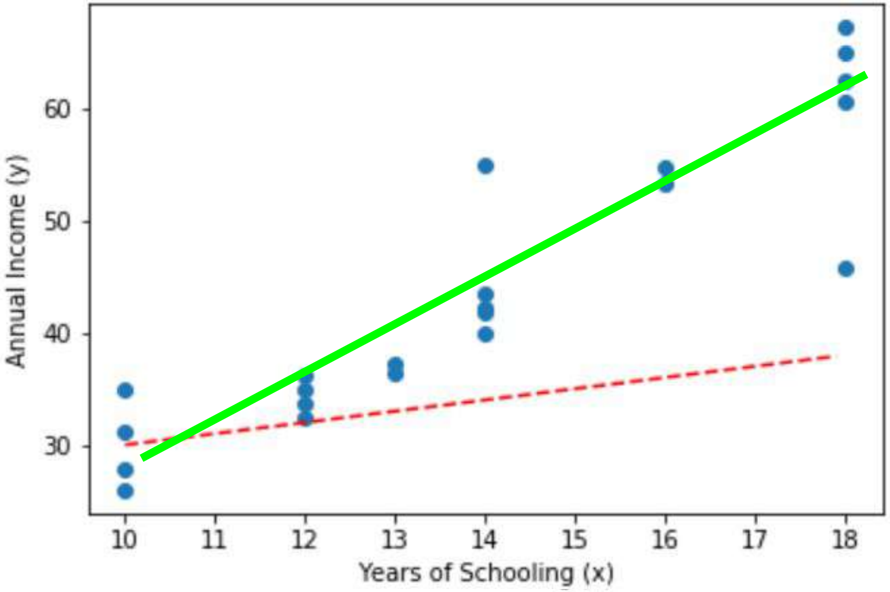


First let's take an initial guess!



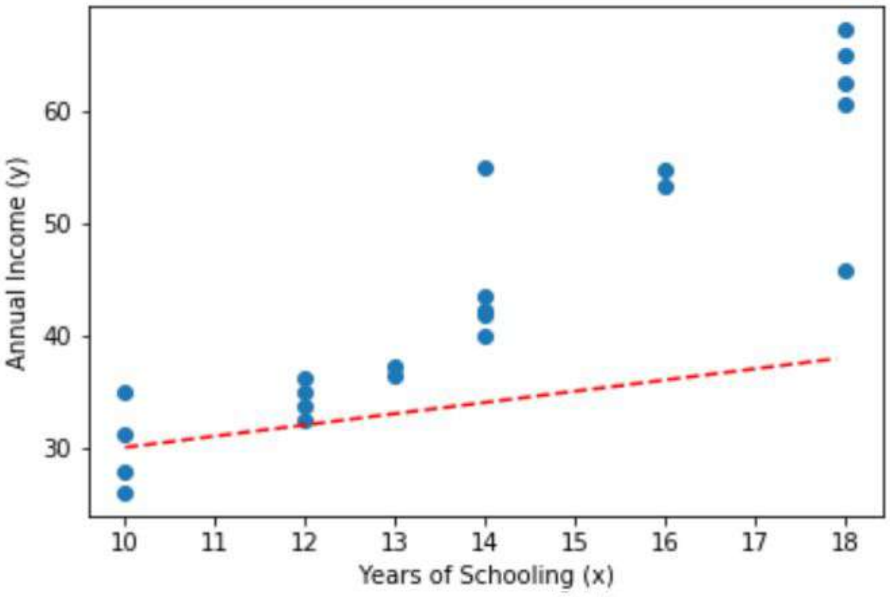
Can we do better?

First let's take an initial guess!



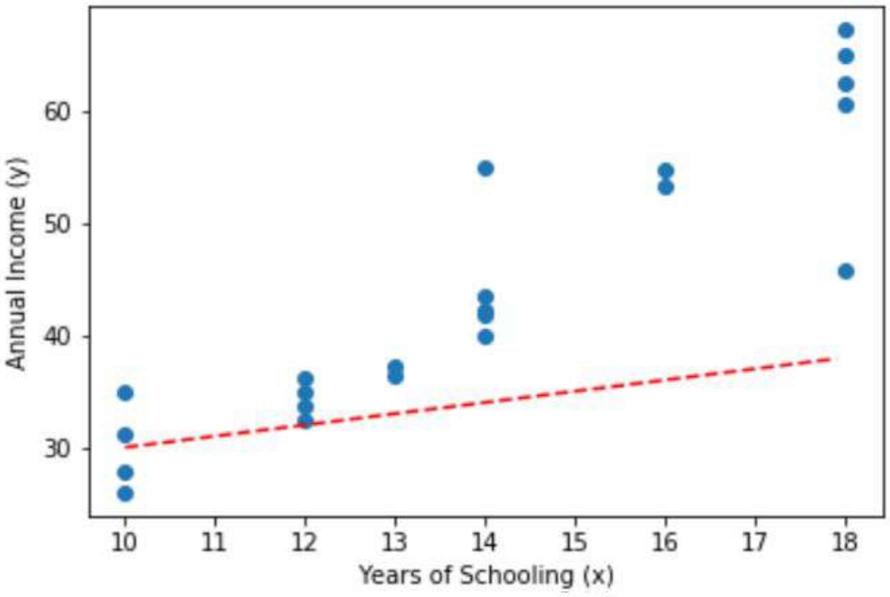
Can we do better?

First let's take an initial guess!

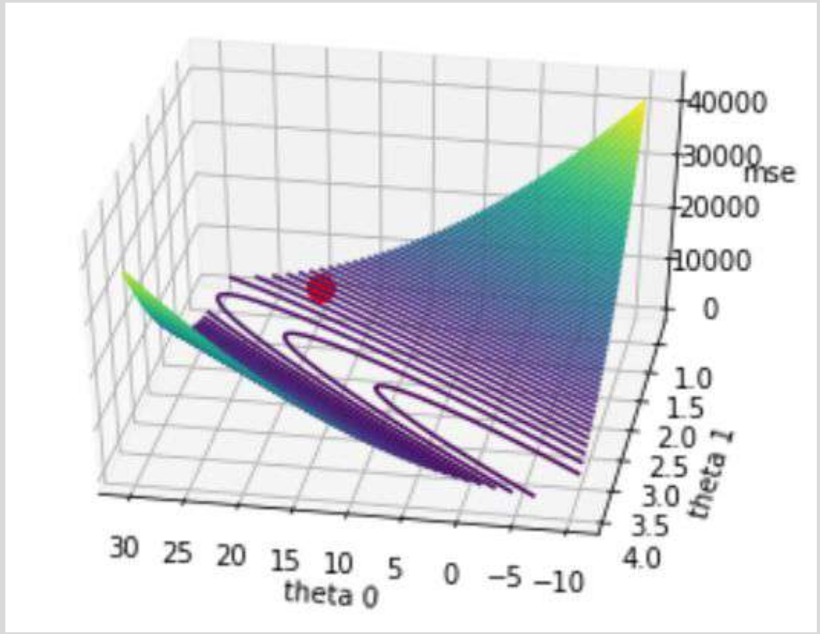


Lets plot the MSE

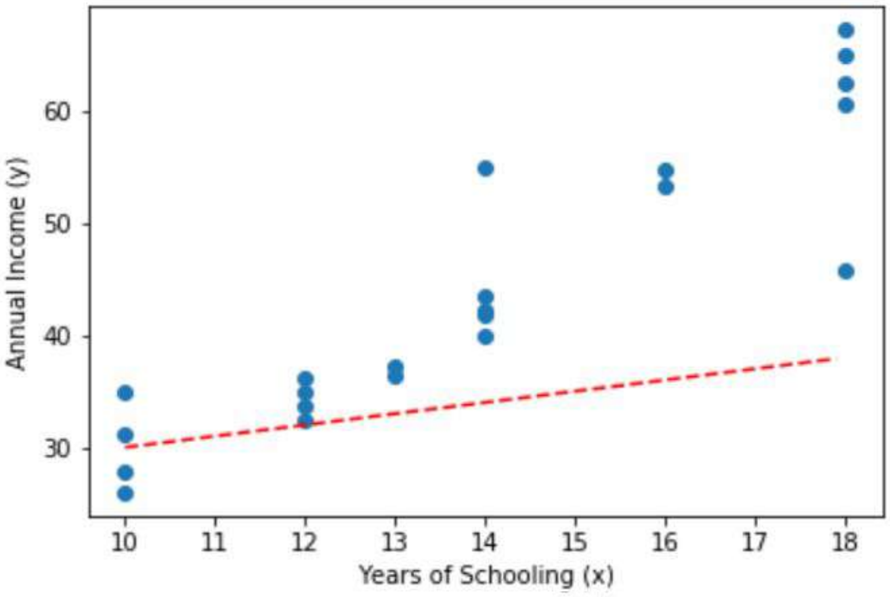
First let's take an initial guess!

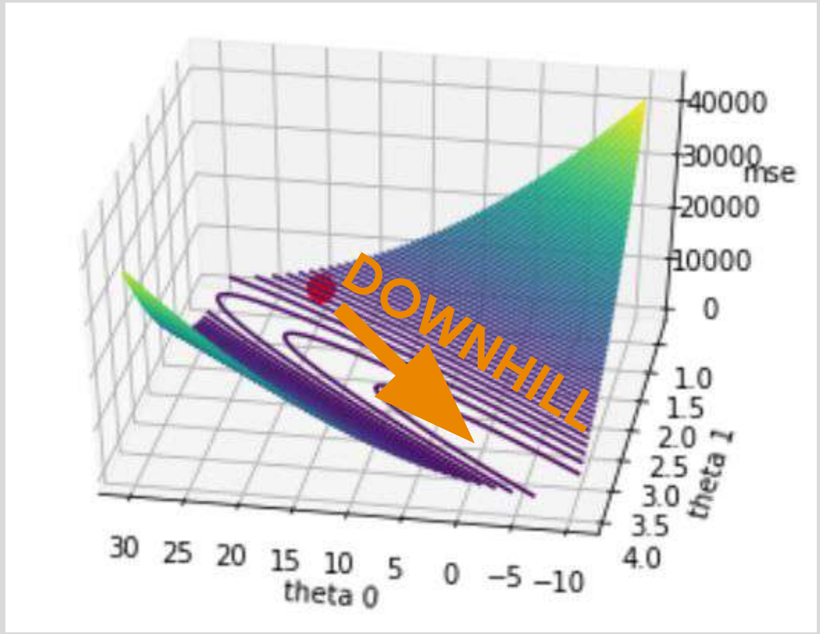


Lets plot the MSE

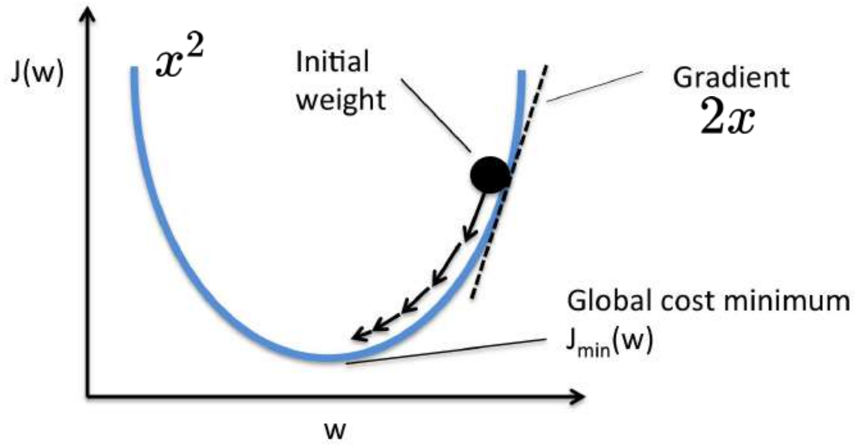


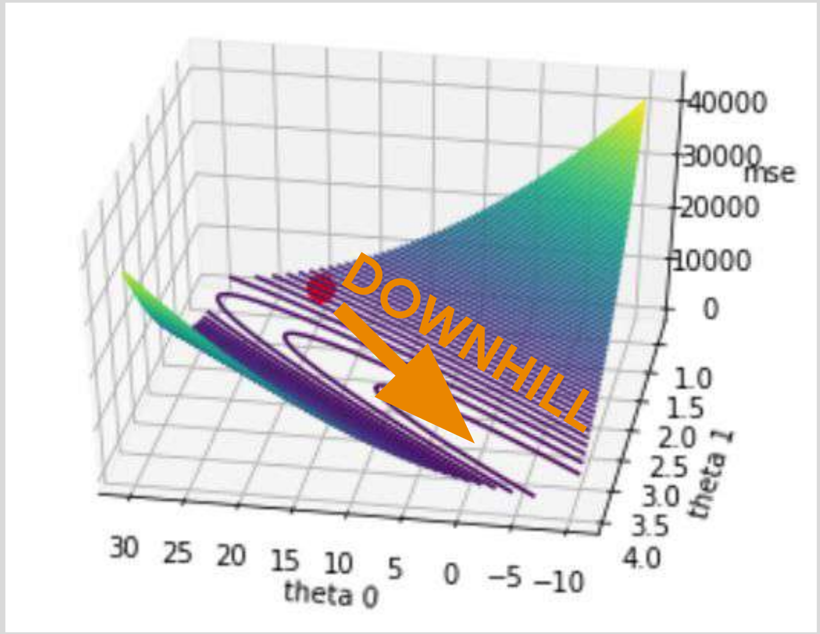
First let's take an initial guess!





First let's take an initial guess!

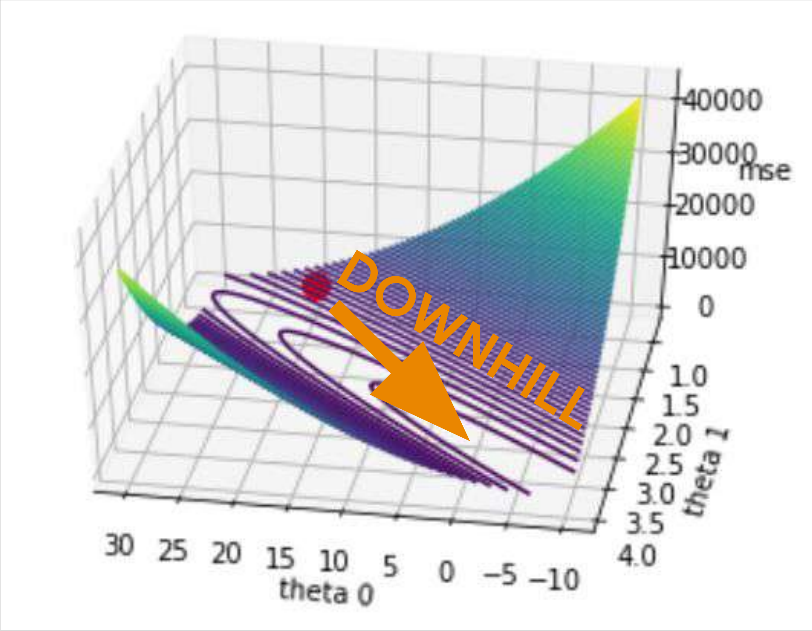




Gradient Descent

$$\theta \leftarrow \theta - \alpha \nabla MSE$$

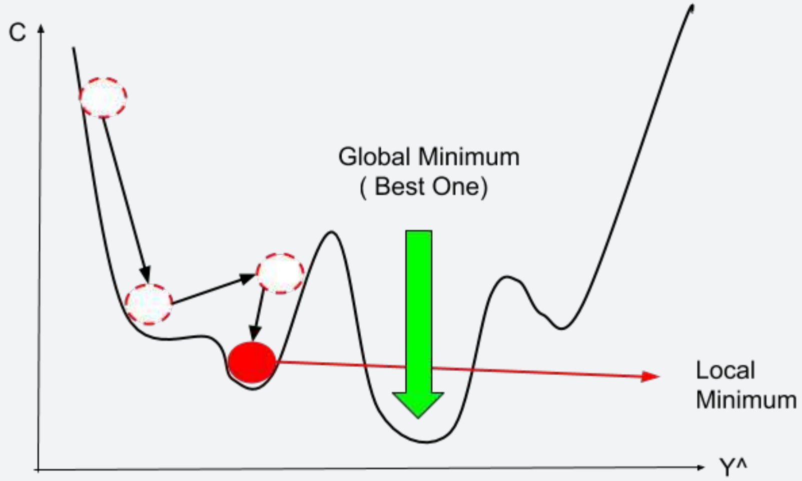
It turns out that if one moves in direction of the **negative gradient** according to some **step size (learning rate)** you will move toward the optimum



Gradient Descent

$$\theta \leftarrow \theta - \alpha \nabla MSE$$

It turns out that if one moves in direction of the **negative gradient** according to some **step size (learning rate)** you will move toward the optimum

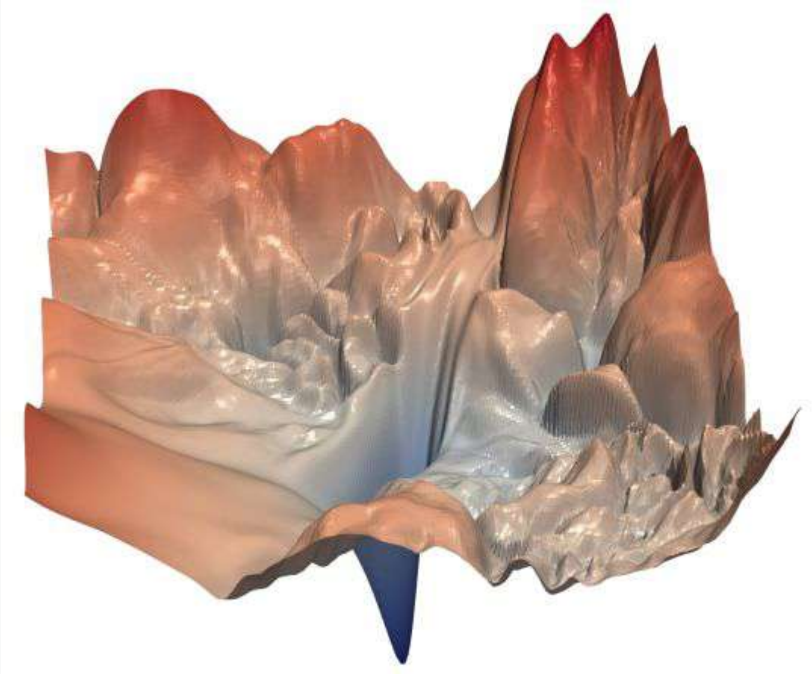


https://www.mltut.com/stochastic-gradient-descent-a-super-easy-complete-guide/

Gradient Descent

$$\theta \leftarrow \theta - \alpha \nabla MSE$$

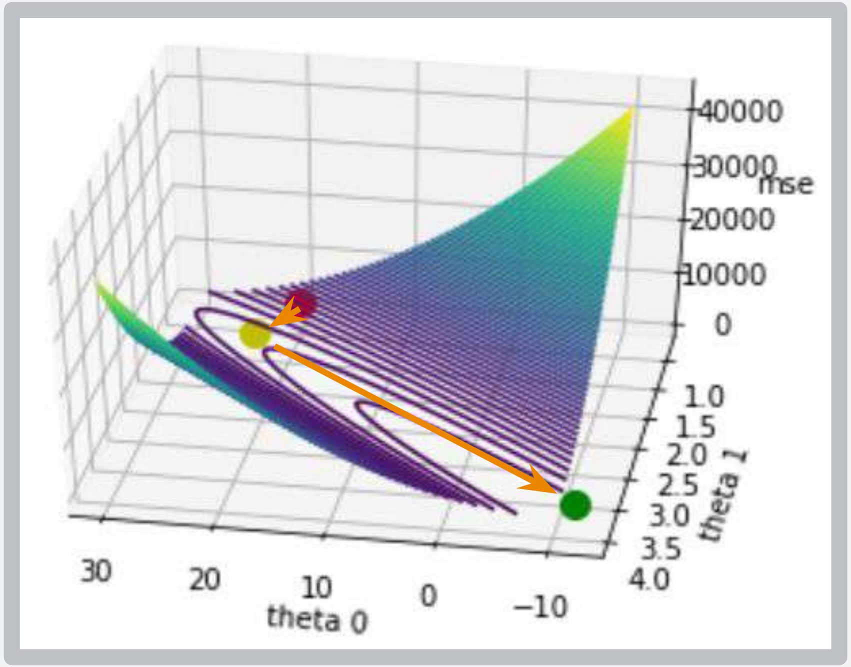
It turns out that if one moves in direction of the **negative gradient** according to some **step size (learning rate)** you will move toward the optimum



https://arxiv.org/pdf/1712.09913.pdf

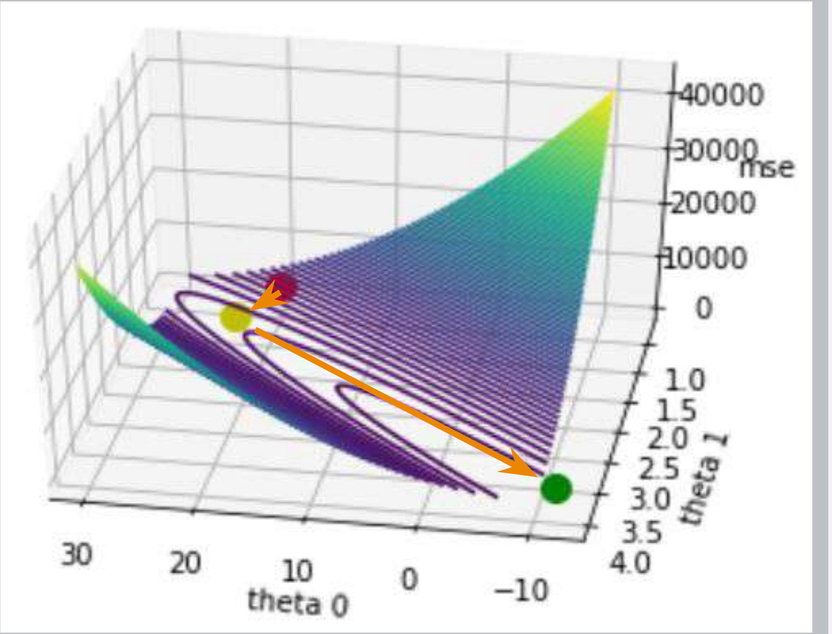
Gradient Descent

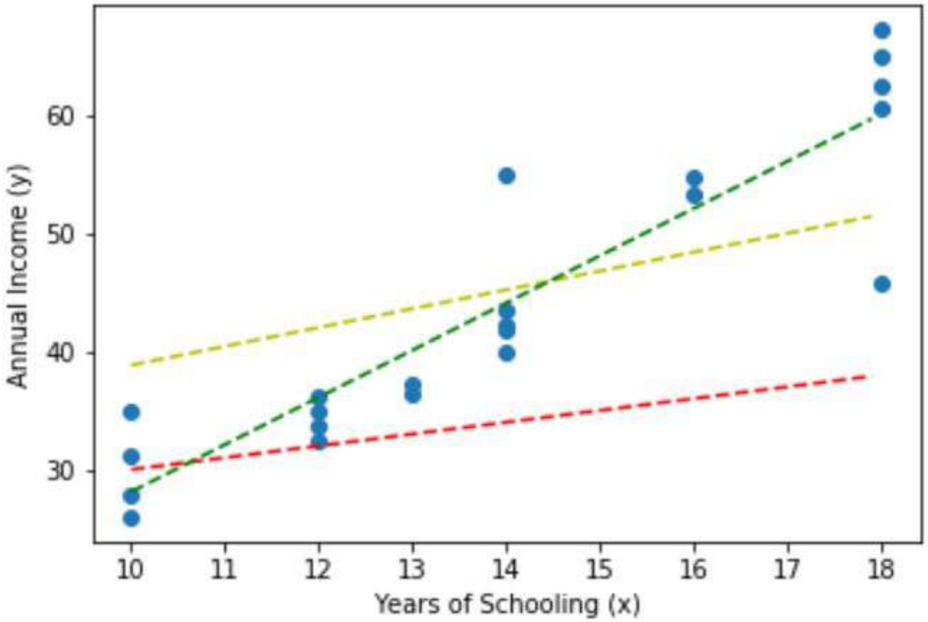
$$\theta \leftarrow \theta - \alpha \nabla MSE$$



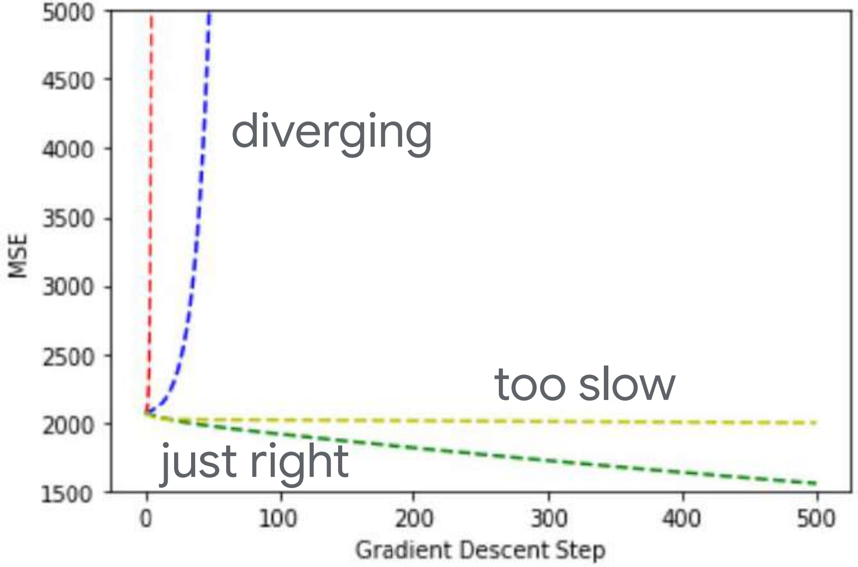
Gradient Descent

$$\theta \leftarrow \theta - \alpha \nabla MSE$$





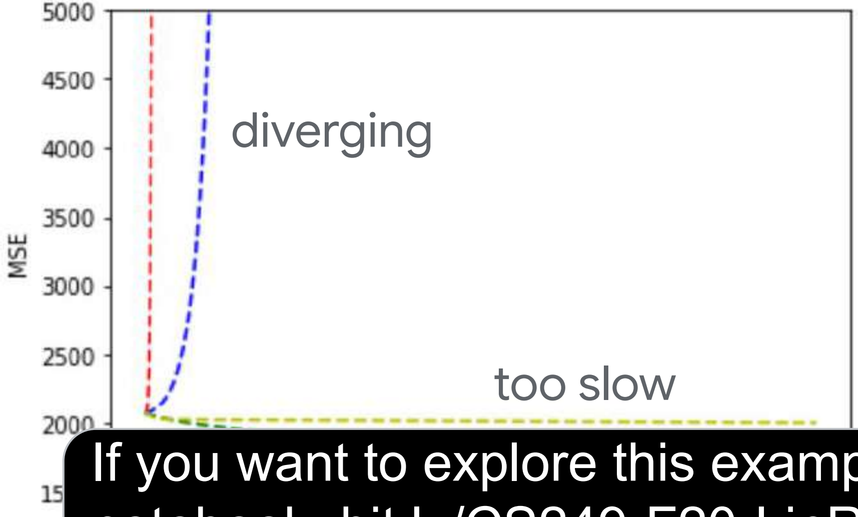
The Learning Rate is a **hyperparameter**



$$\theta \leftarrow \theta - \alpha \nabla MSE$$

- Set it **too low** and it will **take forever** and **get stuck** in local minima
- Set it **too high** and it will **diverge**

The Learning Rate is a **hyperparameter**



$$\theta \leftarrow \theta - \alpha \nabla MSE$$

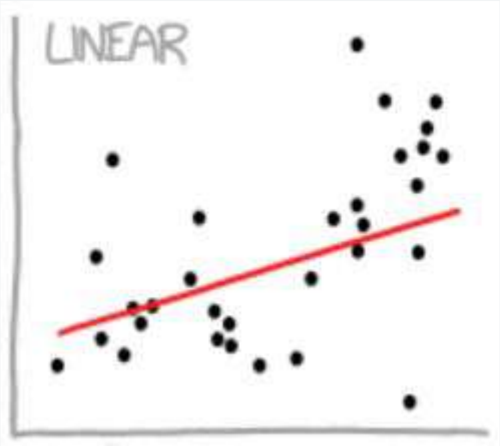
- Set it **too low** and it will **take forever** and **get stuck** in local minima
- Set it **too high** and it will

If you want to explore this example further I made an iPython notebook: bit.ly/CS249-F20-LinReg
And found this other notebook: http://bit.ly/CS249-F20-LinReg2

Beyond Linear Regression

- Our model can be any function of the input

$$f_\theta(x) \rightarrow \hat{y}$$

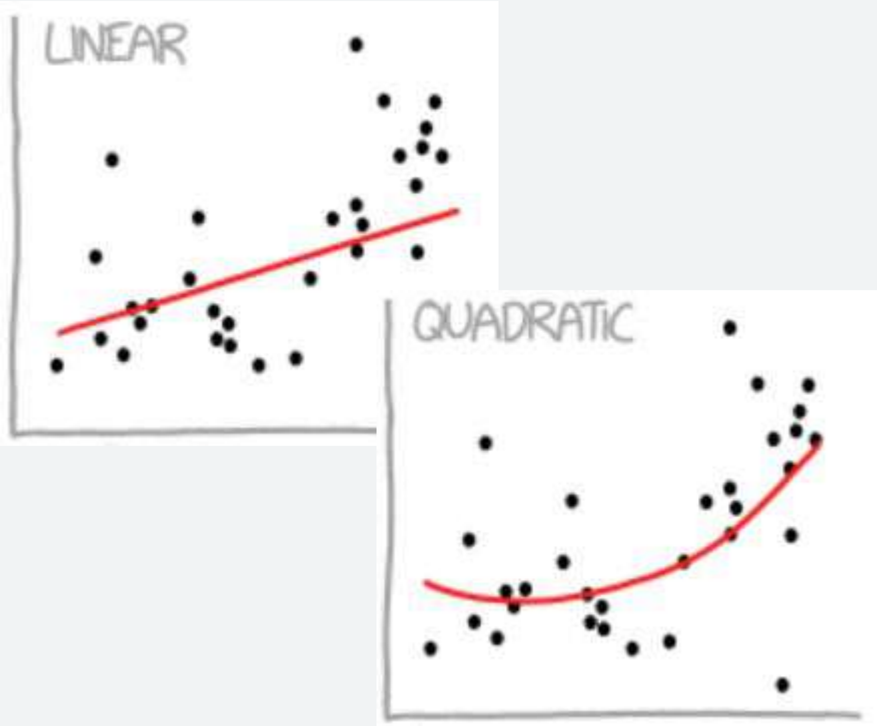


https://xkcd.com/2048/

Beyond Linear Regression

- Our model can be any function of the input
- More complex functions:
  - fit **more complex data**

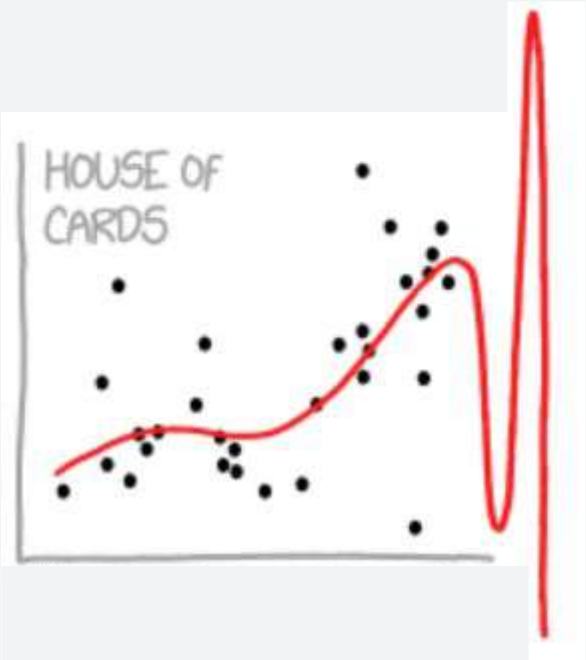
$$f_\theta(x) \rightarrow \hat{y}$$



https://xkcd.com/2048/

Beyond Linear Regression

- Our model can be any function of the input
- More complex functions:
  - fit **more complex data**
  - can **overfit data**



https://xkcd.com/2048/

Beyond Linear Regression

- Our model can be any function of the input
- More complex functions:
    - fit **more complex data**
    - can **overfit data**

Separate out data sets for training and testing to check for overfitting!

https://xkcd.com/2048/

Regularization to avoid overfitting

- One simple way to reduce overfitting is to penalize the model for reacting strongly to the data e.g.,

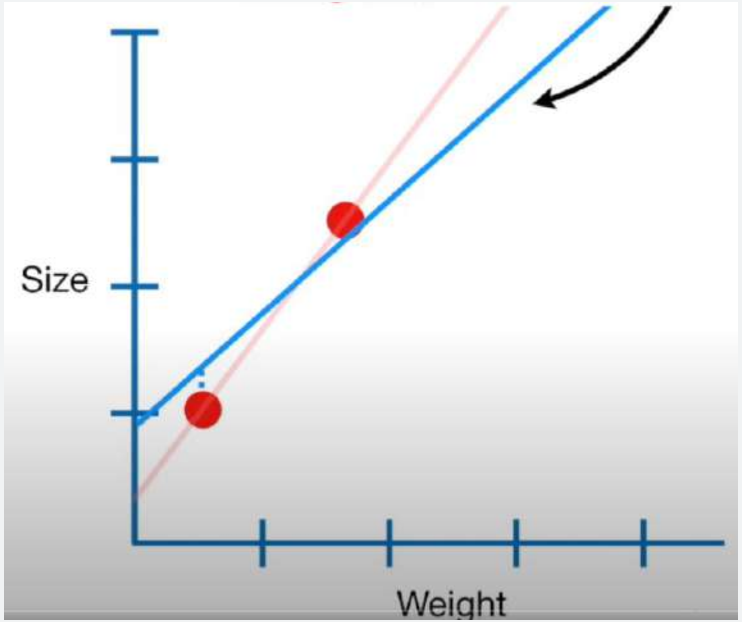
$$\min_\theta \sum_{i=0}^{N} (e_i)^2 \boxed{+\ \theta^2}$$

https://www.youtube.com/watch?v=Q81RR3yKn30

Regularization to avoid overfitting

- One simple way to reduce overfitting is to penalize the model for reacting strongly to the data e.g.,

$$\min_\theta \sum_{i=0}^{N} (e_i)^2 \boxed{+\ \theta^2}$$

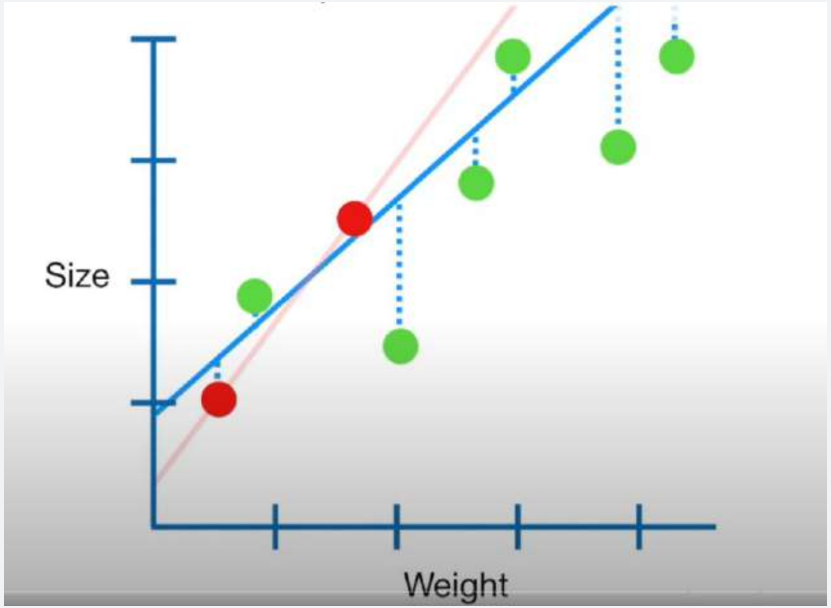


https://www.youtube.com/watch?v=Q81RR3yKn30

Regularization to avoid overfitting

- One simple way to reduce overfitting is to penalize the model for reacting strongly to the data e.g.,

$$\min_\theta \sum_{i=0}^{N} (e_i)^2 \boxed{+ \theta^2}$$



https://www.youtube.com/watch?v=Q81RR3yKn30

Regression

Regression is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

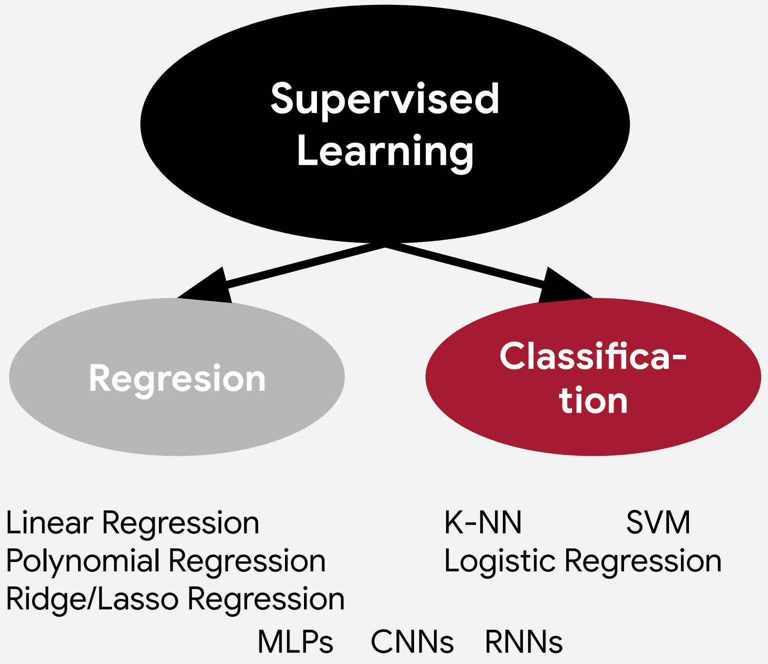
$$f_\theta(x) \rightarrow \hat{y}$$

- Models are often optimized through **gradient descent** on some **loss function** (e.g., MSE)
- **Hyperparameters** like the **learning rate** need to be tuned to have this converge well
- **Regularization** and **separate test data** can help avoid the problem of **overfitting**

Supervised Learning

Classification is when the output is designed to be used as a **class** (e.g., which animal is in a picture).

Regression is when the output is designed to be used as a **value** (e.g., the percentage of the vote a candidate will receive)



Classical Supervised Learning:

Classification

Regression

Regression is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

$$f_\theta(x) \rightarrow \hat{y}$$

Classification

Classification is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

$$f_\theta(x) \rightarrow \hat{y}$$

where the **output is a discrete class (integer)**

Classification

Classification is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

$$f_\theta(x) \rightarrow \hat{y}$$

where the **output is a discrete class (integer)**

Logistic Regression

Uses the same regression machinery plus a **nonlinear activation function** that maps the output into **probability space** (probability of being in a class)

Classification

Classification is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

$$f_\theta(x) \rightarrow \hat{y}$$

where the **output is a discrete class (integer)**

Logistic Regression

Uses the same regression machinery plus a **nonlinear activation function** that maps the output into **probability space** (probability of being in a class) and then a **decision boundary** to decide at what probability something is of a class

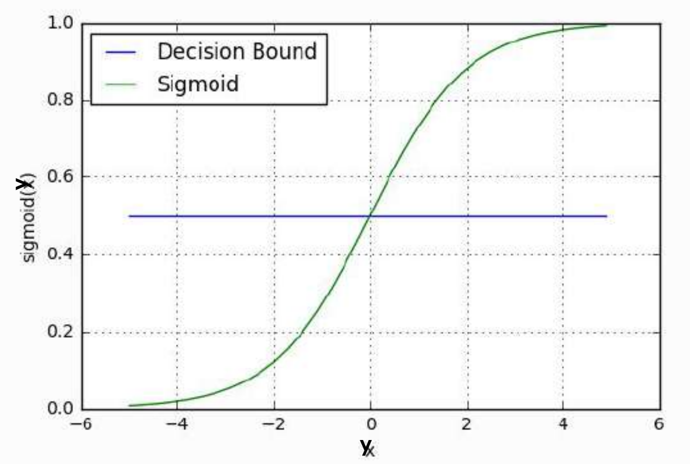
E.g., the Sigmoid Activation Function

Logistic Regression

Uses the same regression machinery plus a **nonlinear activation function** that maps the output into **probability space** (probability of being in a class) and then a **decision boundary** to decide at what probability something is of a class

E.g., the Sigmoid Activation Function

$$S(y) = \frac{1}{1+e^{-y}}$$



$$h_\theta(x) = S(f_\theta(x))$$

Logistic Regression

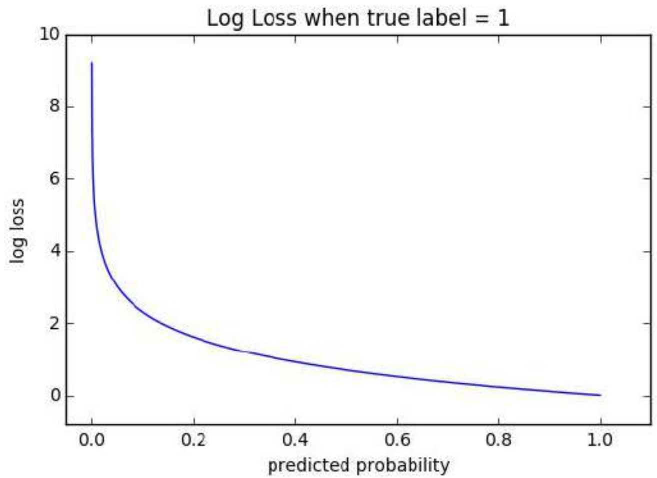
Uses the same regression machinery plus a **nonlinear activation function** that maps the output into **probability space** (probability of being in a class) and then a **decision boundary** to decide at what probability something is of a class

Logistic Regression

We also need a new **loss function** which can better penalize this particular output and can penalize increasingly more for being more sure and wrong

The Cross Entropy Loss Function

$$\ell(x_i, y_{i=1}) = -\log(h_\theta(x_i))$$
$$\ell(x_i, y_{i=0}) = -\log(1 - h_\theta(x_i))$$

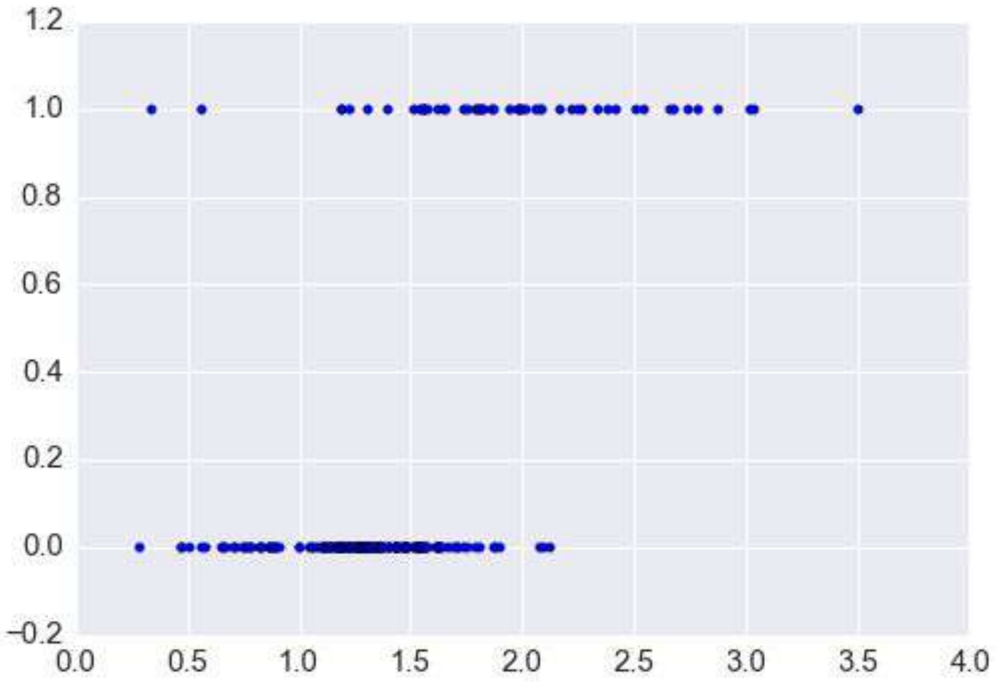


Logistic Regression

We also need a new **loss function** which can better penalize this particular output and can penalize increasingly more for being more sure and wrong

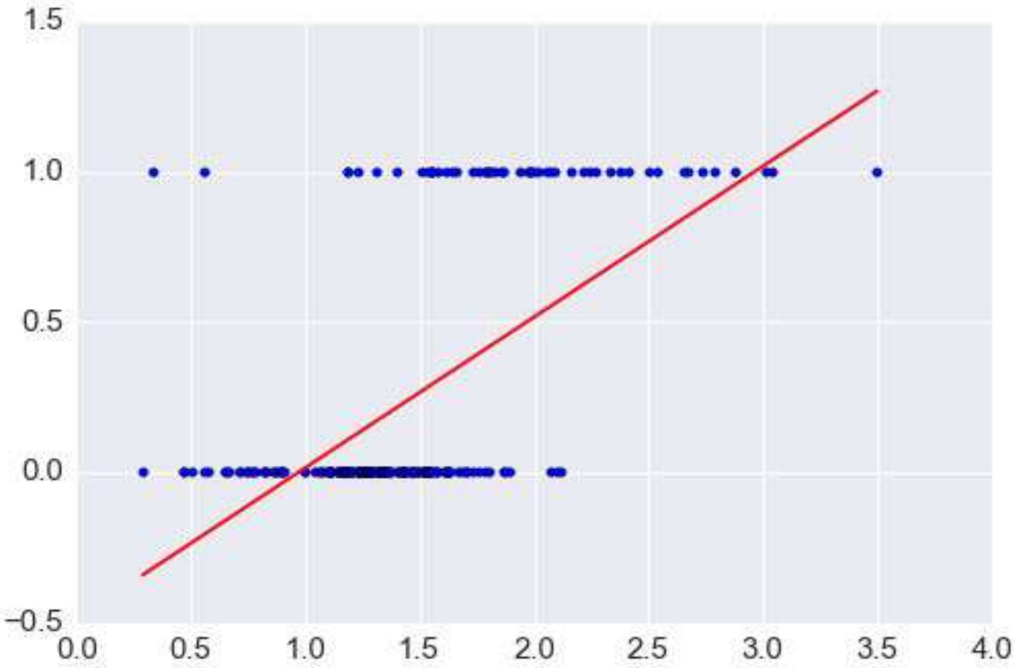
Logistic Regression: Putting it all together

Data



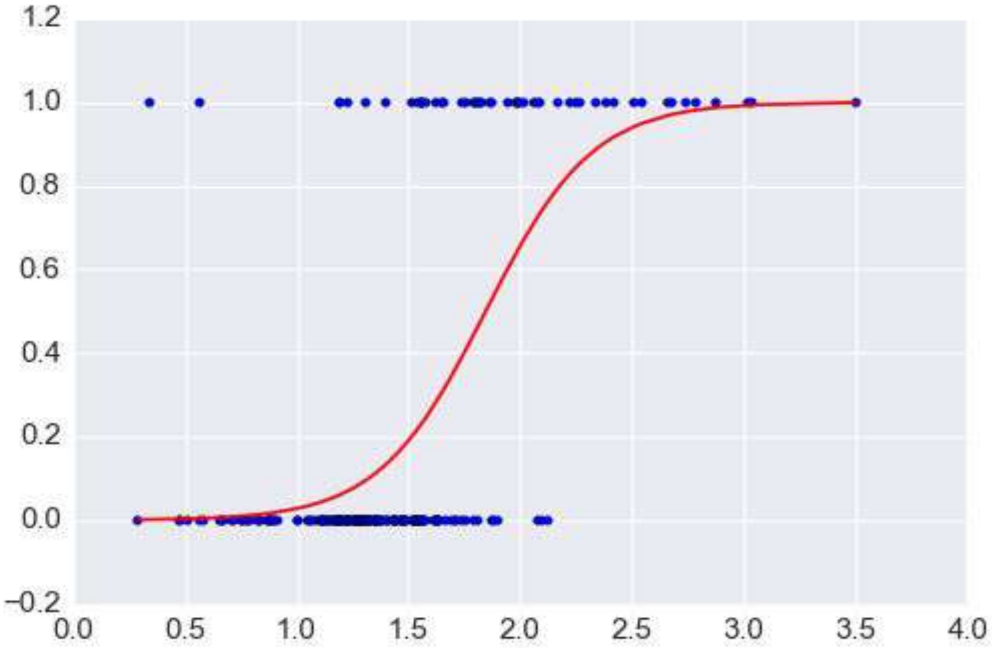
Logistic Regression: Putting it all together

Linear Regression



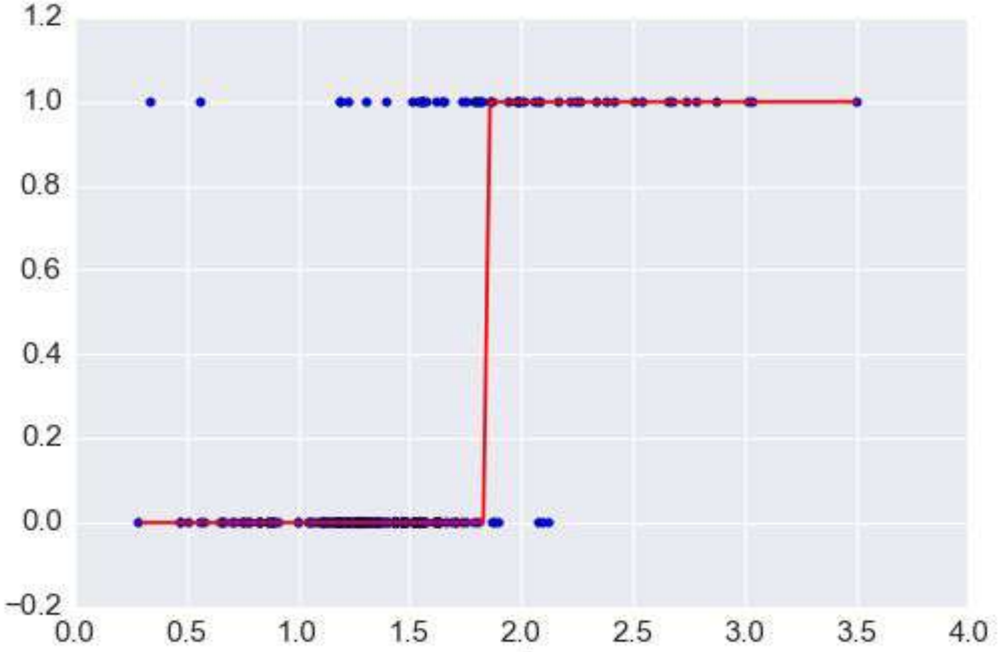
Logistic Regression: Putting it all together

Linear Regression + Sigmoid



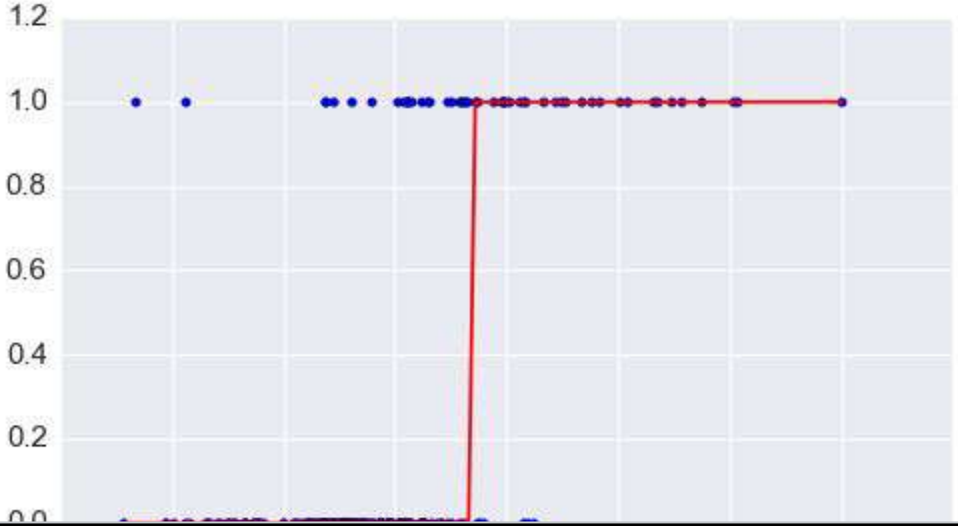
Logistic Regression: Putting it all together

Linear Regression + Sigmoid + Decision Boundary



Logistic Regression: Putting it all together

Linear Regression + Sigmoid + Decision Boundary



If you want to explore this example further I found an iPython notebook: bit.ly/CS249-F20-LogitReg

Classification

Classification is a method of supervised learning which uses labeled data $(\vec{x}, \vec{y})$ to learn a **parameterized** model:

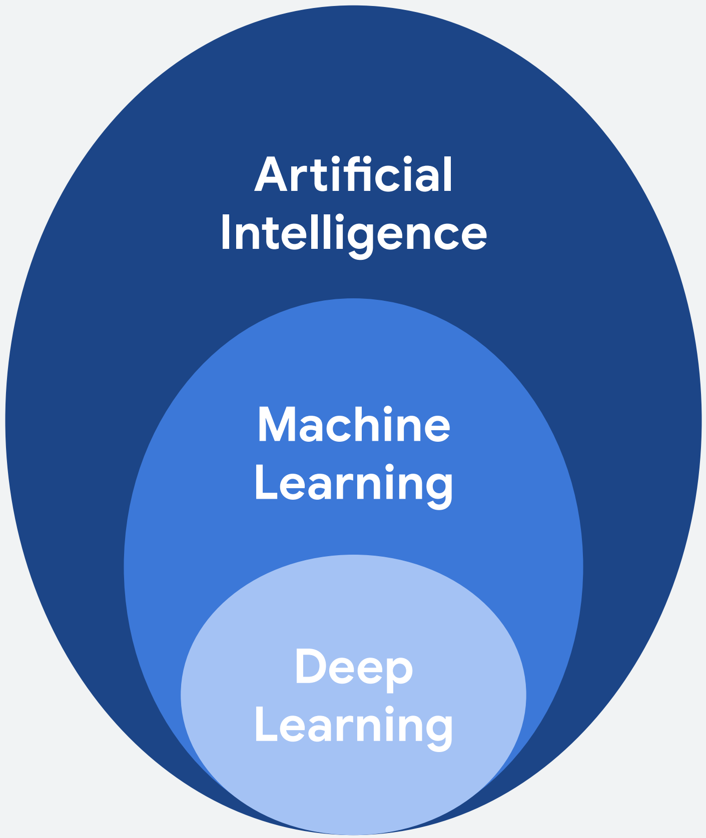
$$f_\theta(x) \rightarrow \hat{y}$$

where the **output is a discrete class (integer)**

- We can still optimize via **gradient descent** we just need new **loss functions**
- We now need a **nonlinear activation function** to help map us into **probability space**
- We then need a **decision boundary**

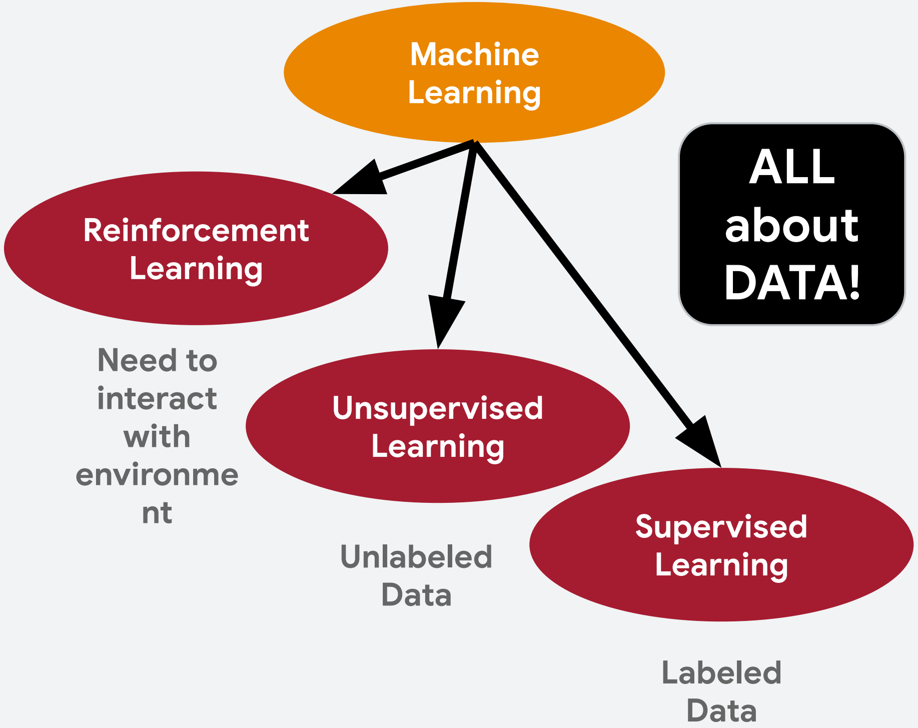
Quick Summary:

- AI > ML > Deep Learning



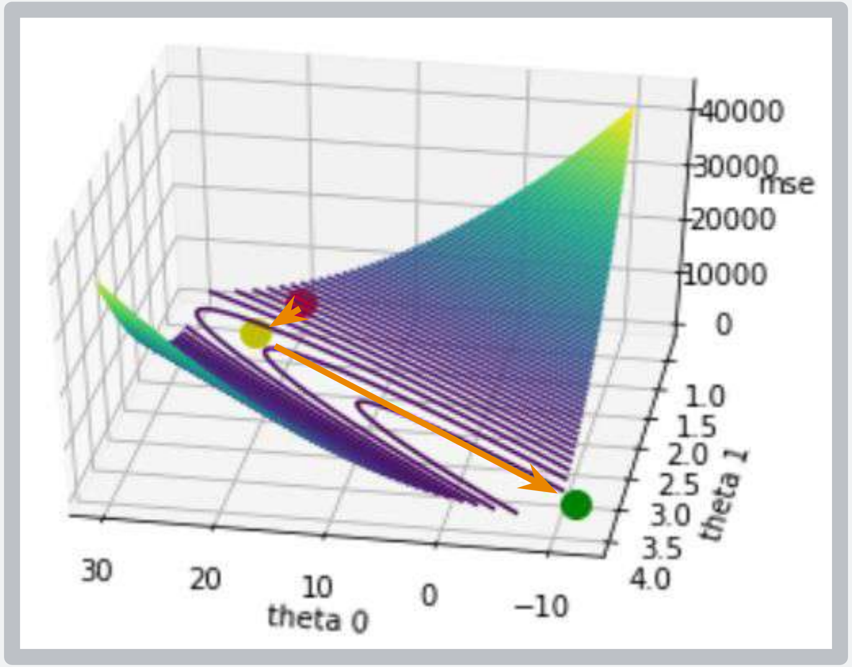
Quick Summary:

- AI > ML > Deep Learning
- Different methods of learning are defined by their **data**
- We will focus on **(Deep) Supervised Learning** in CS249r



Quick Summary:

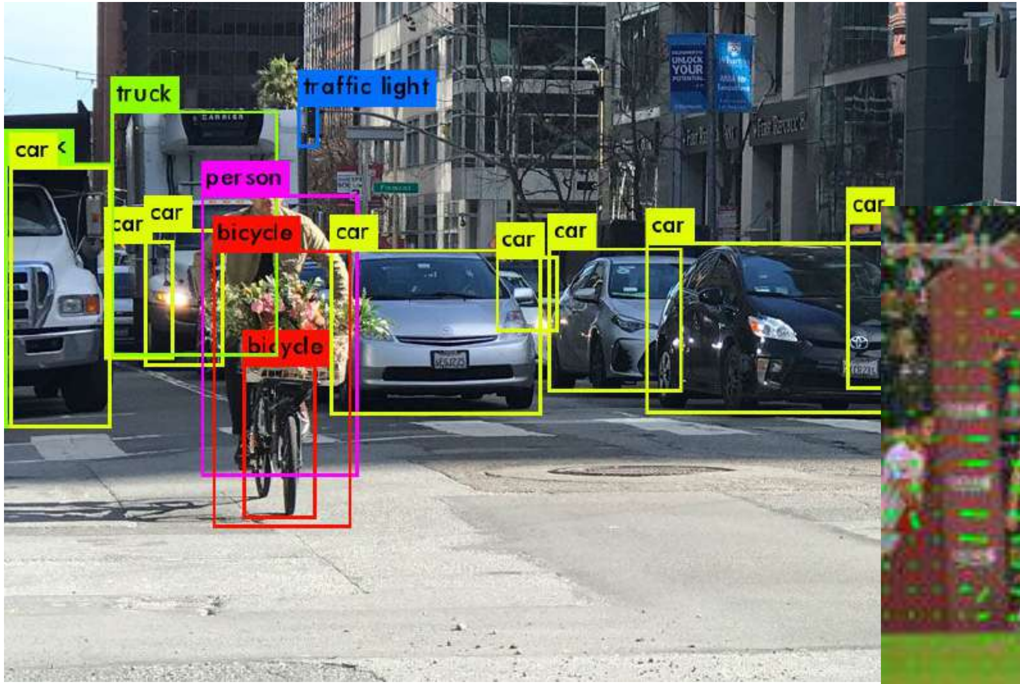
- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a **model with parameters**, **loss and activation functions**, **hyperparameters** (learning rate) and **gradient descent**
- Consider **overfitting** and **regularization/test data**



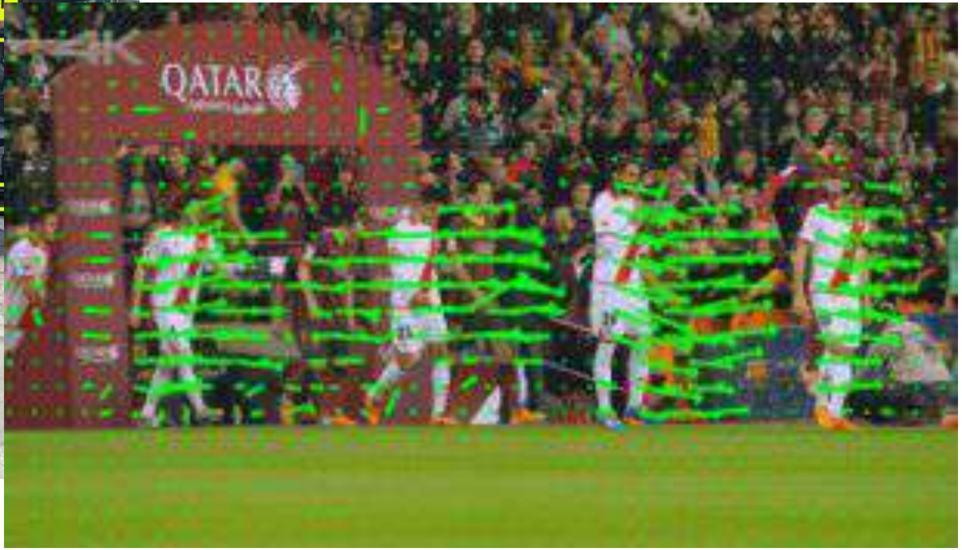
Classical Machine Learning:
Computer Vision

Computer Vision is all about Regression and Classification

Object Detection



Optical Flow



https://developer.nvidia.com/blog/an-introduction-to-the-nvidia-optical-flow-sdk/
https://medium.com/@rishi30.mehta/object-detection-with-yolo-giving-eyes-to-ai-7a3076c6977e

Computer Vision is **HARD!**

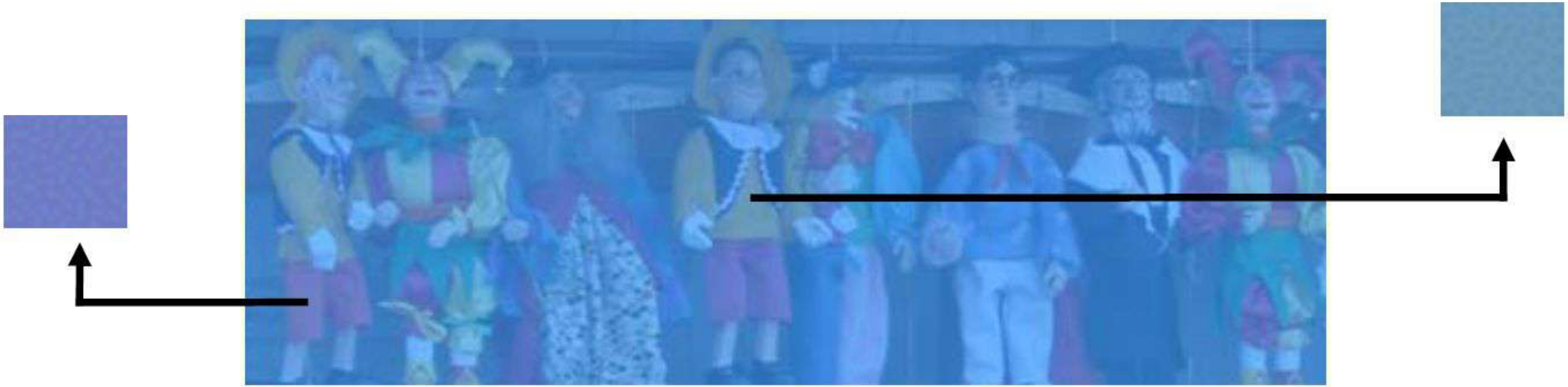
Computer Vision is **HARD!**

What color is the shirt? the pants?

Slide Credit: Hamilton Chong

Computer Vision is **HARD!**

What color is the shirt? the pants?



Slide Credit: Hamilton Chong

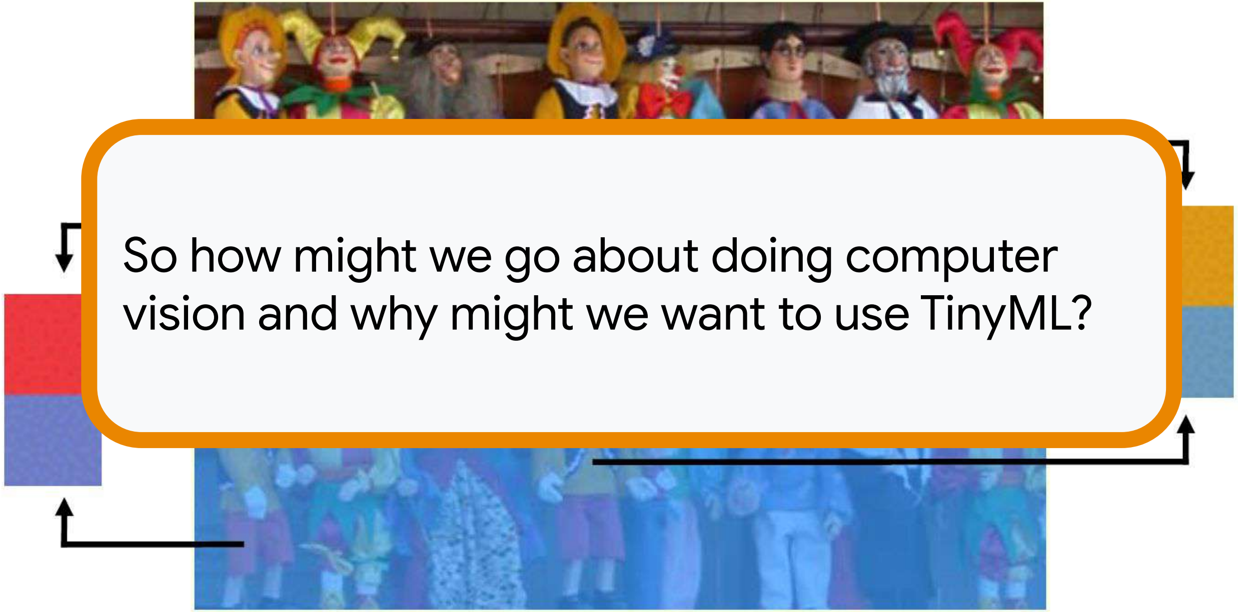
Computer Vision is **HARD!**

What color is the shirt? the pants?

Slide Credit: Hamilton Chong

Computer Vision is **HARD!**

What color is the shirt? the pants?



So how might we go about doing computer vision and why might we want to use TinyML?

Slide Credit: Hamilton Chong

Motivating Example: Stopping Illegal Fishing

- Satellites can track vessels that aren’t broadcasting transponders

Motivating Example: Stopping Illegal Fishing

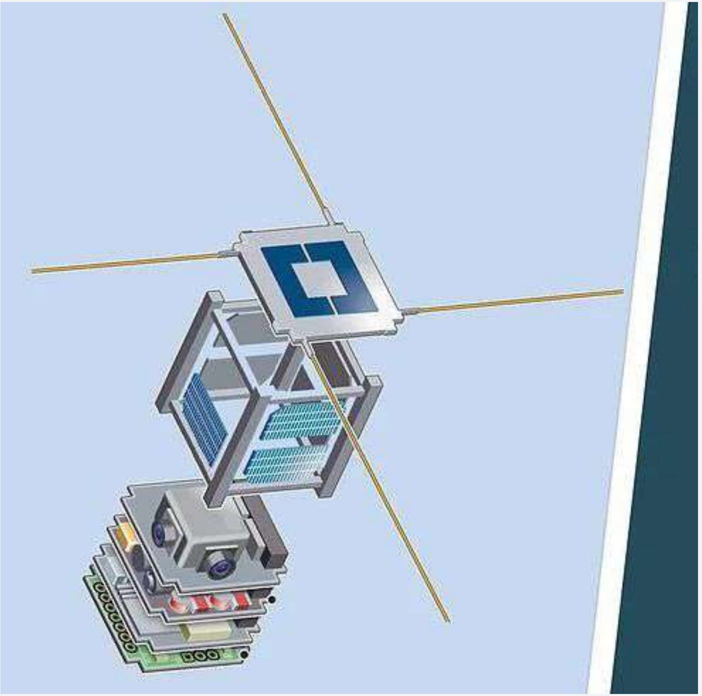
- Satellites can track vessels that aren’t broadcasting transponders
- We have two options

Motivating Example: Stopping Illegal Fishing

- Satellites can track vessels that aren't broadcasting transponders
- We have two options
    a. Launch a few expensive satellites that do a good job covering small areas

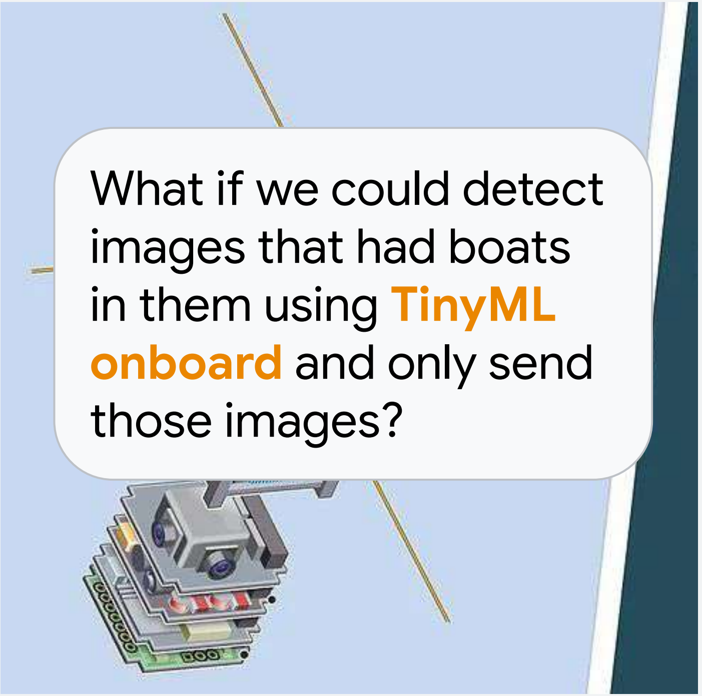
Motivating Example: Stopping Illegal Fishing

- Satellites can track vessels that aren't broadcasting transponders
- We have two options
  a. Launch a few expensive satellites that do a good job covering small areas
  b. Launch many cheap satellites that don't do a good job covering large areas



Motivating Example: Stopping Illegal Fishing

- Satellites can track vessels that aren't broadcasting transponders
- We have two options
  a. Launch a few expensive satellites that do a good job covering small areas
  b. Launch many cheap satellites that don't do a good job covering large areas



Motivating Example: Stopping Illegal Fishing

How can we detect that a boat is in an image?

https://arstechnica.com/tech-policy/2017/02/to-catch-a-thief-with-satellite-data/

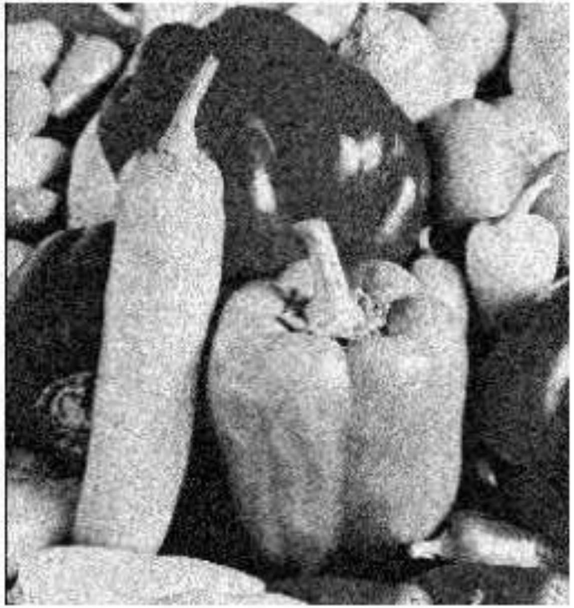
Motivating Example:
Stopping Illegal Fishing

https://arstechnica.com/tech-policy/2017/02/to-catch-a-thief-with-satellite-data/

How can we detect that a boat is in an image?

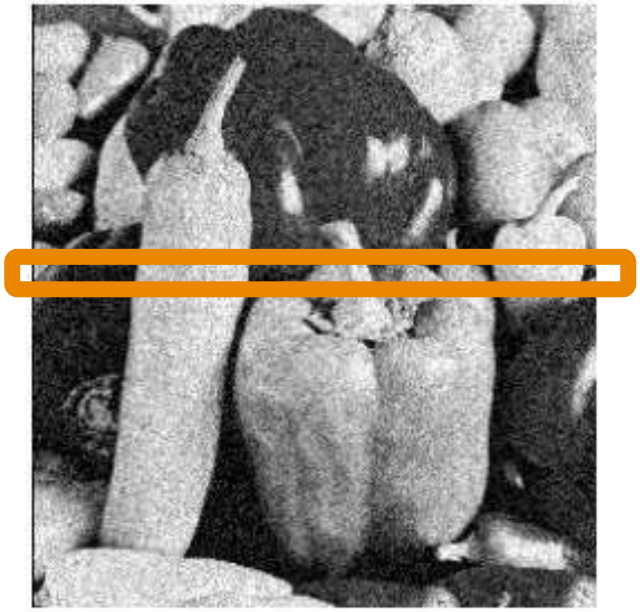
Look for an **EDGE**!

Edge Detection



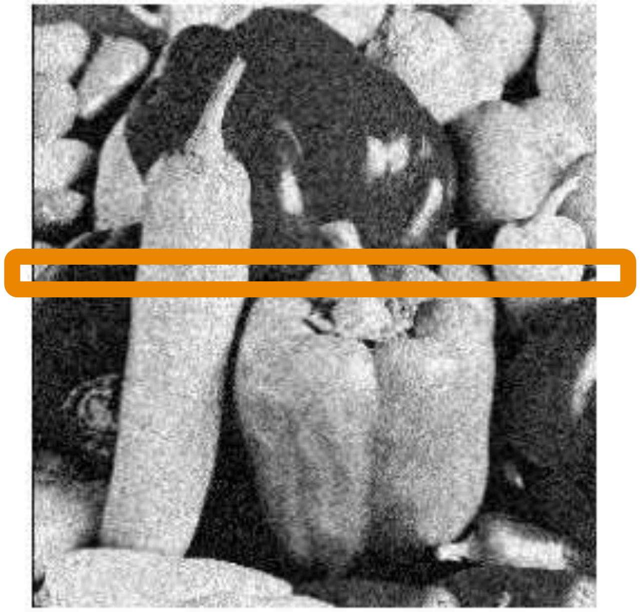
Slide Credit: Todd Zickler CS 283

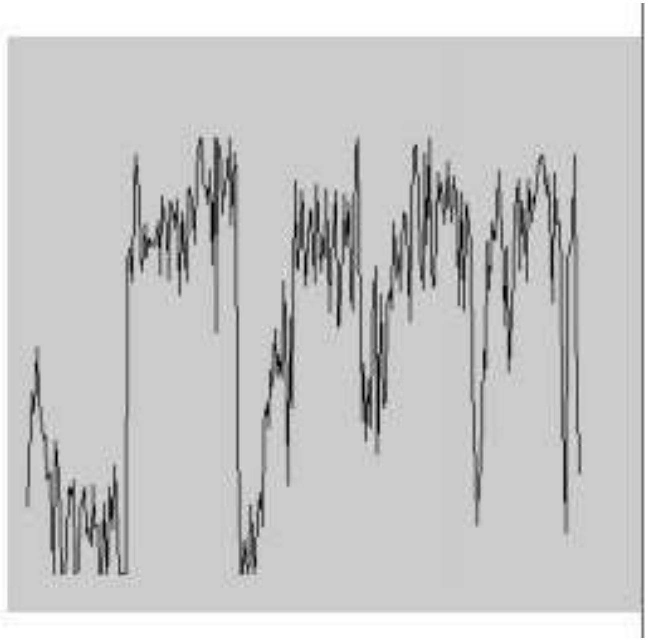
Edge Detection



Slide Credit: Todd Zickler CS 283

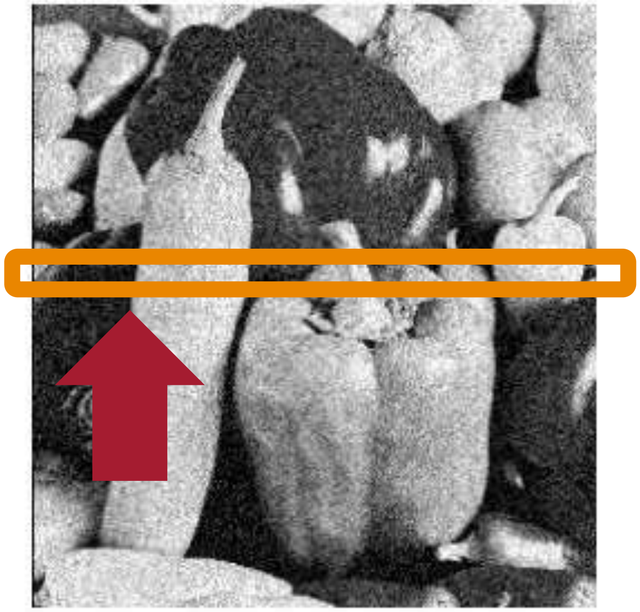
Edge Detection

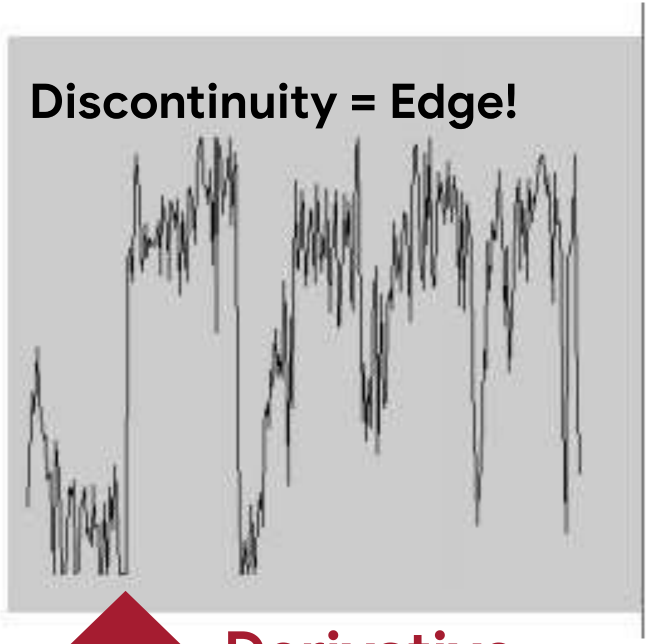




Slide Credit: Todd Zickler CS 283

Edge Detection



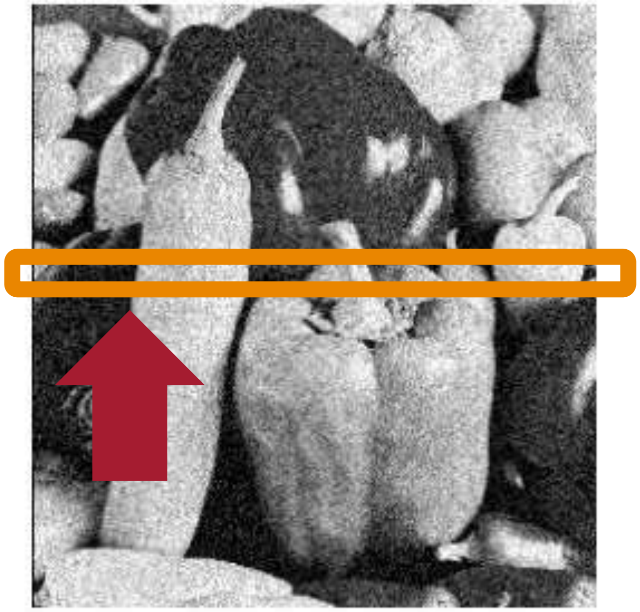


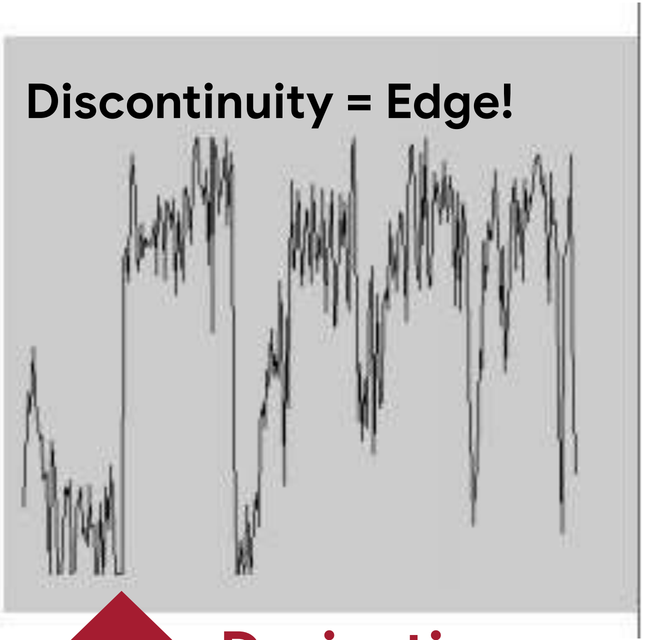
Derivative is large

Slide Credit: Todd Zickler CS 283

Edge Detection

But this data is very noisy





Derivative is large

Slide Credit: Todd Zickler CS 283

Spatial Local Averaging
Reduces Noise!

Remember
Mean Shift?
We can do
something
similar in spirit
here!

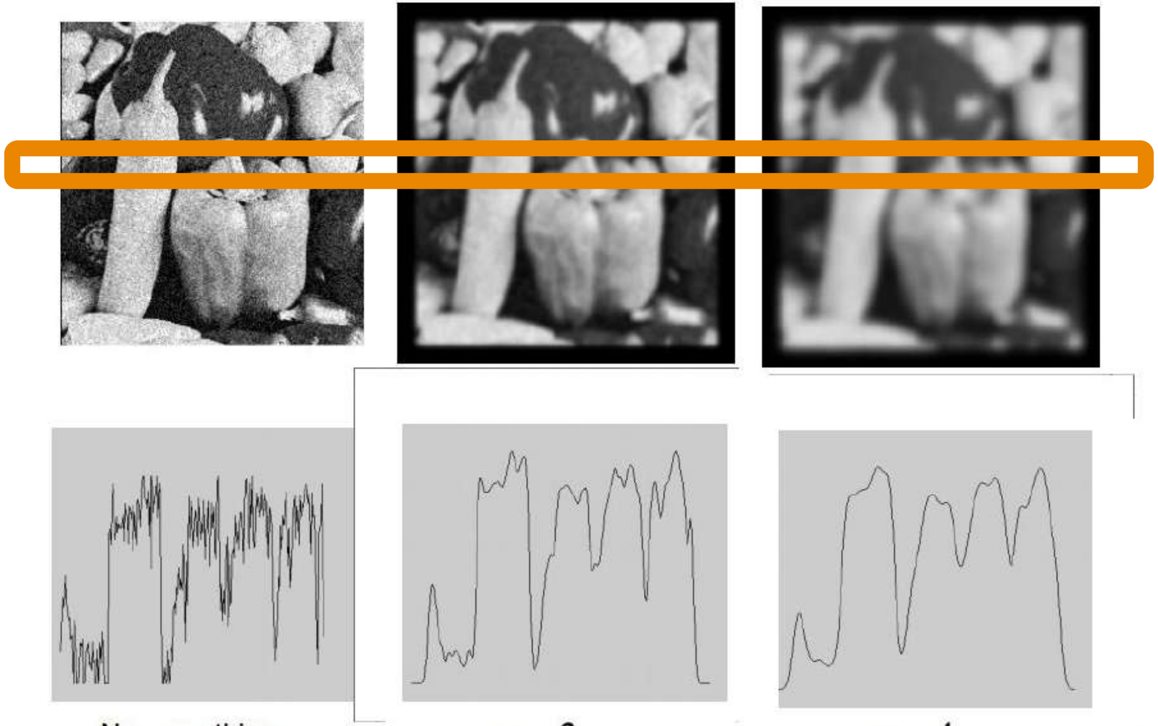
Slide Credit: Todd Zickler CS 283

Spatial Local Averaging
Reduces Noise!

Remember **Mean Shift**? We can do something similar in spirit here!

Slide Credit: Todd Zickler CS 283

Spatial Local Averaging
Reduces Noise!

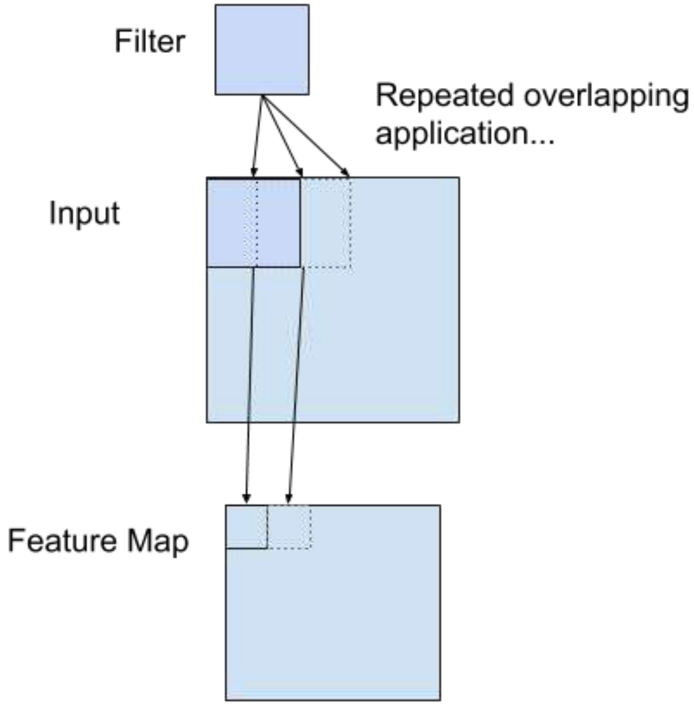


No smoothing $\sigma = 2$ $\sigma = 4$

Remember **Mean Shift**? We can do something similar in spirit here!

Slide Credit: Todd Zickler CS 283

Traditional CV does this through **Convolutions of Linear Filters**

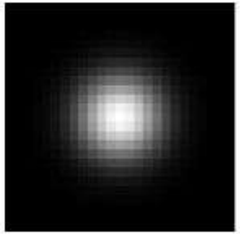


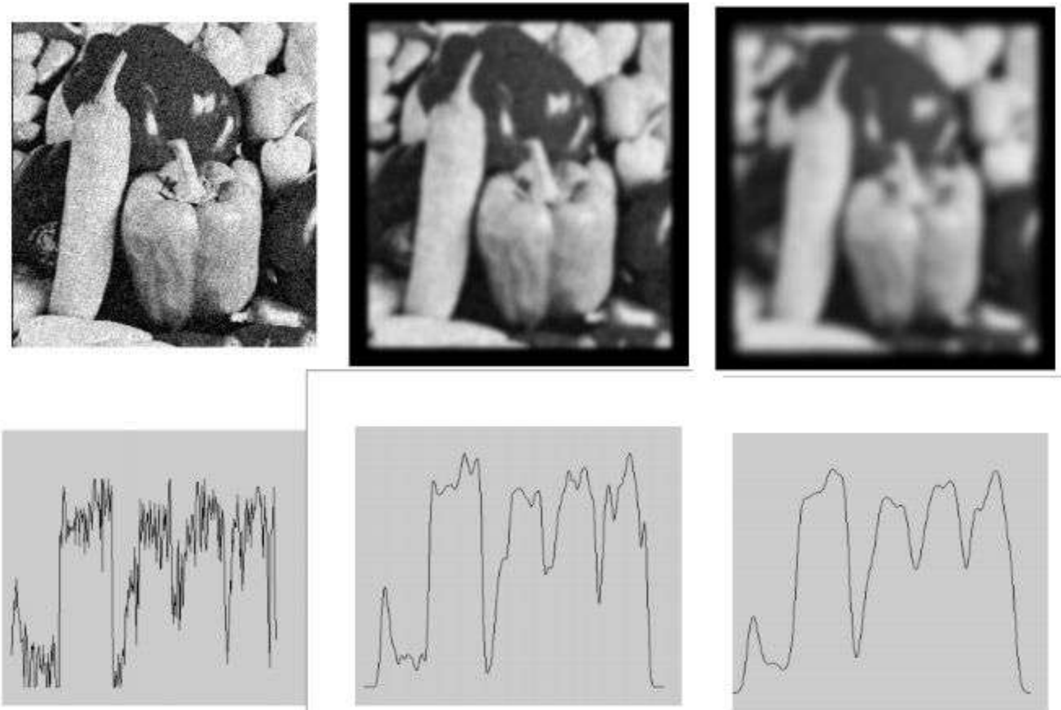
Slide Credit: Todd Zickler CS 283

Traditional CV does this through **Convolutions of Linear Filters**

0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, $\sigma = 1$



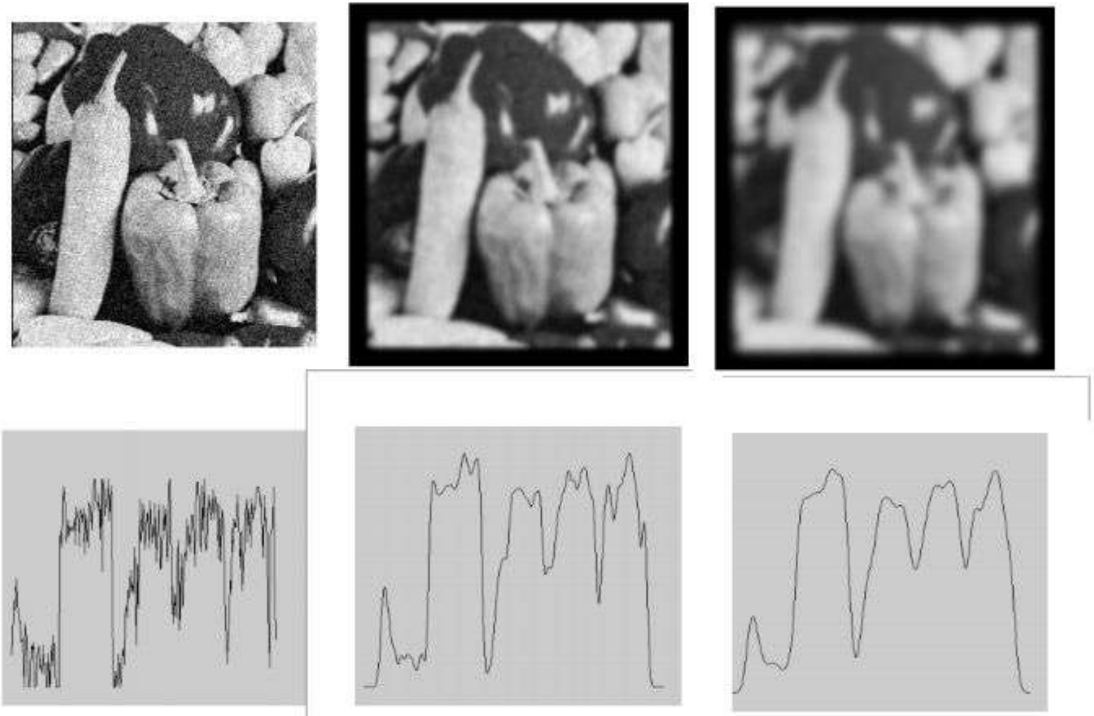


No smoothing $\sigma = 2$ $\sigma = 4$

Slide Credit: Todd Zickler CS 283

Traditional CV does this through
Convolutions of Linear Filters

Smoothing is a **pre-processing** stage that is critical for finding good edges!



No smoothing | $\sigma = 2$ | $\sigma = 4$

Slide Credit: Todd Zickler CS 283

Motivating Example:
Stopping Illegal Fishing

https://arstechnica.com/tech-policy/2017/02/to-catch-a-thief-with-satellite-data/

How can we detect that a boat is in an image?

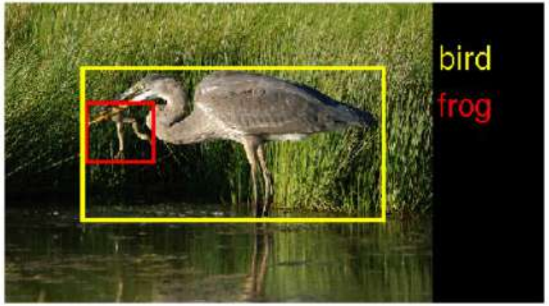
Look for an **EDGE**!

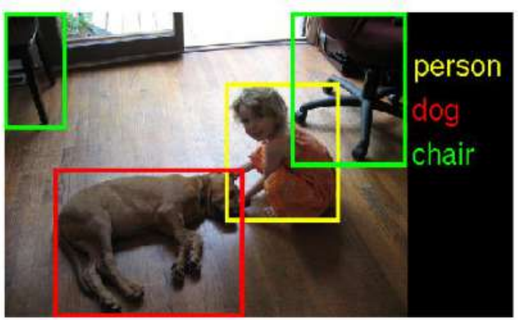
Motivating Example:
Stopping Illegal Fishing

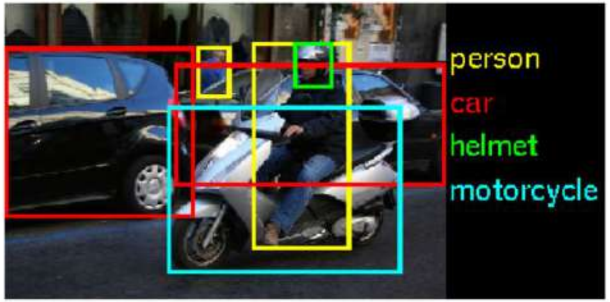
We need to tell the objects apart!

https://arstechnica.com/tech-policy/2017/02/to-catch-a-thief-with-satellite-data/

But what features should we use for **Object Recognition**?







ImageNet Challenge: 1.2 million **labeled** items

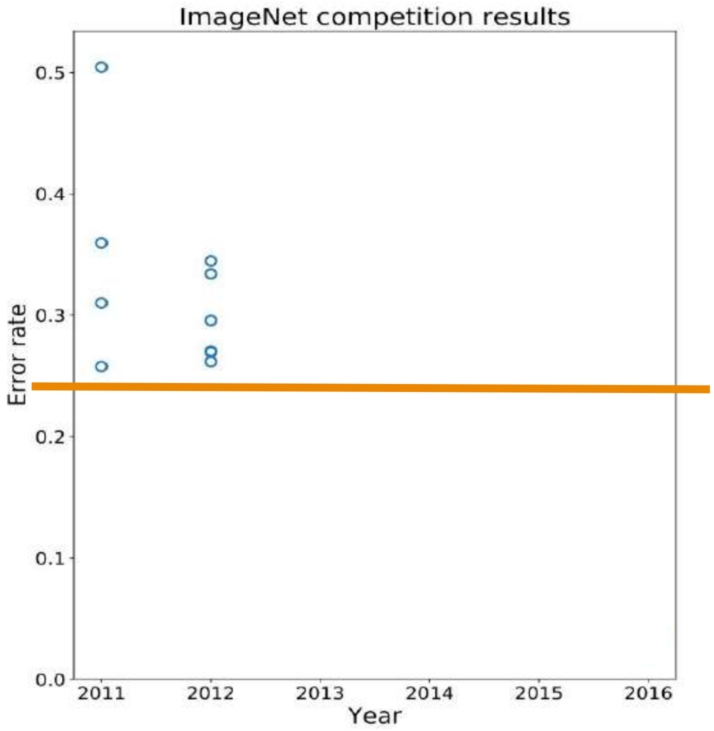
Slide Credit: ImageNet

Deep vs Classical Learning

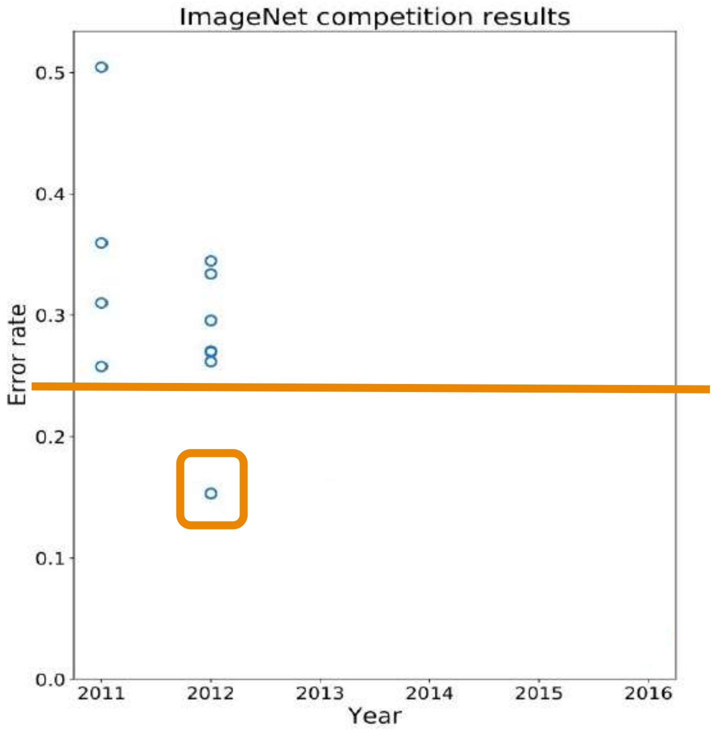
Traditional Machine Learning Flow

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

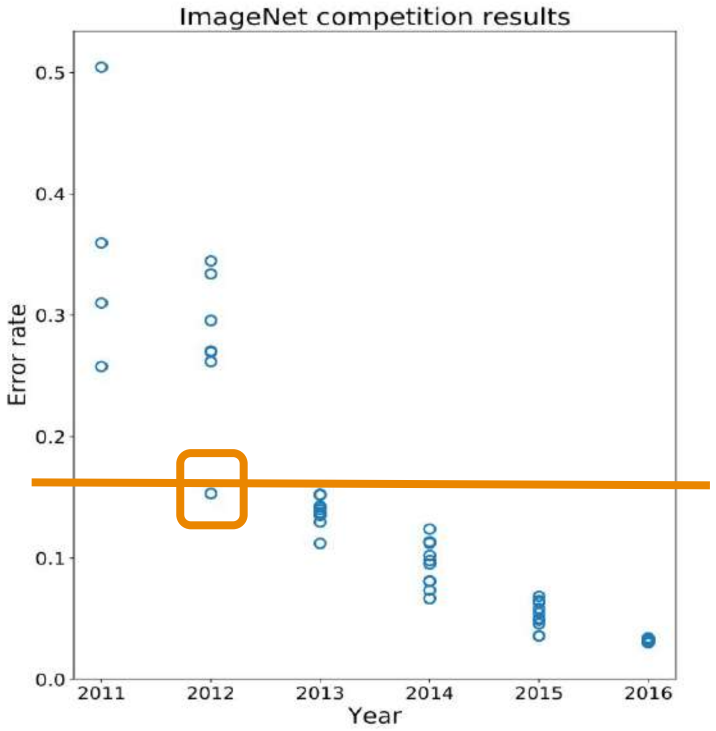
The ImageNet Challenge



The ImageNet Challenge



The ImageNet Challenge



What happened in 2012?

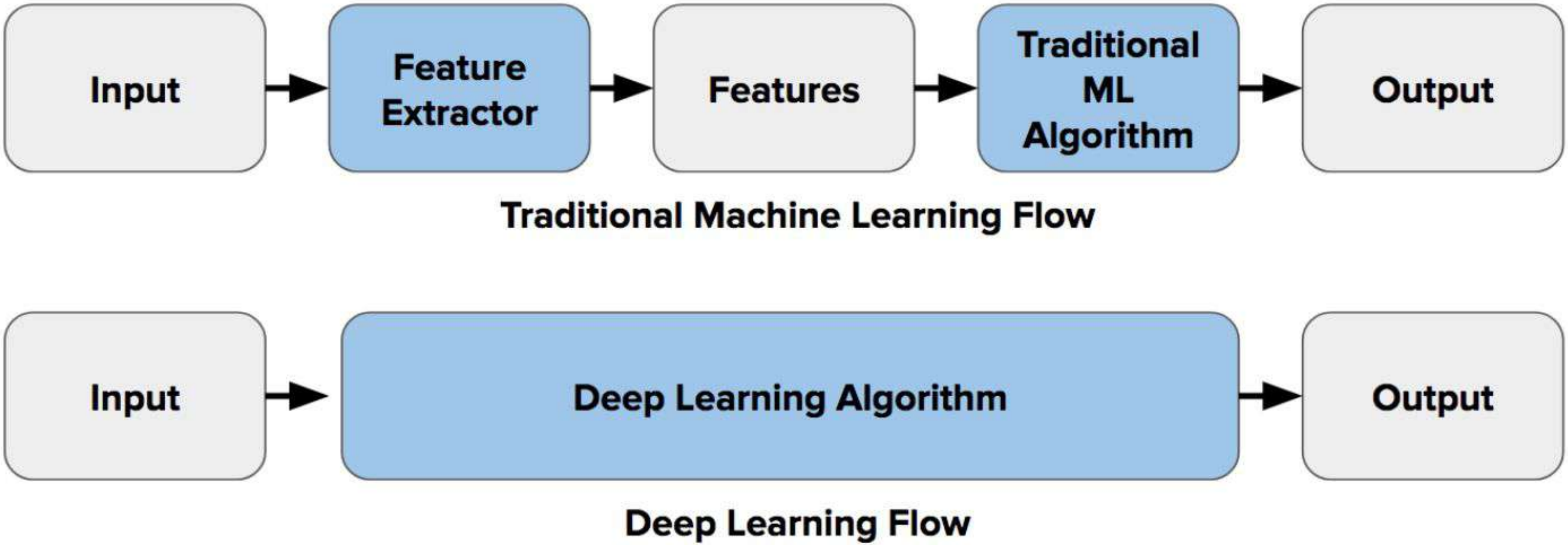
The Rise of Deep Learning:
AlexNet

Deep vs Classical Learning

Traditional Machine Learning Flow

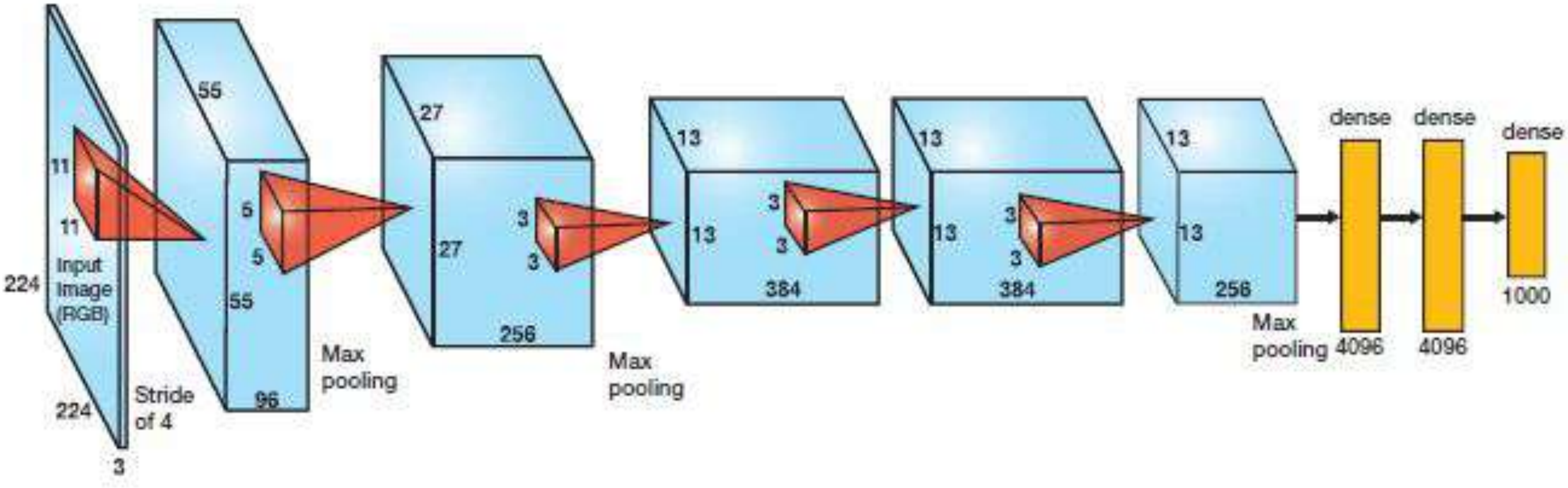
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

Deep vs Classical Learning



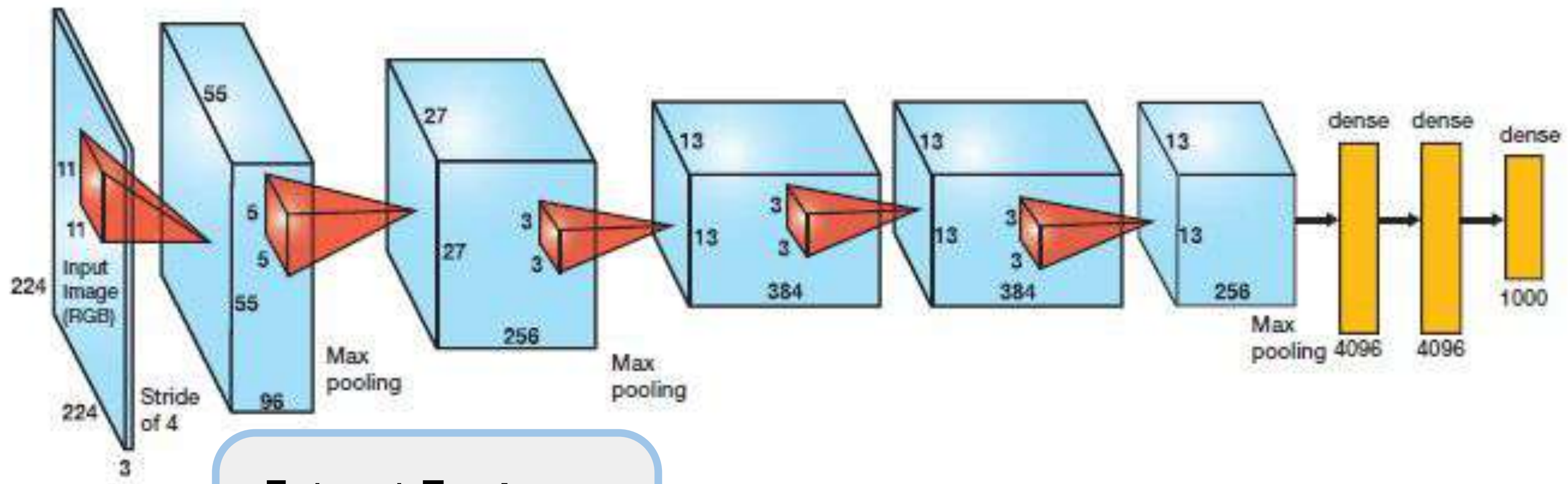
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet

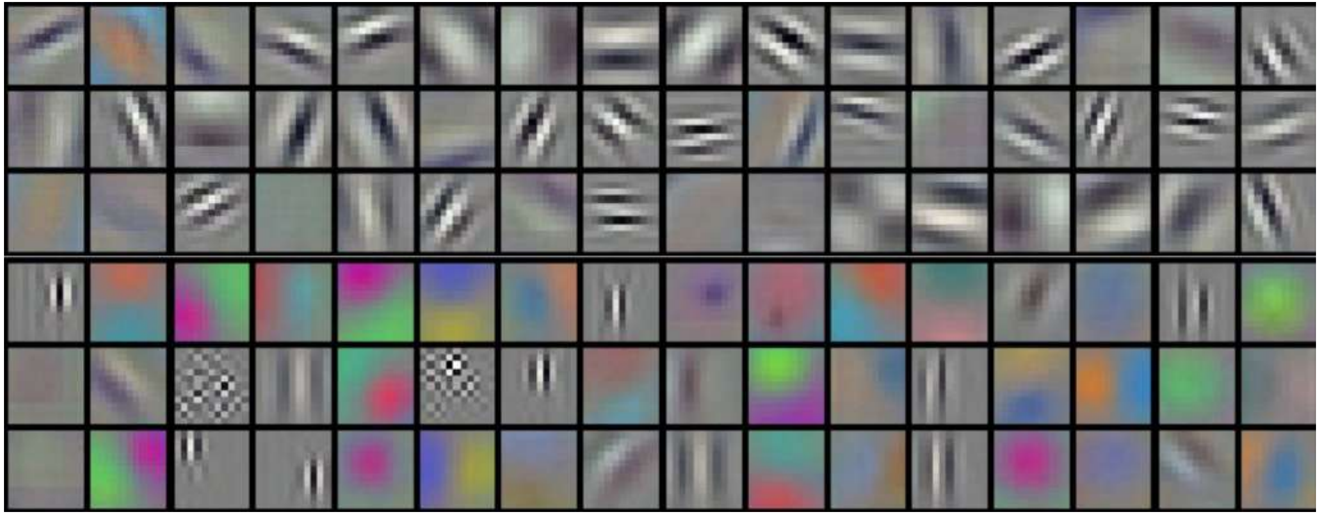


Extract **Features** from data with **Convolutions**

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet

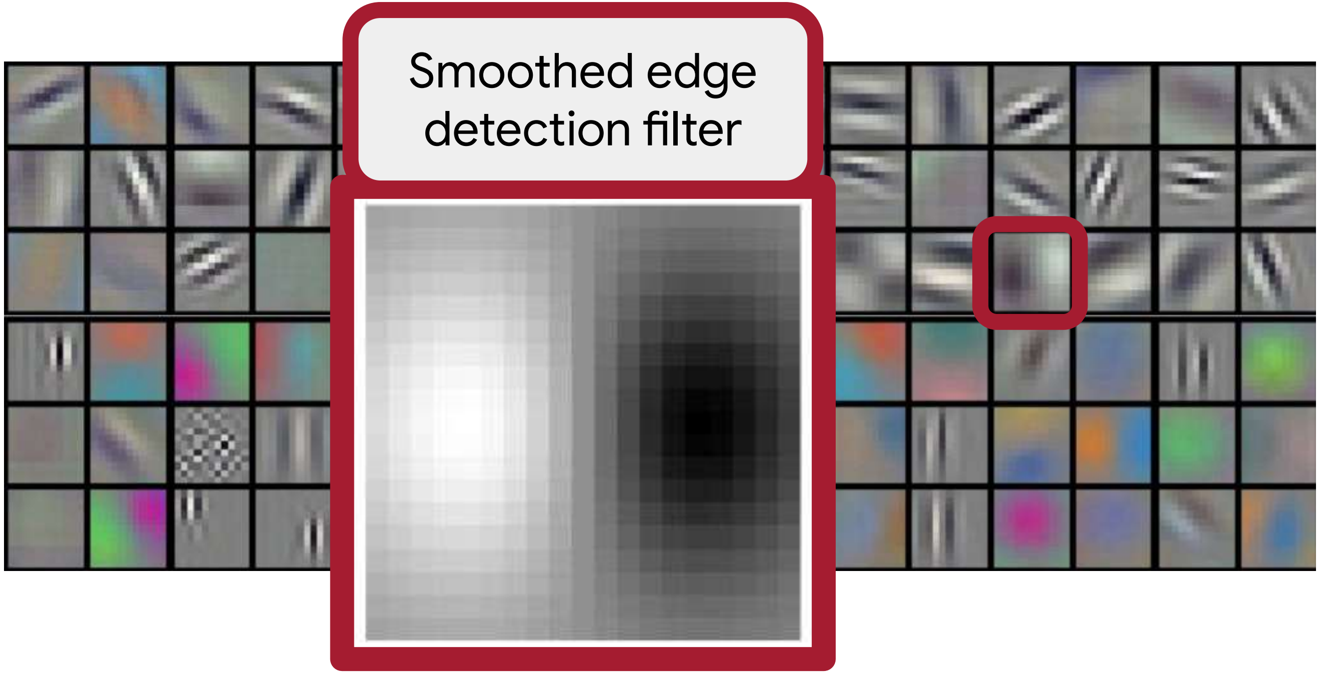
The filters learned by the NN for the initial convolution layer look like standard CV filters!



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

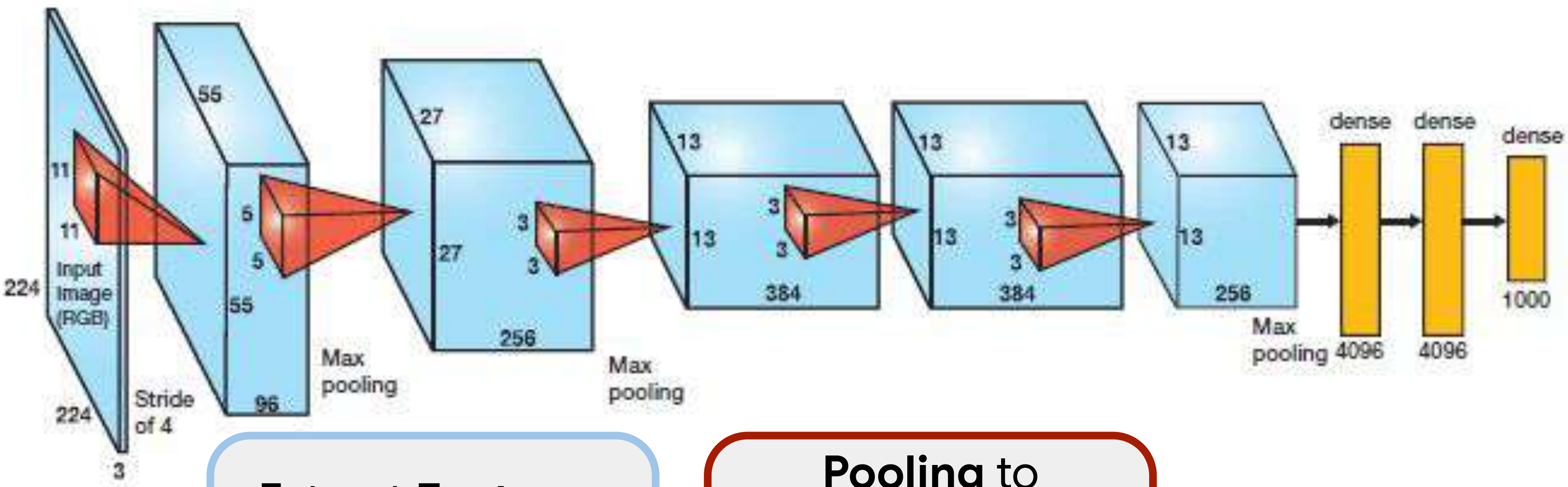
AlexNet

The filters learned by the NN for the initial convolution layer look like standard CV filters!



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet

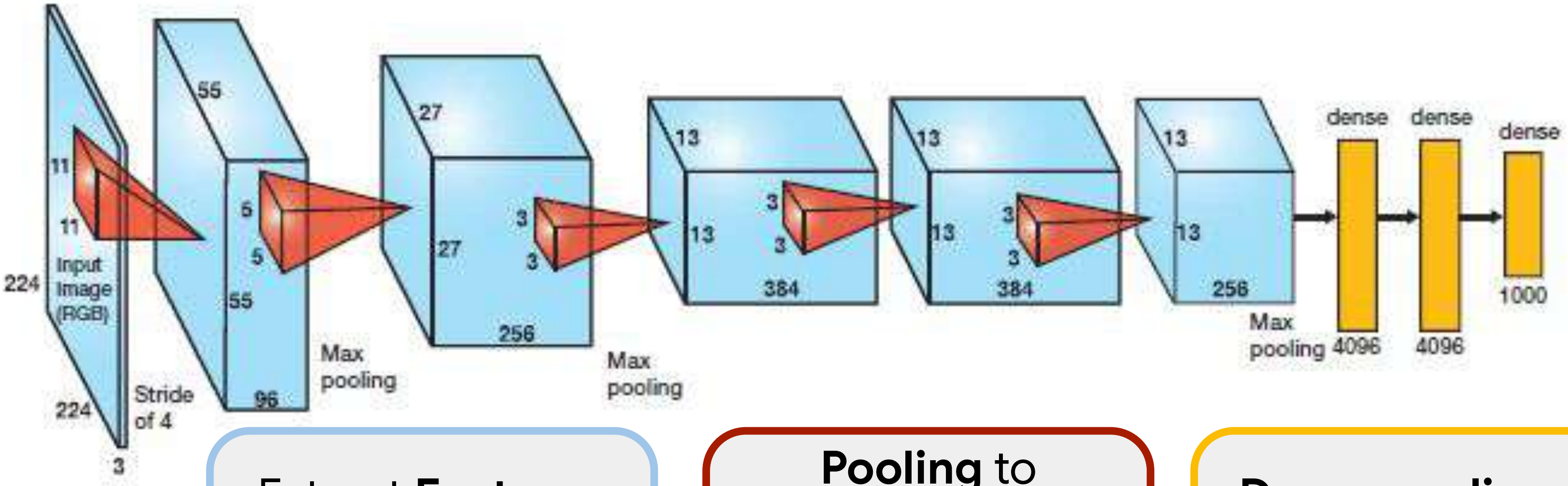


Extract **Features** from data with **Convolutions**

Pooling to **summarize** for higher level features

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet



Extract **Features** from data with **Convolutions**

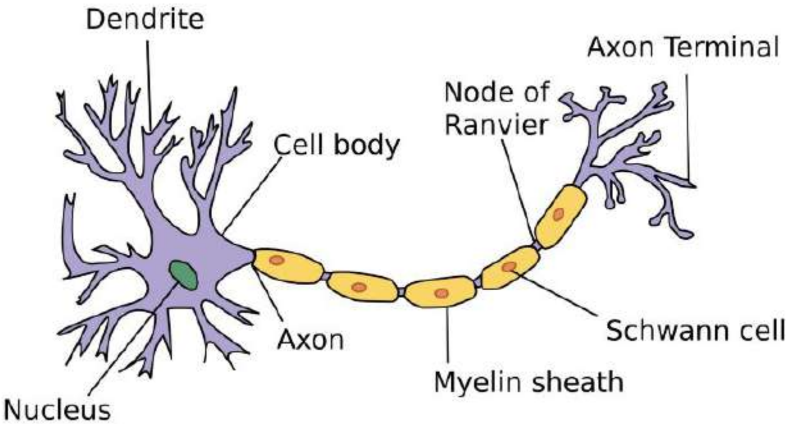
Pooling to **summarize** for higher level features

Dense nonlinear layers to **classify** the final features

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

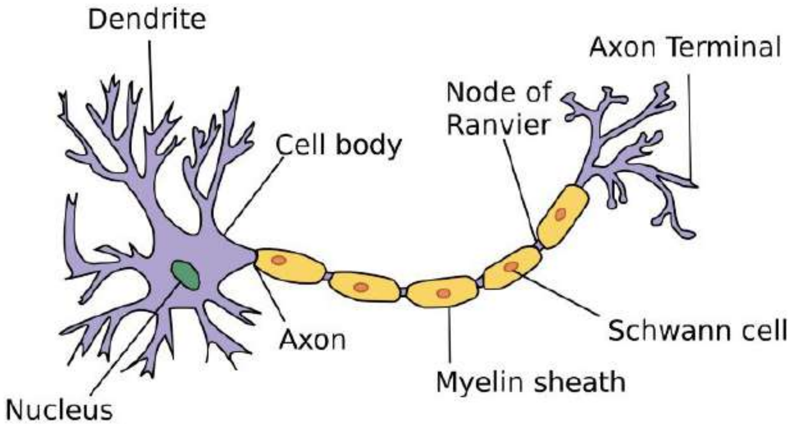
Supervised (tiny) Deep Learning 101

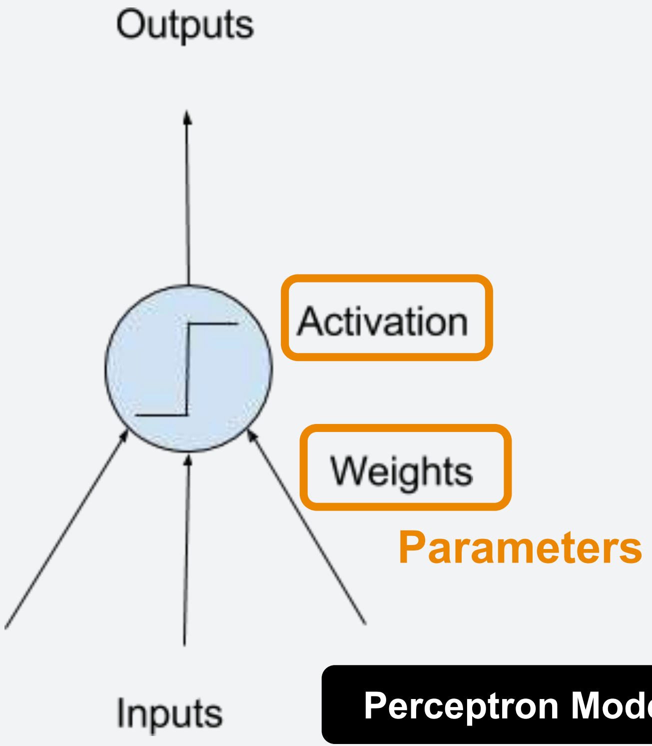
Neural Networks a model of the brain!



https://towardsdatascience.com/perceptron-explanation-implementation-and-a-visual-example-3c8e76b4e2d1
https://machinelearningmastery.com/neural-networks-crash-course/

Neural Networks a model of the brain!

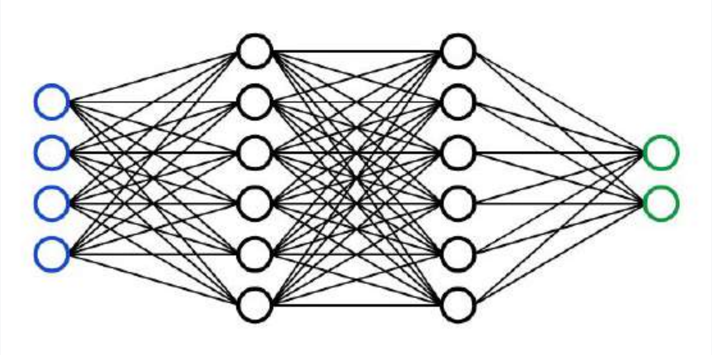




Perceptron Model

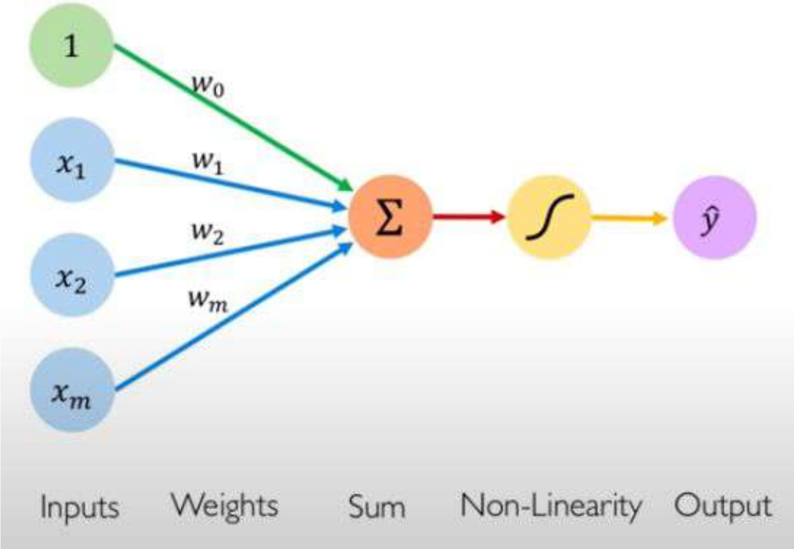
https://towardsdatascience.com/perceptron-explanation-implementation-and-a-visual-example-3c8e76b4e2d1
https://machinelearningmastery.com/neural-networks-crash-course/

Neural Networks a model of the brain!

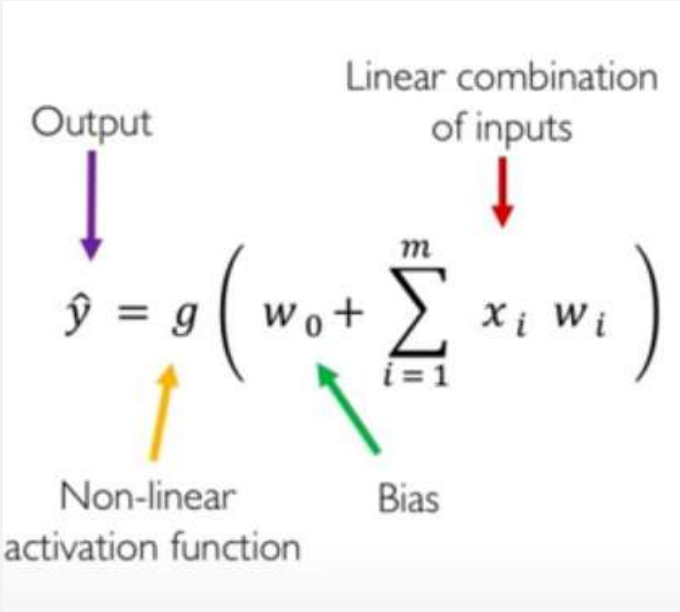


https://towardsdatascience.com/perceptron-explanation-implementation-and-a-visual-example-3c8e76b4e2d1
https://machinelearningmastery.com/neural-networks-crash-course/

Neural Networks a model of the brain!

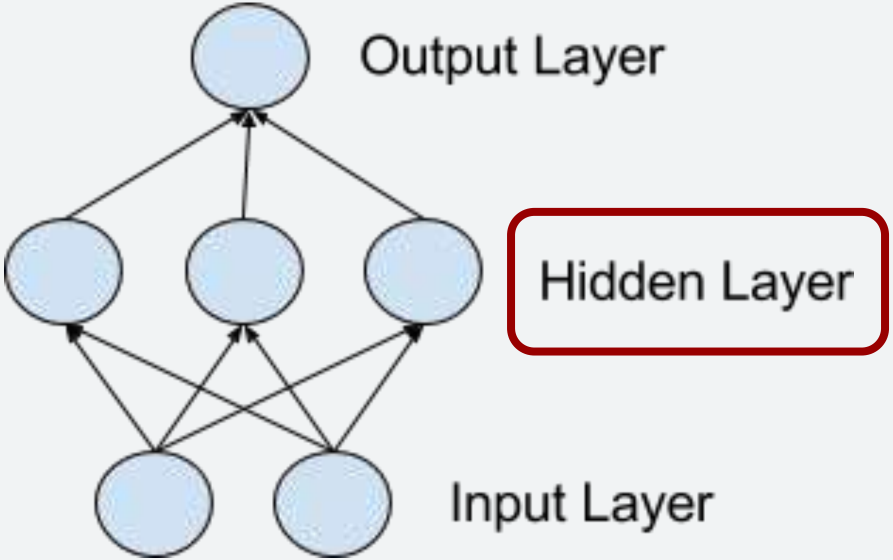


https://www.youtube.com/watch?v=njKP3FqW3Sk



This is just like logistic regression

Nonlinearities and Depth for Generality

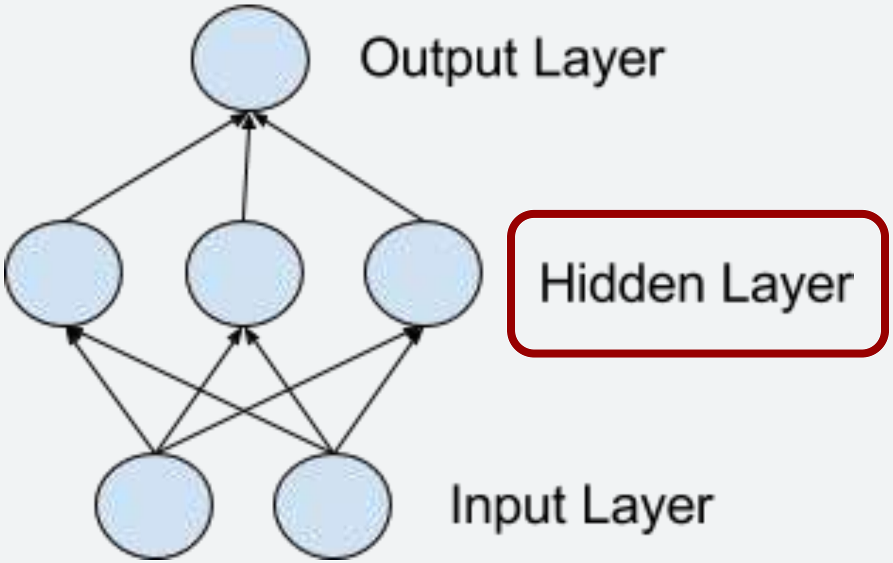


https://arxiv.org/pdf/1512.03965.pdf

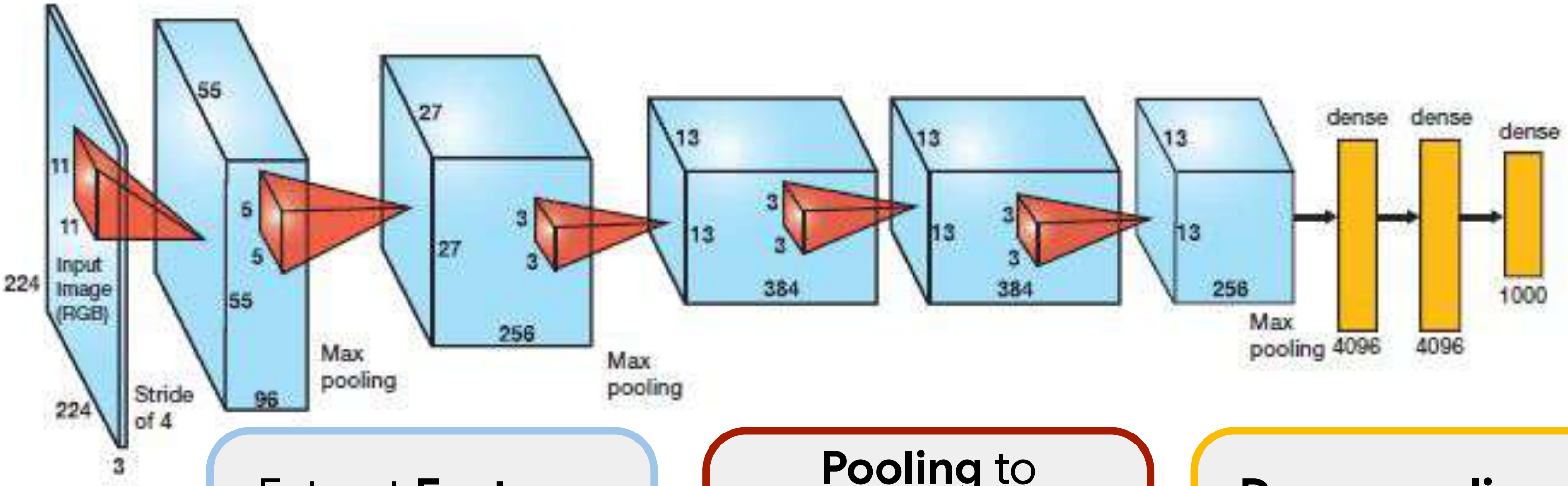
Nonlinearities and Depth for Generality

"It is well-known that sufficiently large depth-2 neural networks, **using reasonable activation functions**, **can approximate any continuous function** on a bounded domain"

https://arxiv.org/pdf/1512.03965.pdf



AlexNet



Extract **Features** from data with **Convolutions**

Pooling to **summarize** for higher level features

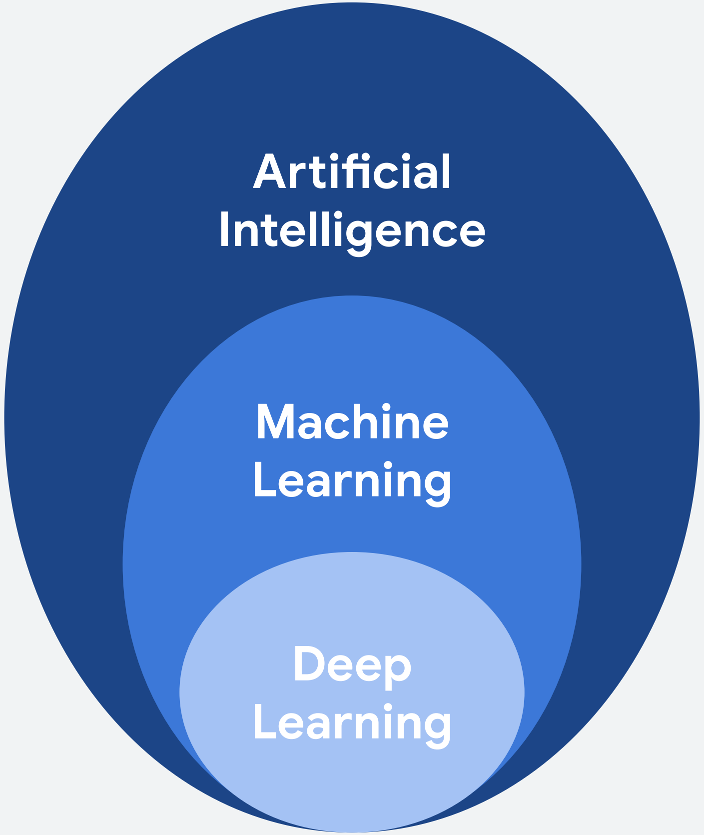
Dense nonlinear layers to **classify** the final features

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

Next Class:
(Tiny) Deep Learning

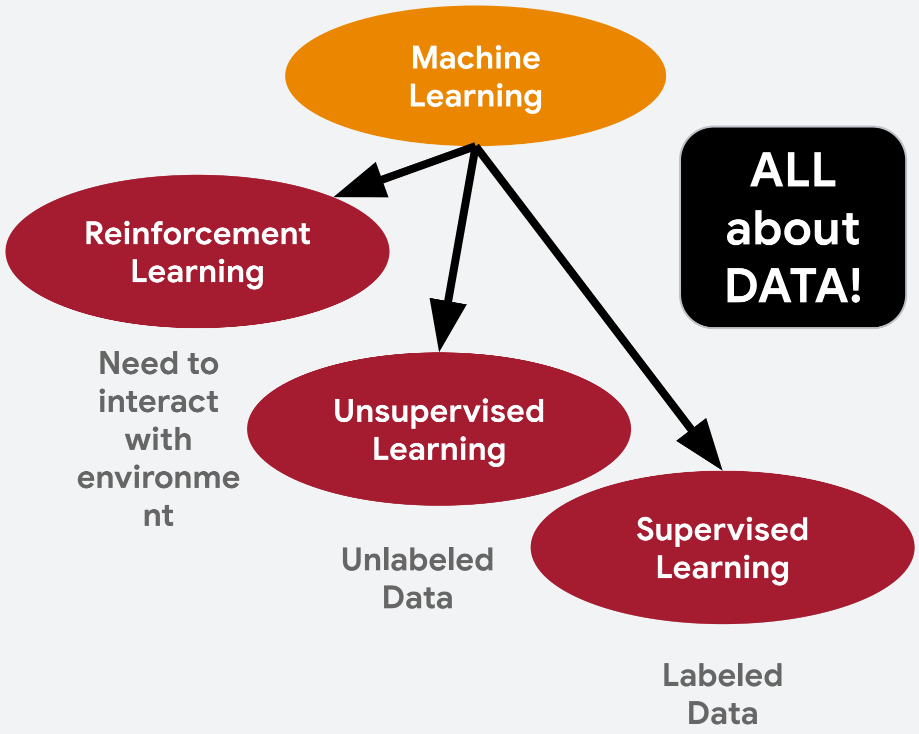
Quick Summary:

- AI > ML > Deep Learning



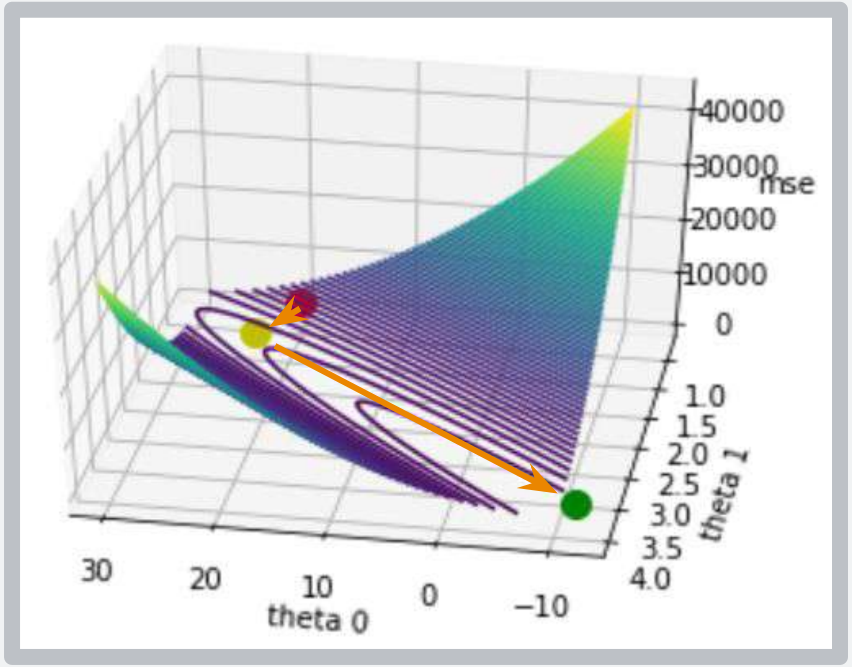
Quick Summary:

- AI > ML > Deep Learning
- Different methods of learning are defined by their **data**
- We will focus on **(Deep) Supervised Learning** in CS249r



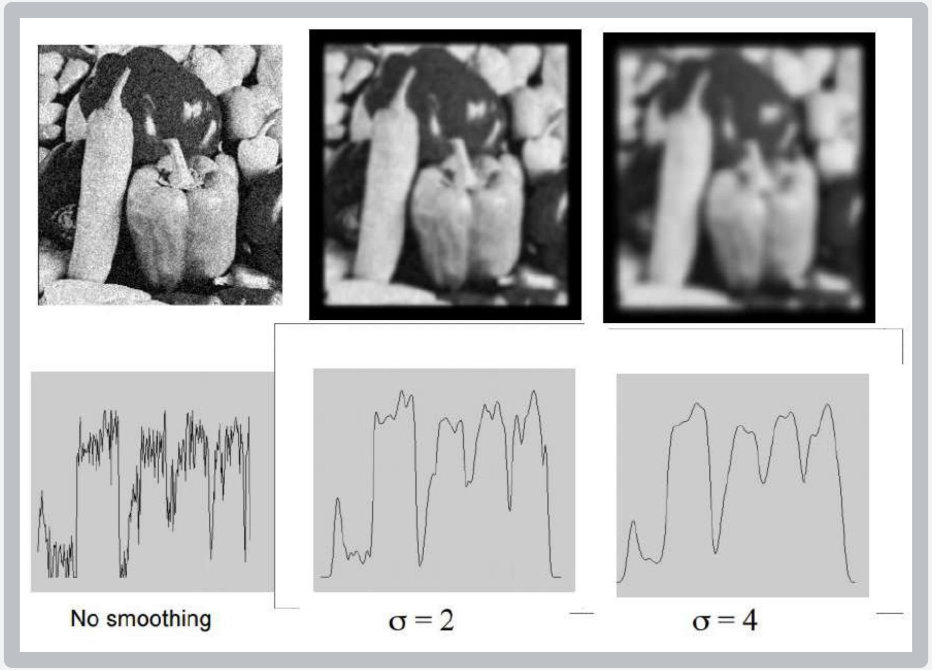
Quick Recap:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a **model with parameters**, **loss and activation functions**, **hyperparameters** (learning rate) and **gradient descent**
- Consider **overfitting** and **regularization/test data**



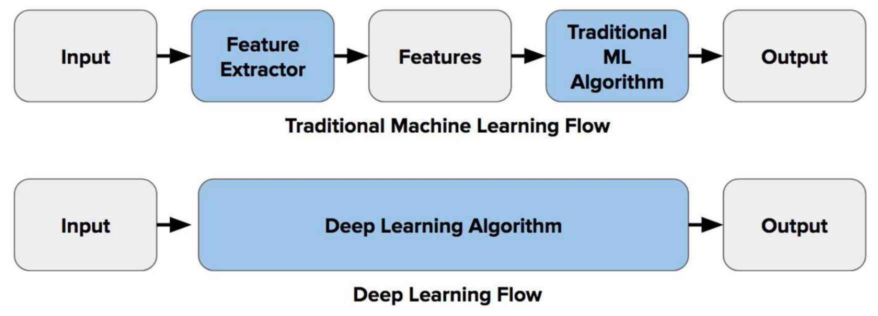
Quick Summary:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters (learning rate) and gradient descent
- Consider overfitting and regularization/test data
- Computer vision **pre-processes** data and finds **features** through **convolutions**



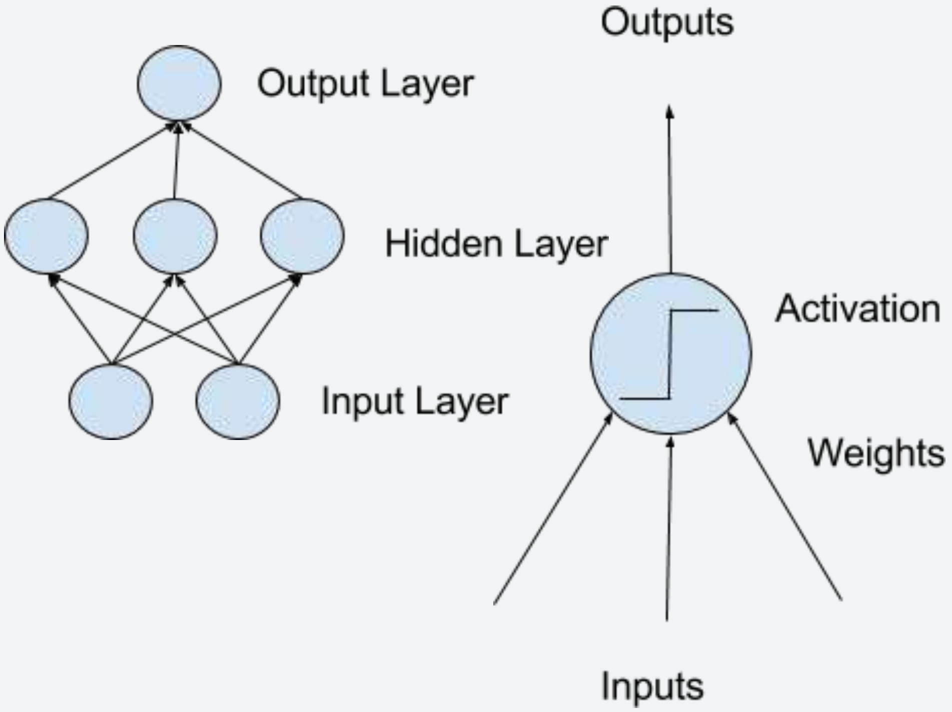
Quick Summary:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters (learning rate) and gradient descent
- Consider overfitting and regularization/test data
- Computer vision pre-processes data and finds features through convolutions
- **Deep Learning automates** the designs and interactions of features



Quick Summary:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters (learning rate) and gradient descent
- Consider overfitting and regularization/test data
- Computer vision pre-processes data and finds features through convolutions
- Deep Learning automates the designs and interactions of features by constructing **deep networks** of **nonlinearly activated**, connected neurons



Quick Summary:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters (learning rate) and gradient descent
- Consider overfitting and regularization
- Computer vision pre-processes data and finds features through convolutions
- Deep Learning automates the designs and interactions of features by constructing deep networks of nonlinearly activated, connected neurons
- Convolutions are a key way of finding spatial features in data

Next Class: (tiny) Deep Learning

Please fill out the feedback poll!

Please fill out the feedback poll!

place zoom
headshot
window here

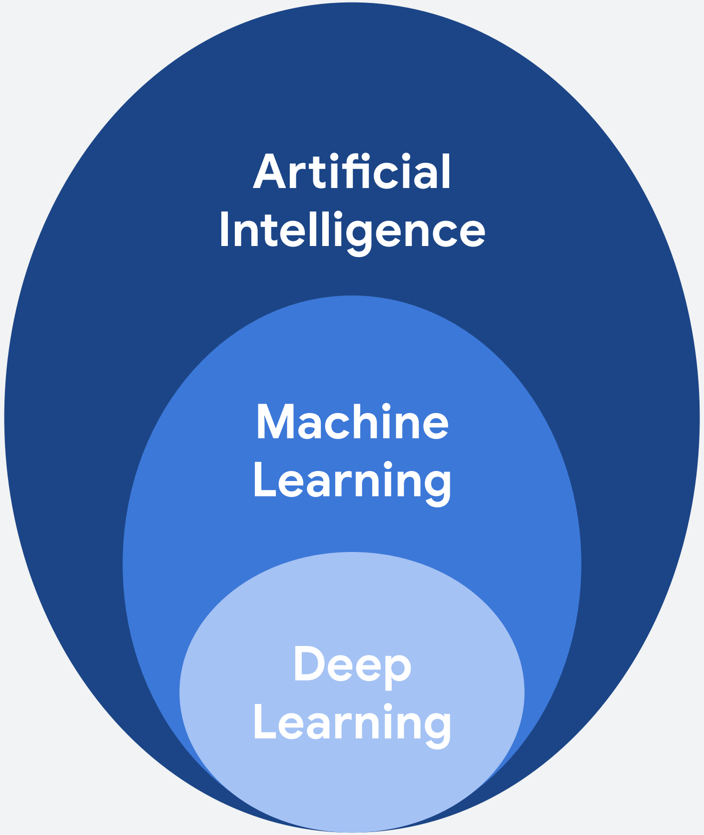
CS249r

Intro to (tiny) Machine Learning Part 2

Brian Plancher 9/14/2020

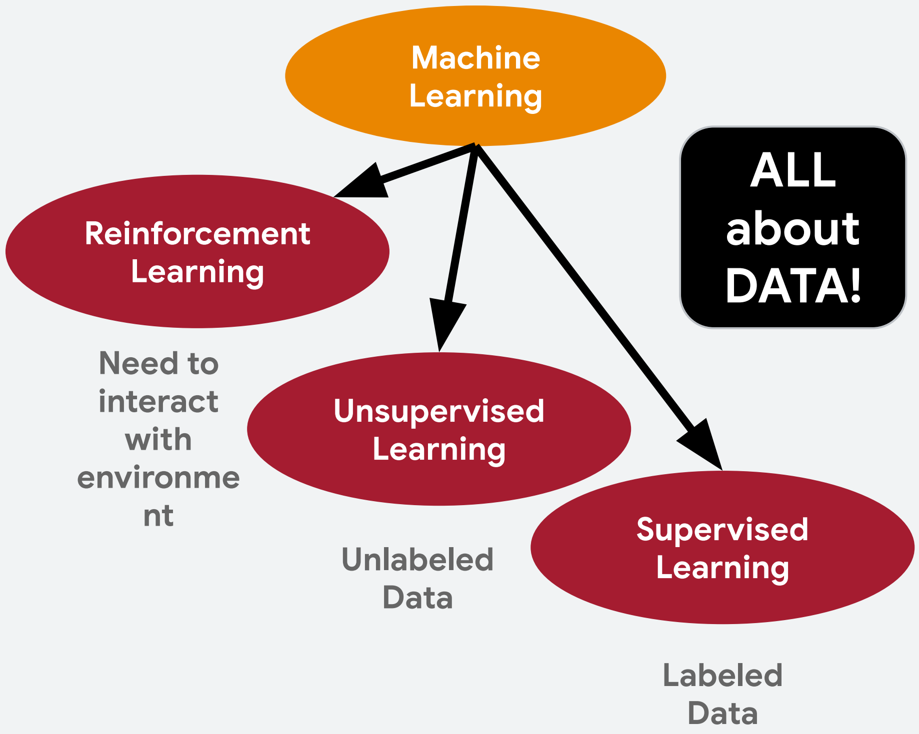
Quick Recap:

- AI > ML > Deep Learning



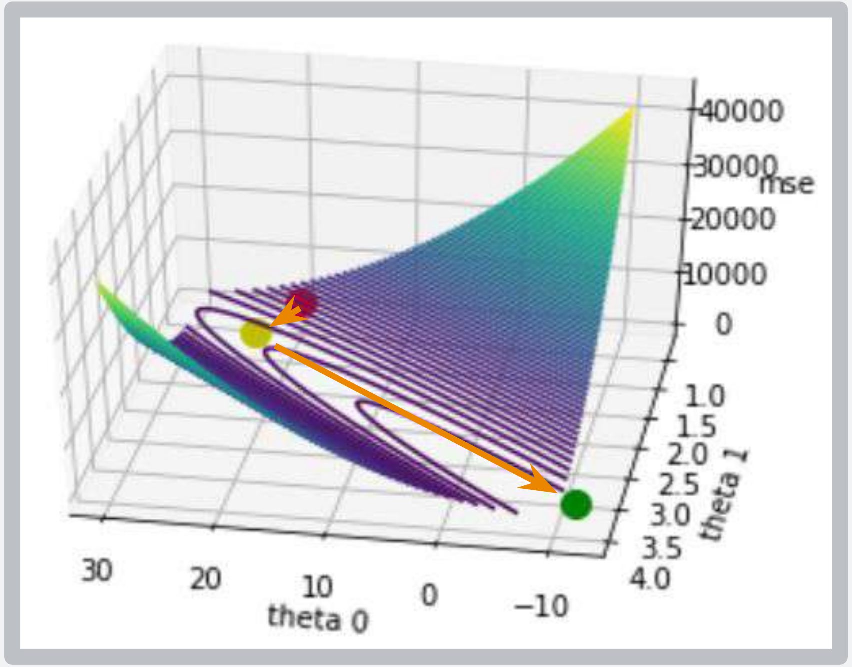
Quick Recap:

- AI > ML > Deep Learning
- Different methods of learning are defined by their **data**
- We will focus on **(Deep) Supervised Learning** in CS249r



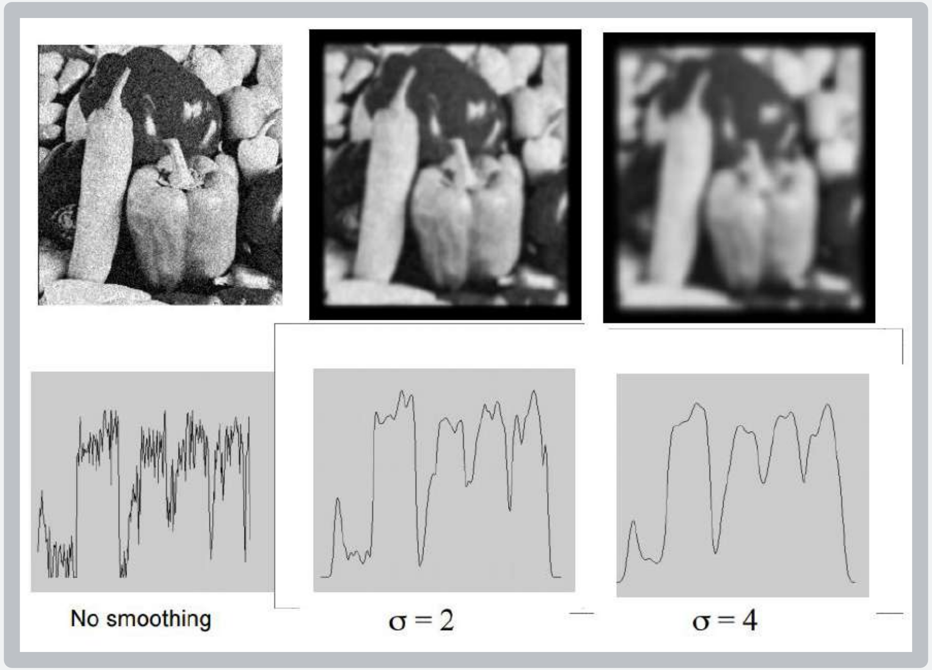
Quick Recap:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a **model with parameters, loss and activation functions, hyperparameters** (learning rate) and **gradient descent**
- Consider **overfitting** and **regularization/test data**



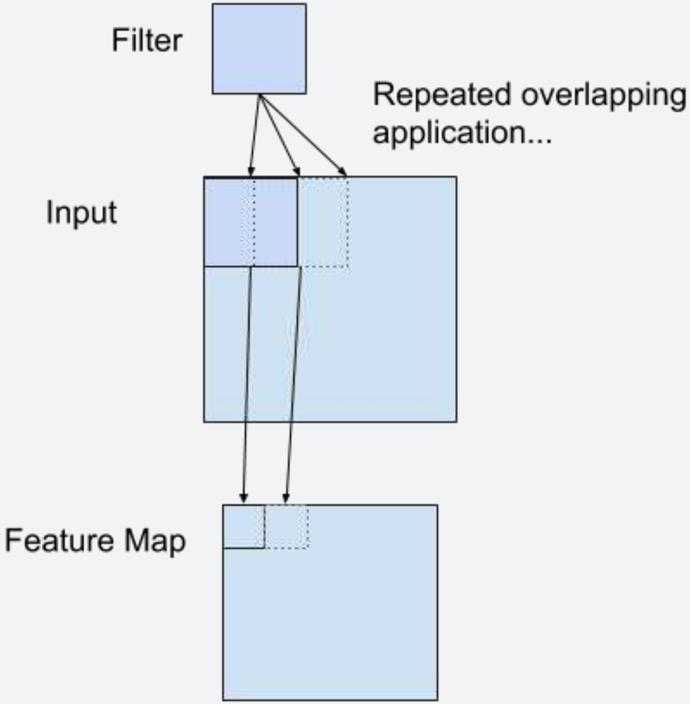
Quick Recap:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters and gradient descent
- Consider overfitting and regularization
- Data needs to be **pre-processed**



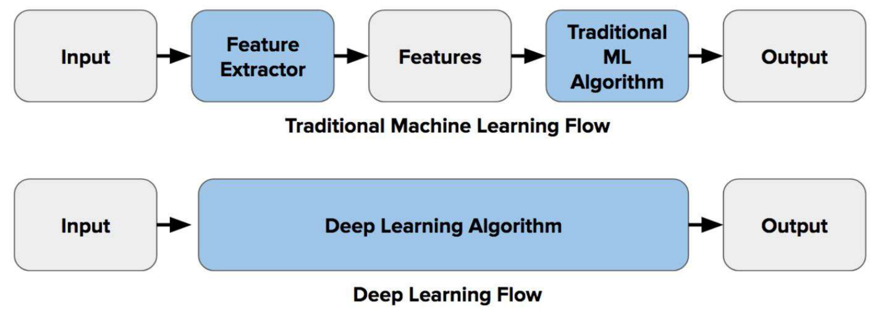
Quick Recap:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters and gradient descent
- Consider overfitting and regularization
- Data needs to be pre-processed
- **Spatial features** can be found through **convolutions**



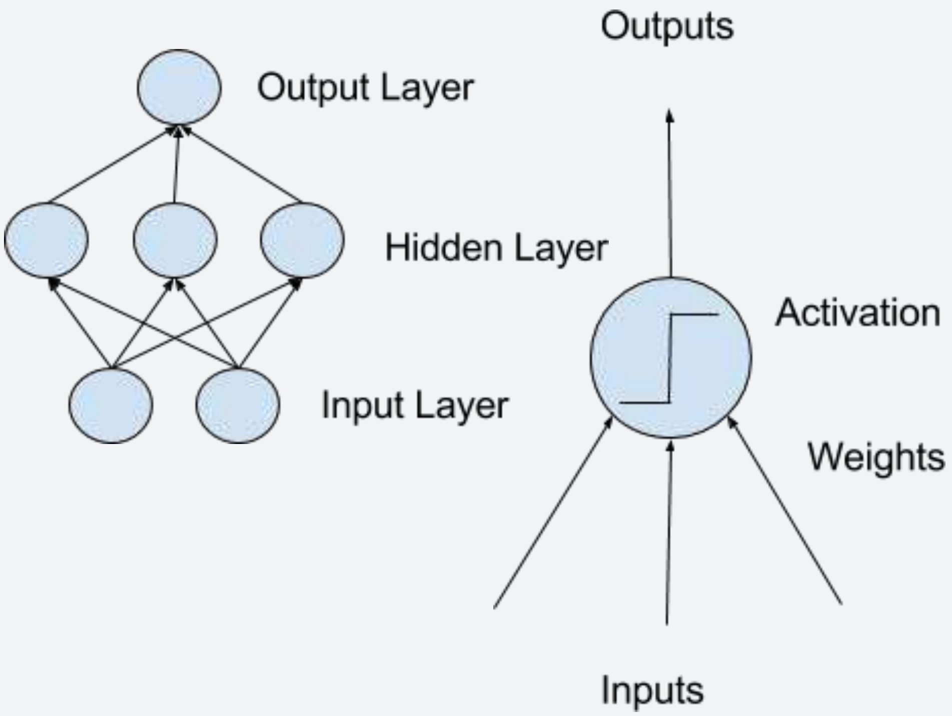
Quick Recap:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters and gradient descent
- Consider overfitting and regularization
- Data needs to be pre-processed
- Spatial features can be found through convolutions
- **Deep Learning automates** the designs and interactions of features



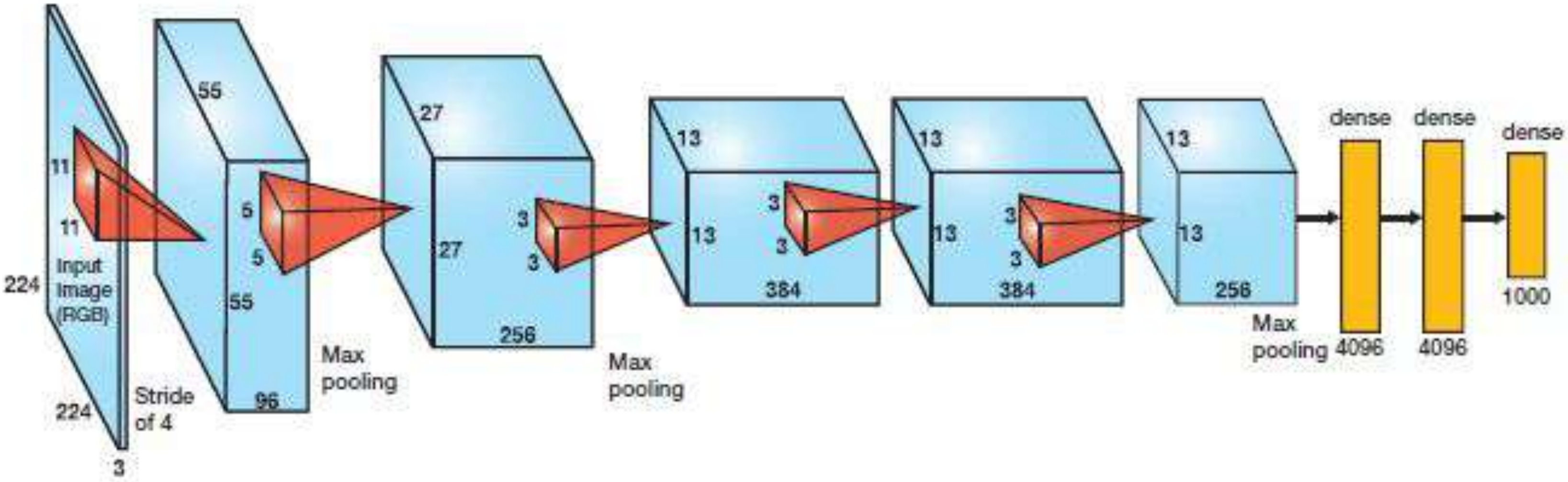
Quick Recap:

- AI > ML > Deep Learning
- Different methods of learning are defined by their data. We will focus on (Deep) Supervised Learning in CS249r
- Classification and Regression rely on a model with parameters, loss and activation functions, hyperparameters and gradient descent
- Consider overfitting and regularization
- Data needs to be pre-processed
- Spatial features can be found through convolutions
- Deep Learning automates the designs and interactions of features by constructing **deep networks** of **nonlinearly activated**, connected neurons



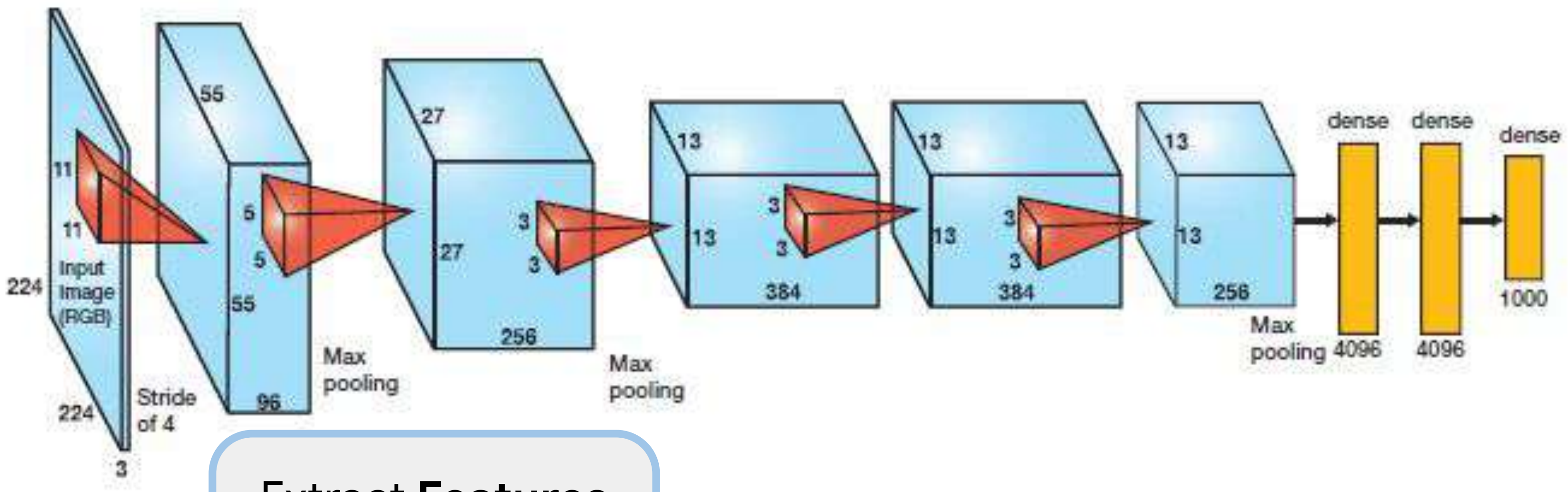
The Rise of Deep Learning:
AlexNet

AlexNet



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

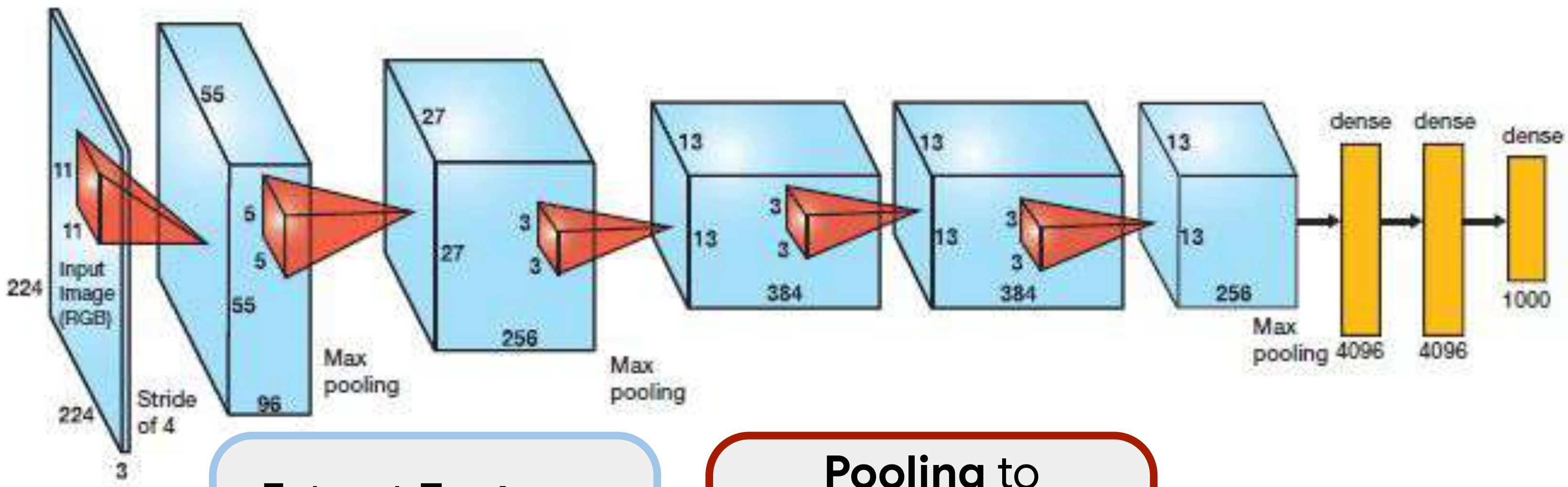
AlexNet Model Design



Extract **Features** from data with **Convolutions**

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet Model Design

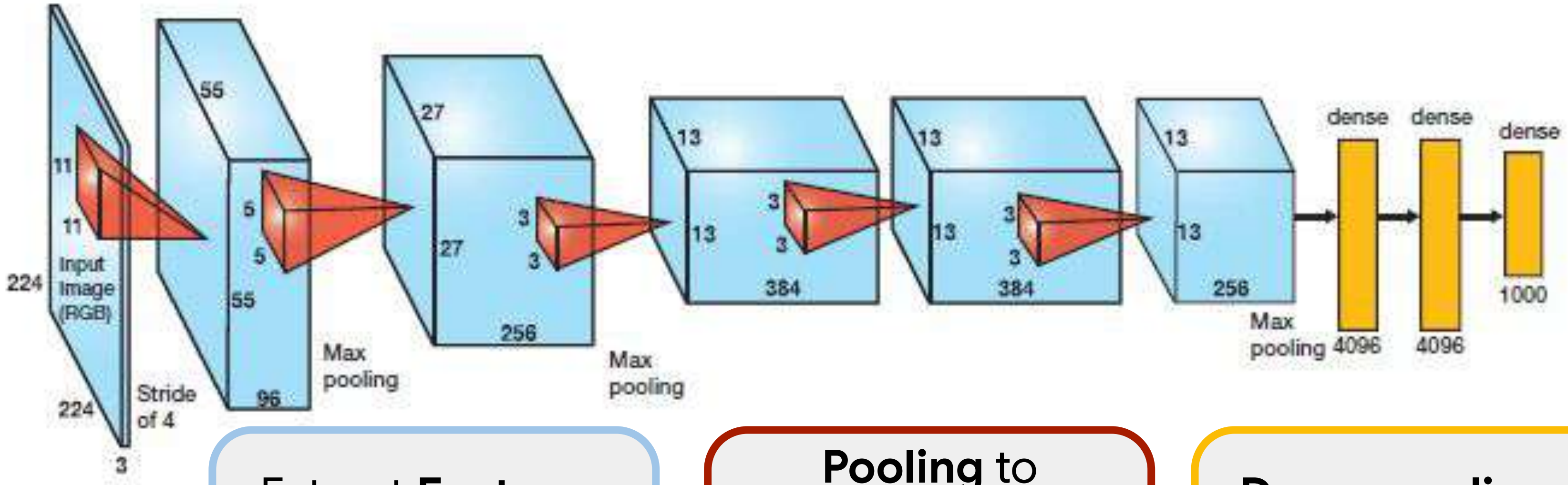


Extract **Features** from data with **Convolutions**

Pooling to **summarize** for higher level features

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet Model Design



Extract **Features** from data with **Convolutions**

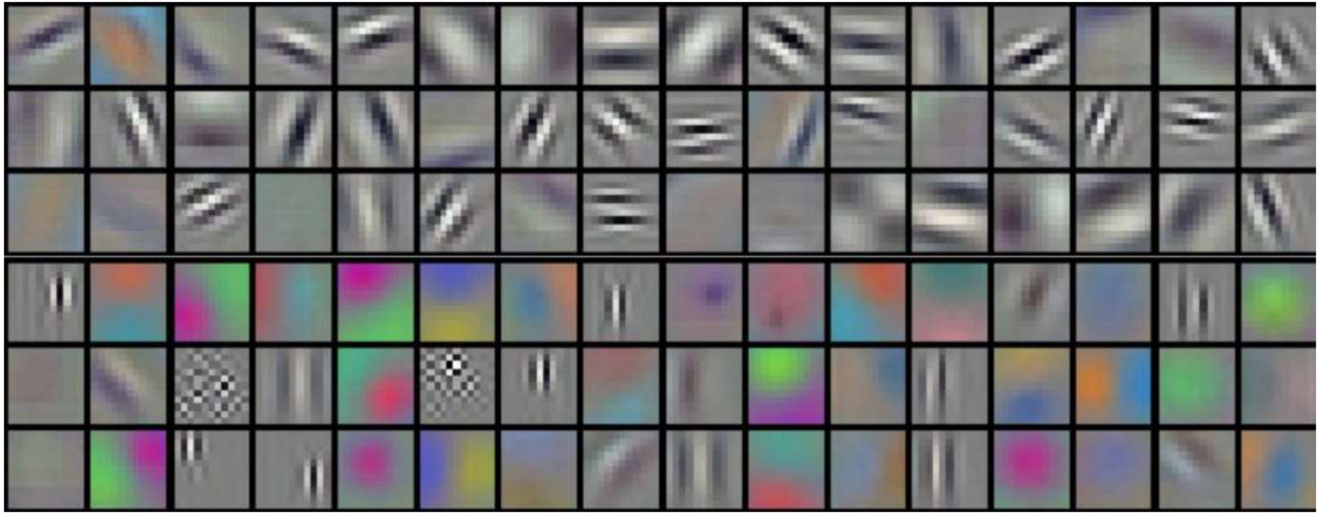
Pooling to **summarize** for higher level features

Dense nonlinear layers to **classify** the final features

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet Model Design

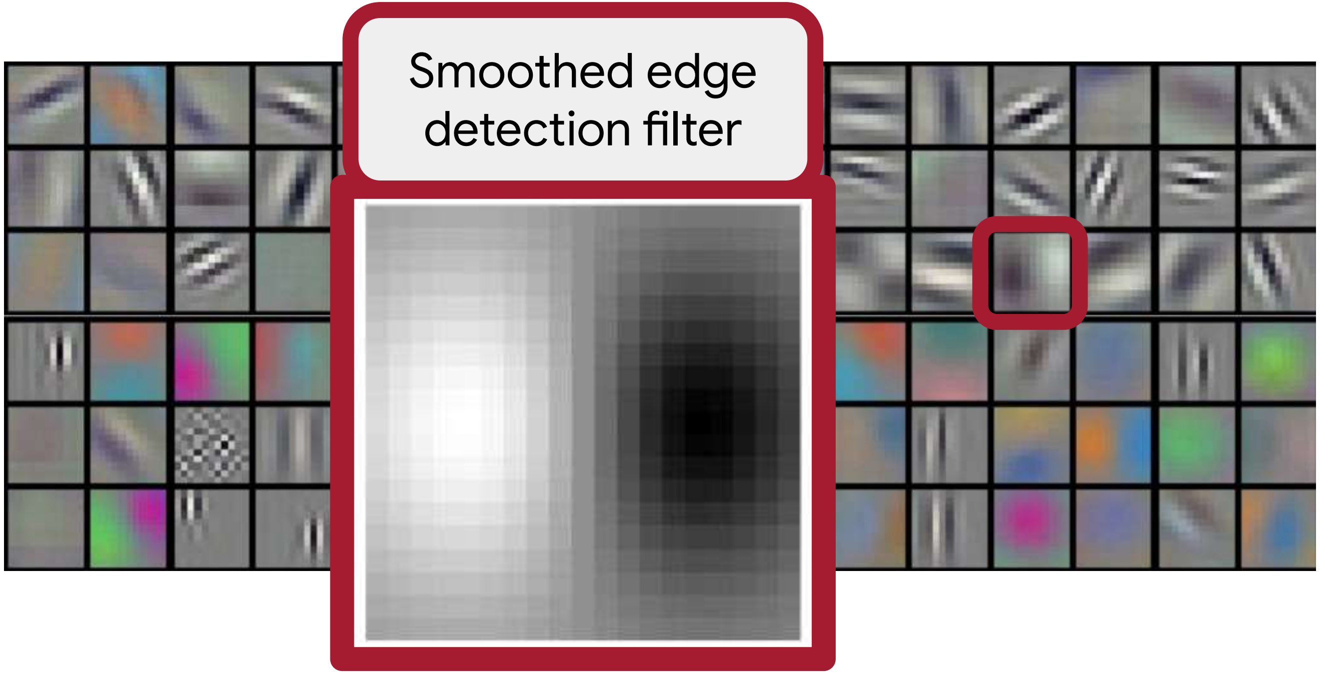
The filters learned by the NN for the initial convolution layer look like standard CV filters!



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet Model Design

The filters learned by the NN for the initial convolution layer look like standard CV filters!



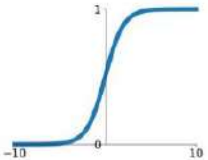
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

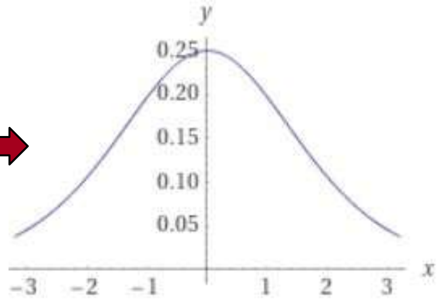
AlexNet Model Design

ReLU activation function to avoid
Saturation and **Vanishing Gradients**

Sigmoid

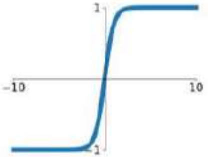
$\sigma(x) = \frac{1}{1+e^{-x}}$

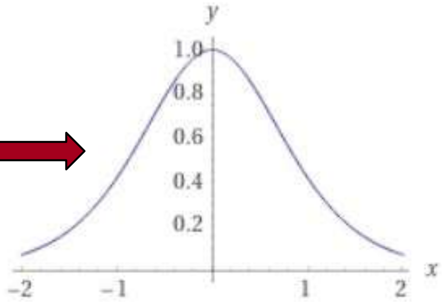




tanh

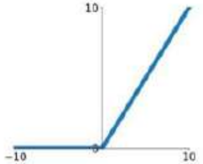
$\tanh(x)$





ReLU

$\max(0, x)$



https://towardsdatascience.com/complete-guide-of-activation-functions-34076e95d044

AlexNet Model Design

More than just a design the training process mattered too!

Extract **Features** from data with **Convolutions**

Pooling to **summarize** for higher level features

Dense nonlinear layers to **classify** the final features

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

AlexNet Training

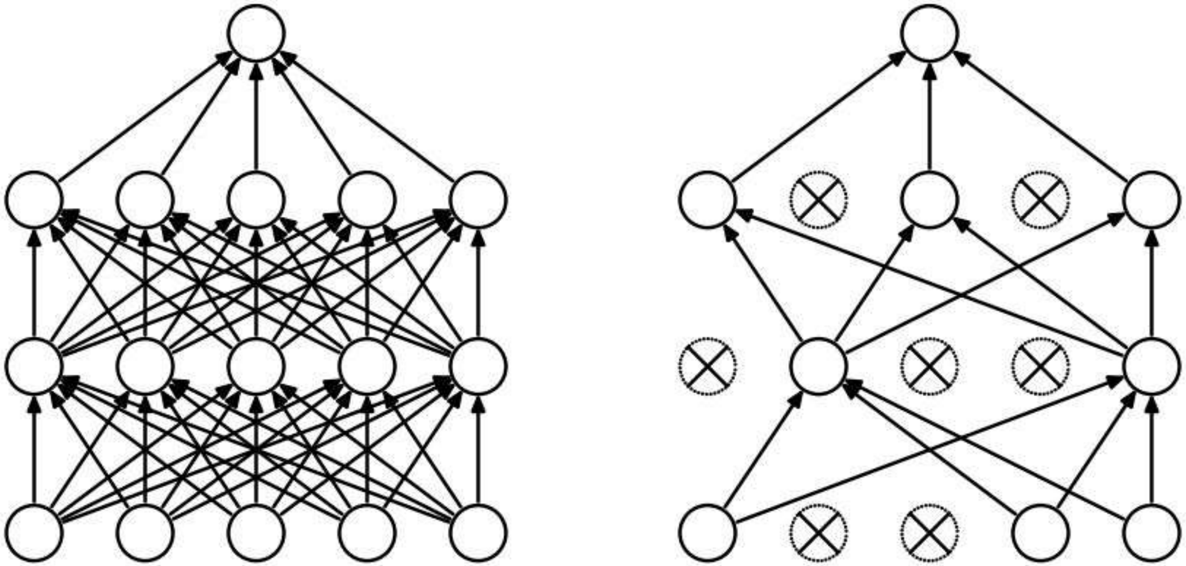
Dropout, and **Data Augmentation**, to avoid overfitting

Batch Updates for stability

Multi-GPU for speed

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

Dropout



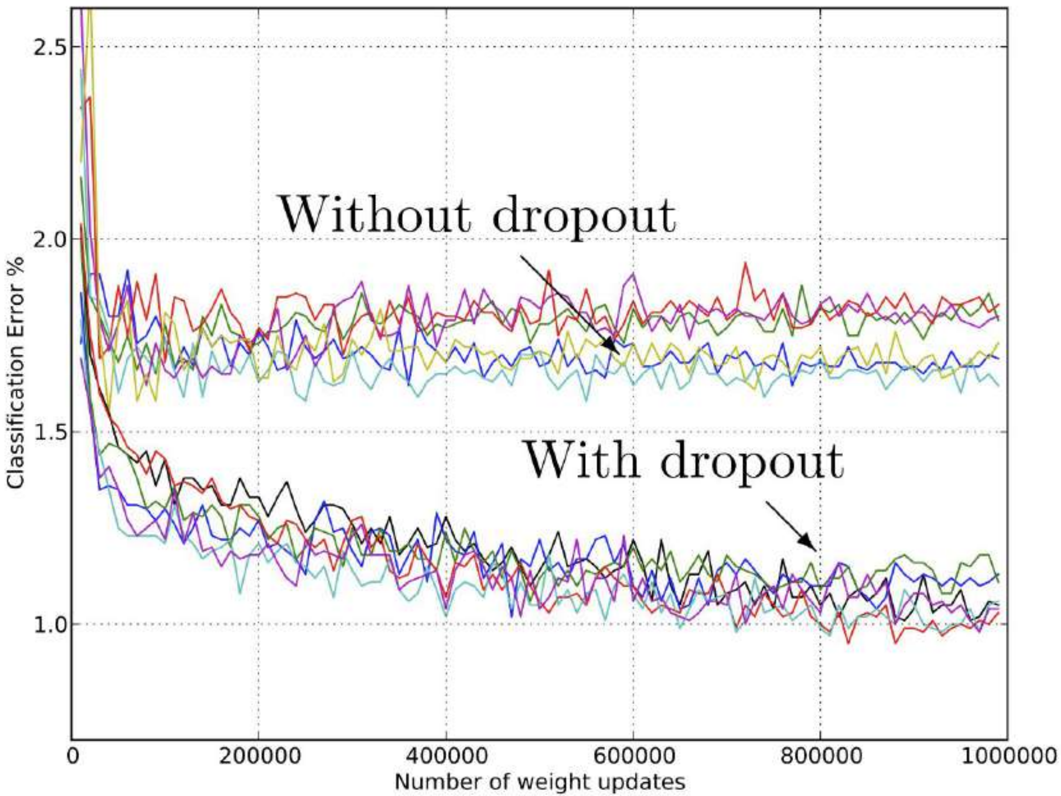
(a) Standard Neural Net

(b) After applying dropout.

Randomly remove nodes from the network during training

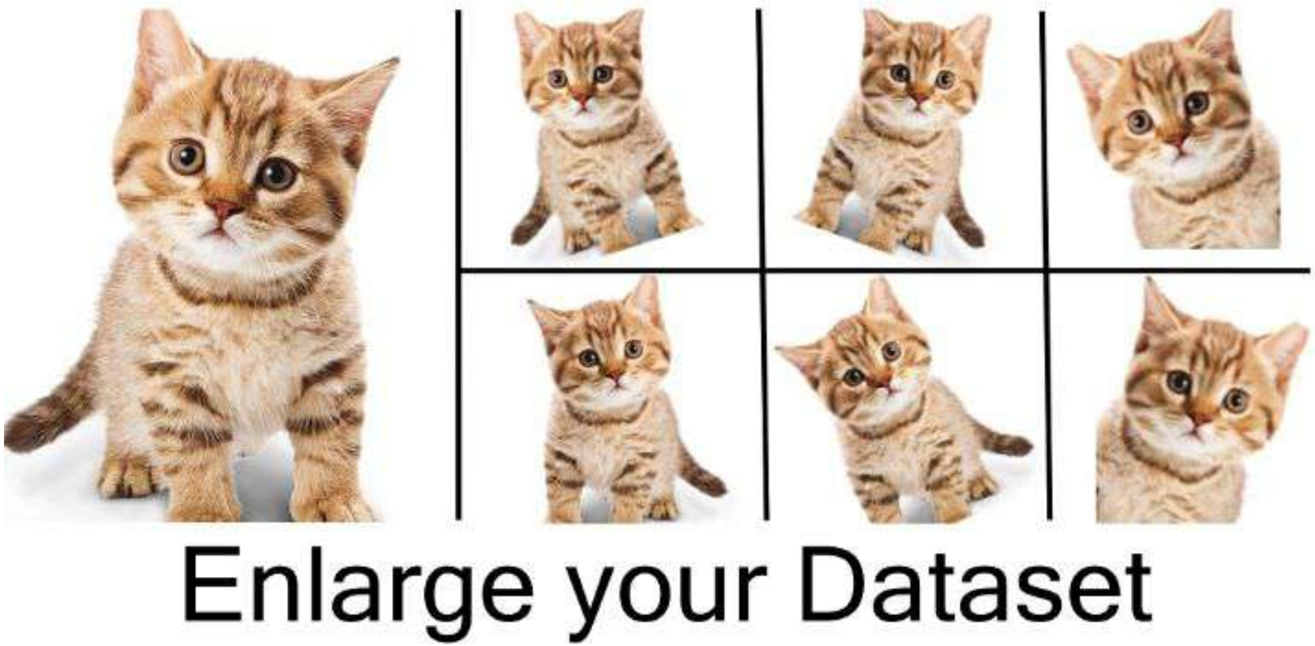
https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Dropout



https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Data Augmentation



https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/

AlexNet Training

Dropout, and **Data Augmentation**, to avoid overfitting

Batch Updates for stability

Multi-GPU for speed

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

Quick Aside
Stochastic Gradient Descent

We want to **update the model parameters** based on the training data -- we use **(stochastic) gradient descent**!

Quick Aside

Stochastic Gradient Descent

We want to **update the model parameters** based on the training data -- we use **(stochastic) gradient descent**!

Instead of computing the gradient based on all of the data we use **(one) sample** of data **at a time**

Quick Aside
Stochastic Gradient Descent

We want to **update the model parameters** based on the training data -- we use **(stochastic) gradient descent**!

Instead of computing the gradient based on all of the data we use **(one) sample** of data **at a time**

The full gradient is:
O(|outputs| |weights| **|data|**)

E.g., visual wake words has a 40GB dataset…

So this is **MUCH CHEAPER**

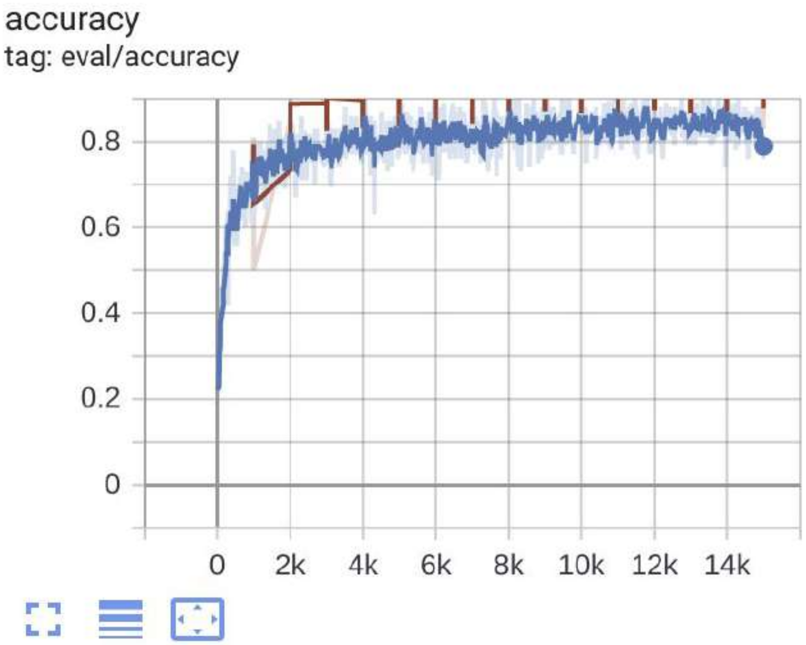
Quick Aside
Stochastic Gradient Descent

Pros
• Computationally cheap and doesn’t require full data for an update

Quick Aside
Stochastic Gradient Descent

Pros
• Computationally cheap and doesn't require full data for an update

Cons
• Makes **myopic / noisy** updates



Quick Aside
Stochastic Gradient Descent

Pros
• Computationally cheap and doesn't require full data for an update

Cons
• Makes myopic / noisy updates which often requires a small learning rate **requiring many iterations**

https://openai.com/blog/learning-dexterity/

Quick Aside
Stochastic Gradient Descent

Pros
• Computationally cheap and doesn't require full data for an update

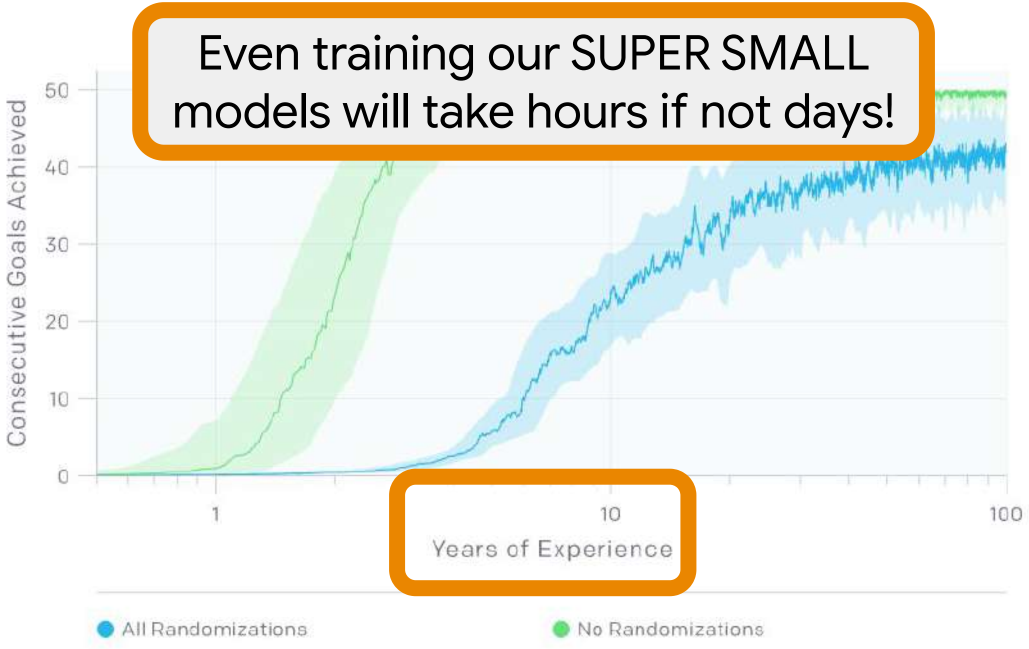
Cons
• Makes myopic / noisy updates which often requires a small learning rate **requiring many iterations**

Even training our SUPER SMALL models will take hours if not days!

https://openai.com/blog/learning-dexterity/

Quick Aside
Stochastic Gradient Descent

So how might we speed up training?

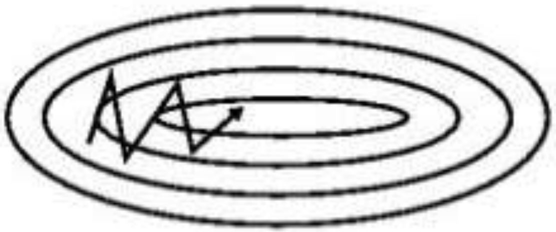


https://openai.com/blog/learning-dexterity/

Momentum to accelerate Stochastic Gradient Descent

$$v \leftarrow \boxed{\gamma v} + (1-\gamma)\nabla$$
$$\theta \leftarrow \theta - \alpha v$$

(a) SGD without momentum



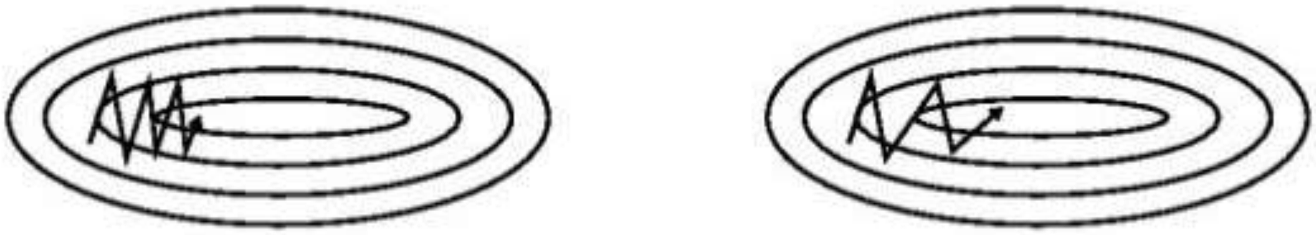
(b) SGD with momentum

https://www.slideshare.net/SebastianRuder/optimization-for-deep-learning

Momentum to accelerate Stochastic Gradient Descent

$$v \leftarrow \boxed{\gamma v} + (1 - \gamma)\nabla$$
$$\theta \leftarrow \theta - \alpha v$$

There are a ton of ways to do this:
[https://ruder.io/optimizing-gradient- descent/](https://ruder.io/optimizing-gradient-descent/)



(a) SGD without momentum (b) SGD with momentum

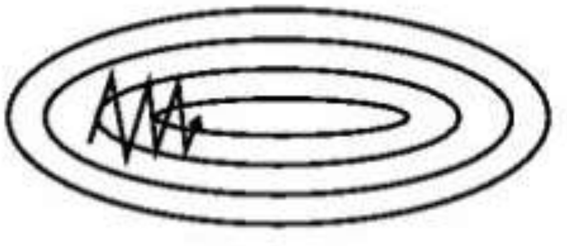
https://www.slideshare.net/SebastianRuder/optimization-for-deep-learning

Momentum to accelerate Stochastic Gradient Descent

$$v \leftarrow \boxed{\gamma v} + (1-\gamma)\nabla$$
$$\theta \leftarrow \theta - \alpha v$$

There are a ton of ways to do this:
https://ruder.io/optimizing-gradient- descent/

The best ones use an
adaptive learning rate

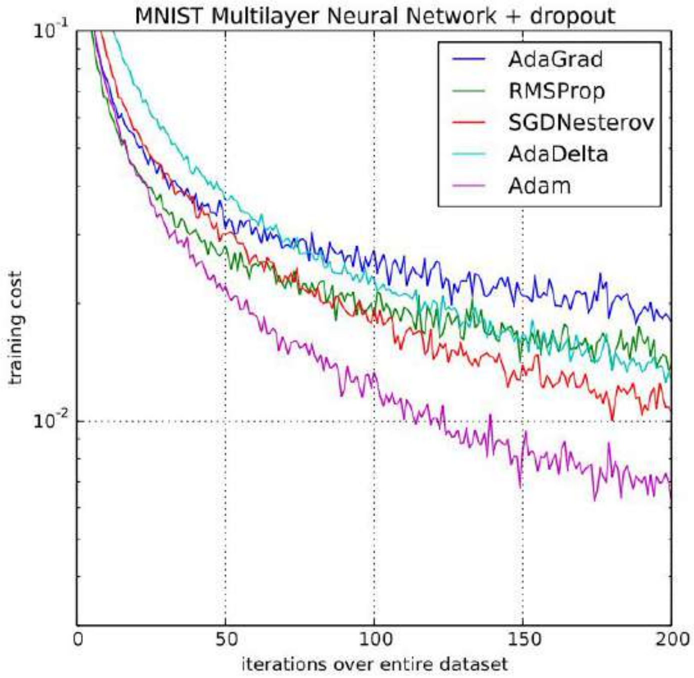


(a) SGD without momentum

(b) SGD with momentum

https://www.slideshare.net/SebastianRuder/optimization-for-deep-learning

Momentum to accelerate Stochastic Gradient Descent

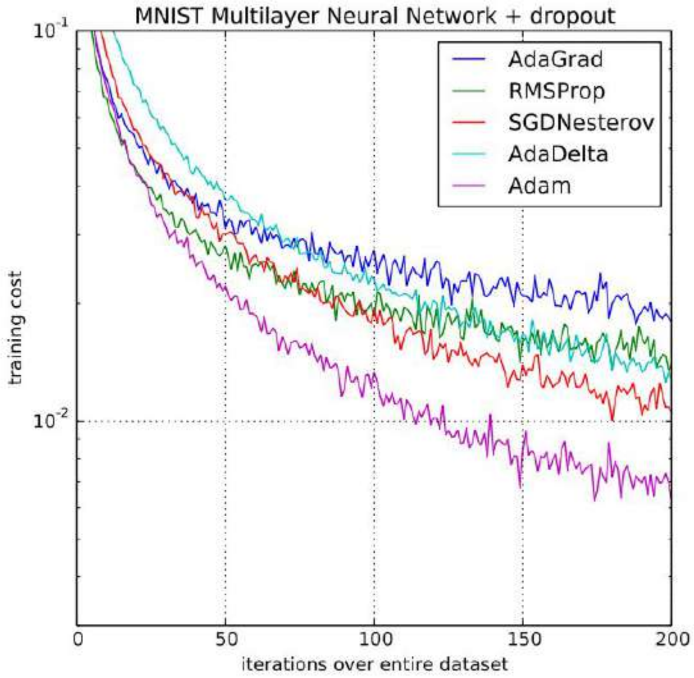


There are a ton of ways to do this:
https://ruder.io/optimizing-gradient- descent/

The best ones use an
adaptive learning rate

https://arxiv.org/pdf/1412.6980.pdf

Momentum to accelerate Stochastic Gradient Descent



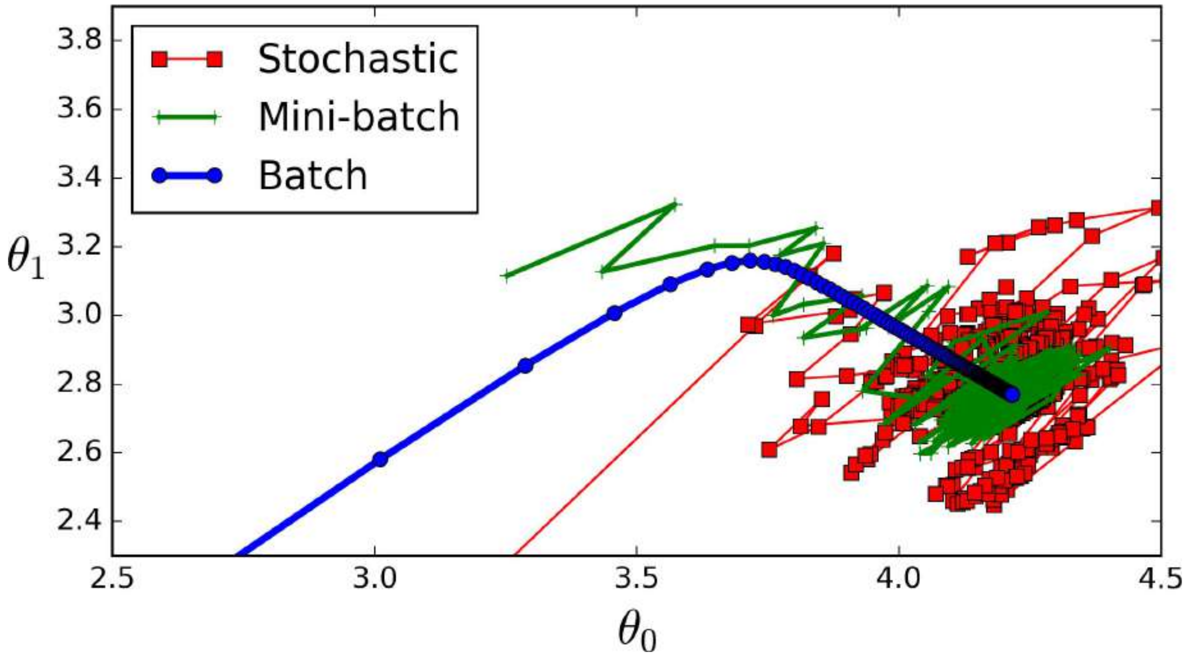
There are a ton of ways to do this:
https://ruder.io/optimizing-gradient- descent/

The best ones use an
adaptive learning rate

But the best optimizer for a given NN is still an open problem!

https://arxiv.org/pdf/1412.6980.pdf

(Mini) Batch Updates using Stochastic Gradient Descent



Batch size is another hyperparameter we can tune

https://stats.stackexchange.com/questions/153531/what-is-batch-size-in-neural-network

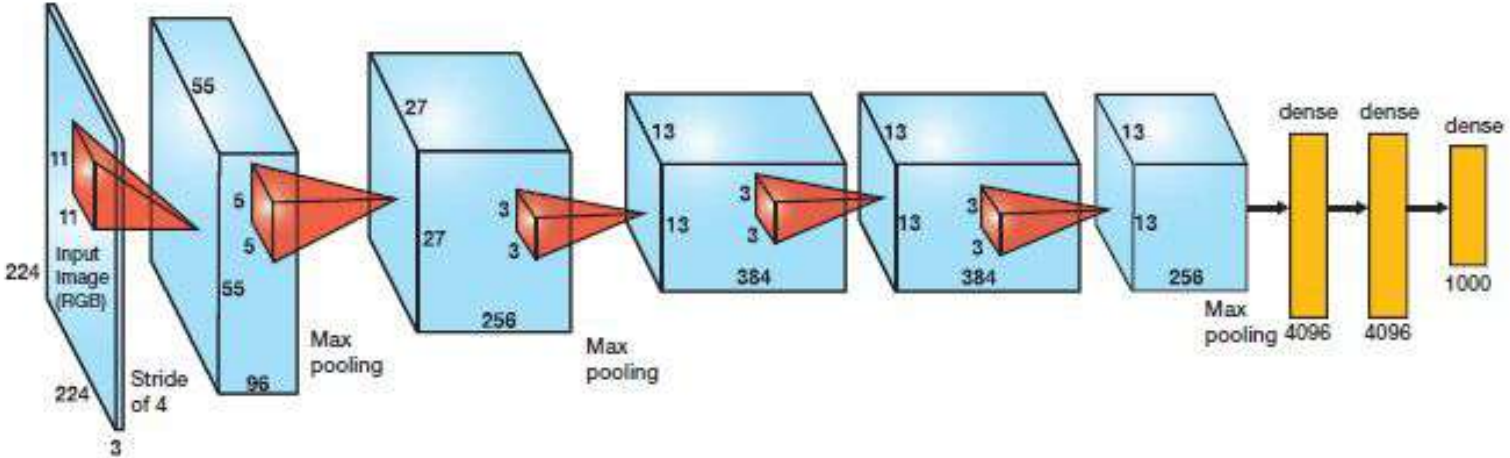
Multi-GPU Training

AlexNet **COULD NOT FIT INTO 3GB of RAM!!!!**

Today lots of **data-parallel training**

AlexNet
Deep Learning Insights

- Providing a **structured** model allows a computer to effectively learn (e.g., **convolutions**)

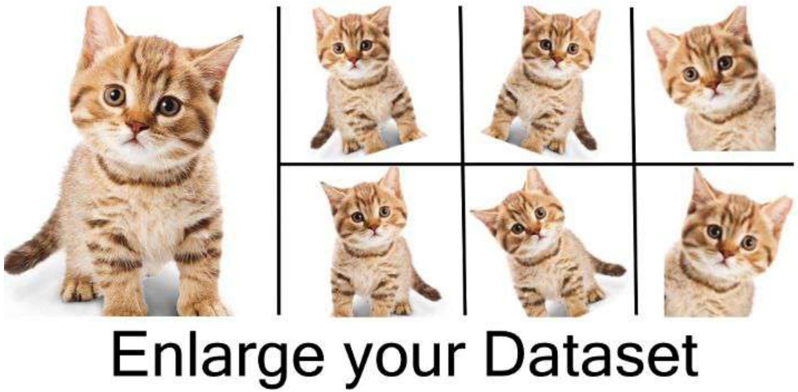


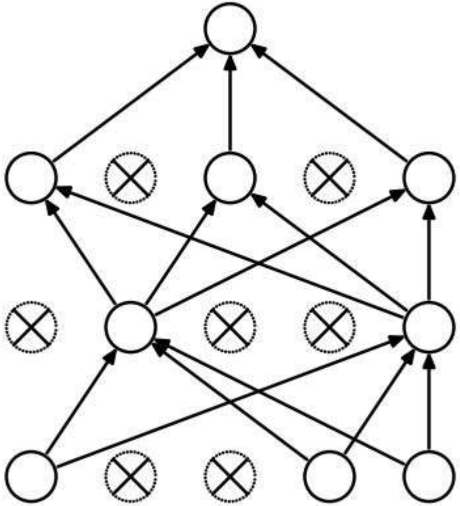
AlexNet
Deep Learning Insights

- Providing a structured model allows a computer to effectively learn (e.g., convolutions)
- There are a handful of practical activation functions - **ReLU** has become quite popular

AlexNet
Deep Learning Insights

- Providing a structured model allows a computer to effectively learn (e.g., convolutions)
- There are a handful of practical activation functions - ReLU has become quite popular
- Efficient learning requires **regularization** (e.g., **Dropout**, **Data Augmentation**)

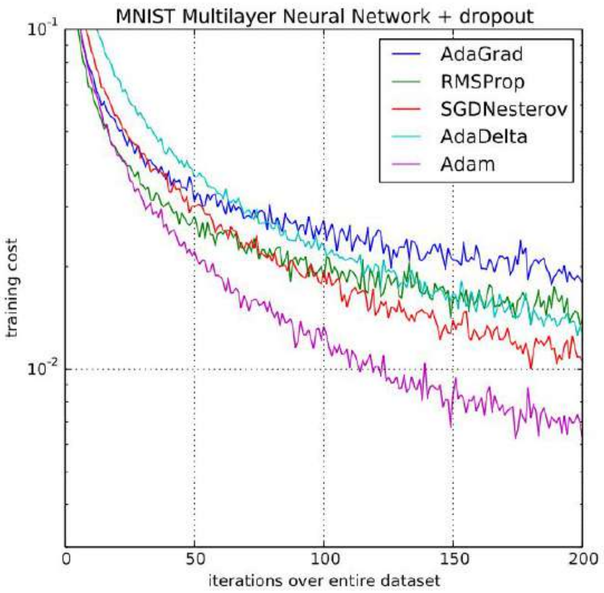




(b) After applying dropout.

AlexNet
Deep Learning Insights

- Providing a structured model allows a computer to effectively learn (e.g., convolutions)
- There are a handful of practical activation functions - ReLU has become quite popular
- Efficient learning requires regularization (e.g., Dropout, Data Augmentation, and Weight Decay)
- **Batch Updates** and **Adaptive Learning Rates** (often based on **momentum**) provide fast convergence with stability to **Stochastic Gradient Descent**

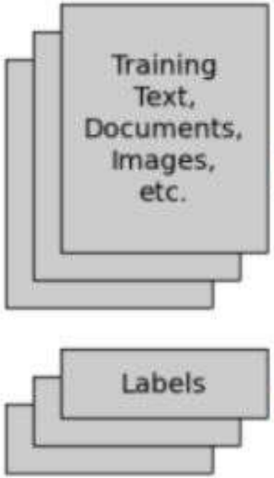


Deep Learning in Practice

Deep Supervised Learning

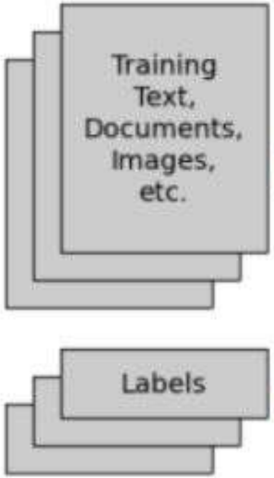
1. **Collect**
2. **Preprocess** and **design**
3. **Train**
4. **Evaluate**
5. **Deploy**

Deep Supervised Learning

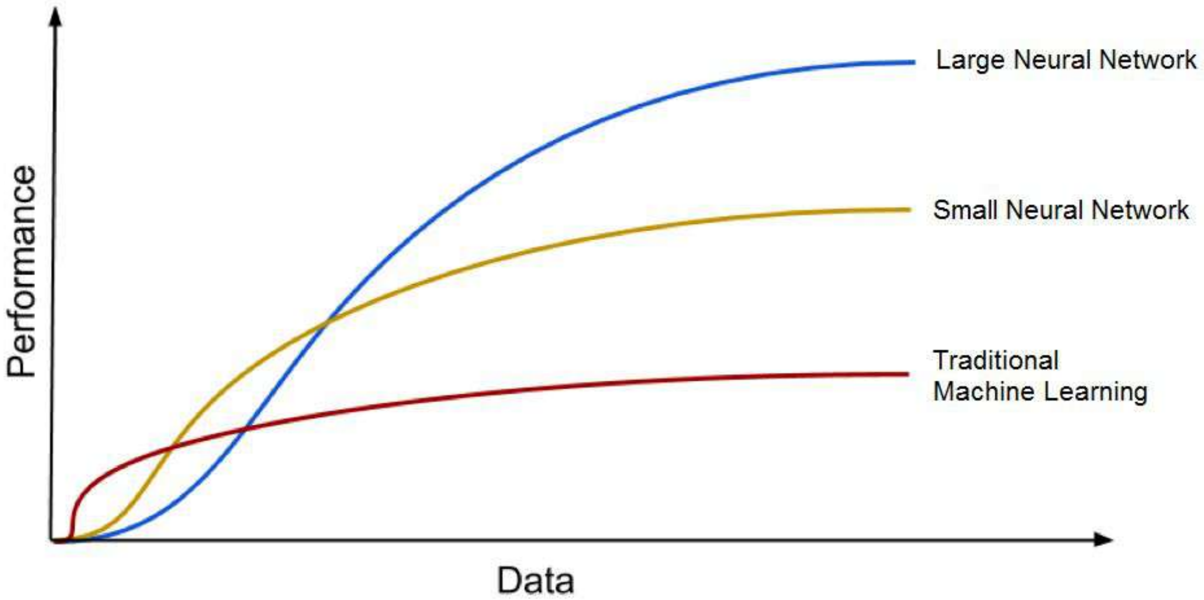


1. Collect **LOTS** of data!

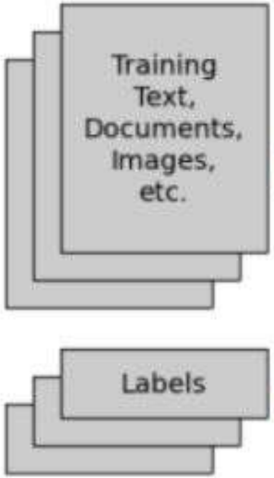
Deep Supervised Learning



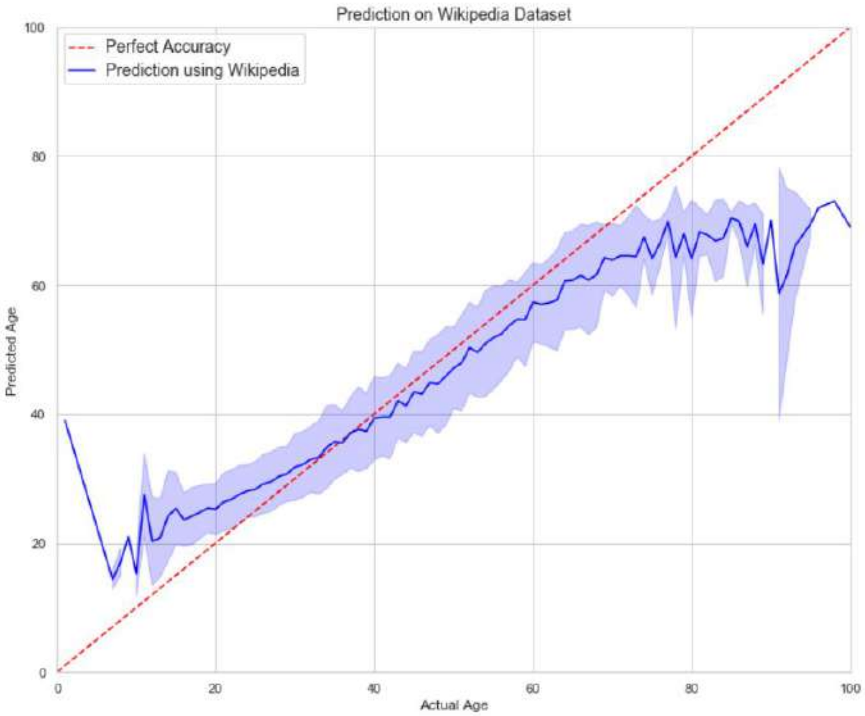
1. Collect **LOTS** of data!



Deep Supervised Learning

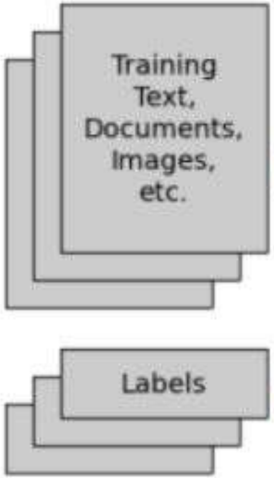


1. Collect **LOTS** of **UNBIASED** data!



https://towardsdatascience.com/bias-in-machine-learning-how-facial-recognition-models-show-signs-of-racism-sexism-and-ageism-32549e2c972d

Deep Supervised Learning



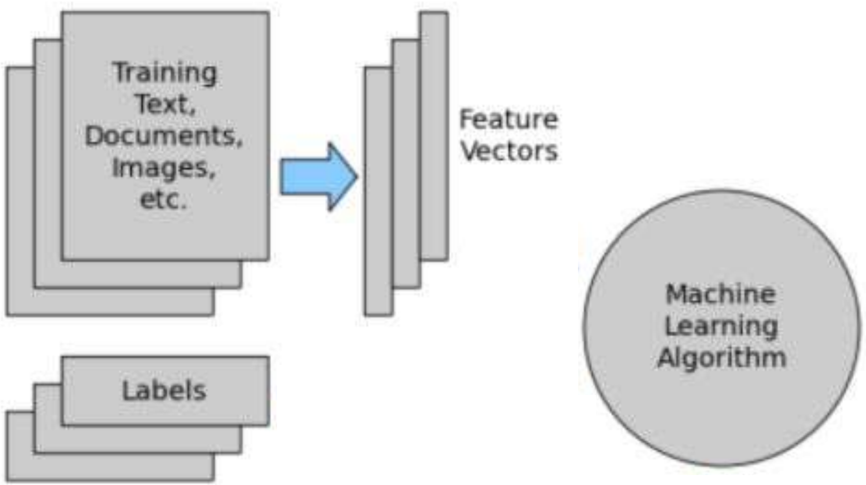
Error Rates in Commercial Gender Classification Products

		Face++ 旷视	IBM
Dark Skinned Female	20.8%	34.5%	34.7%
Light Skinned Female	1.7%	6.0%	7.1%
Dark Skinned Male	6.0%	0.7%	12.0%
Light Skinned Male	0.0%	0.8%	0.3%

1. Collect **LOTS** of **UNBIASED** data!

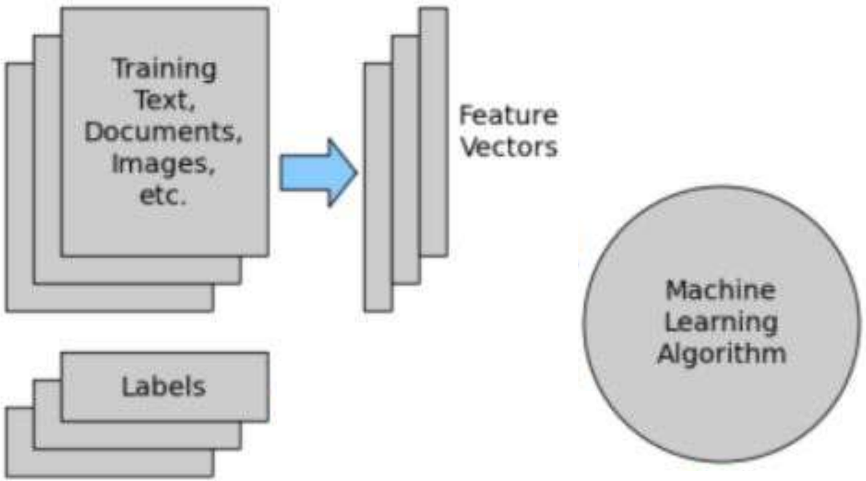
http://proceedings.mlr.press/v81/buolamwini18a/buolamwini18a.pdf
https://towardsdatascience.com/bias-in-machine-learning-how-facial-recognition-models-show-signs-of-racism-sexism-and-ageism-32549e2c972d

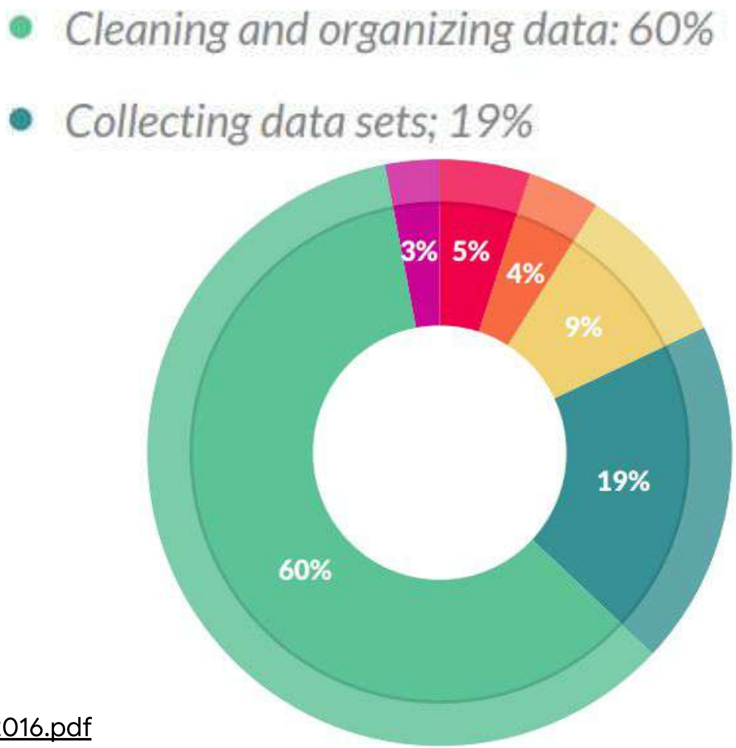
Deep Supervised Learning



2. **Preprocess** the data and **design** your Machine Learning **model**

Deep Supervised Learning

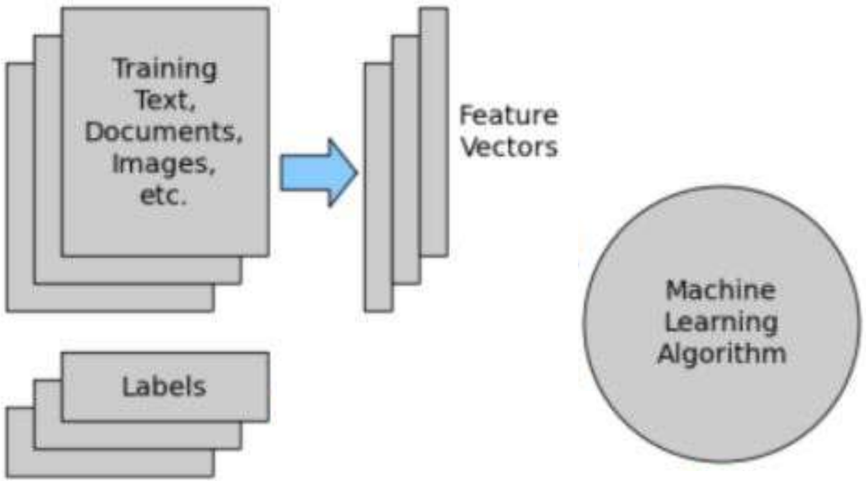


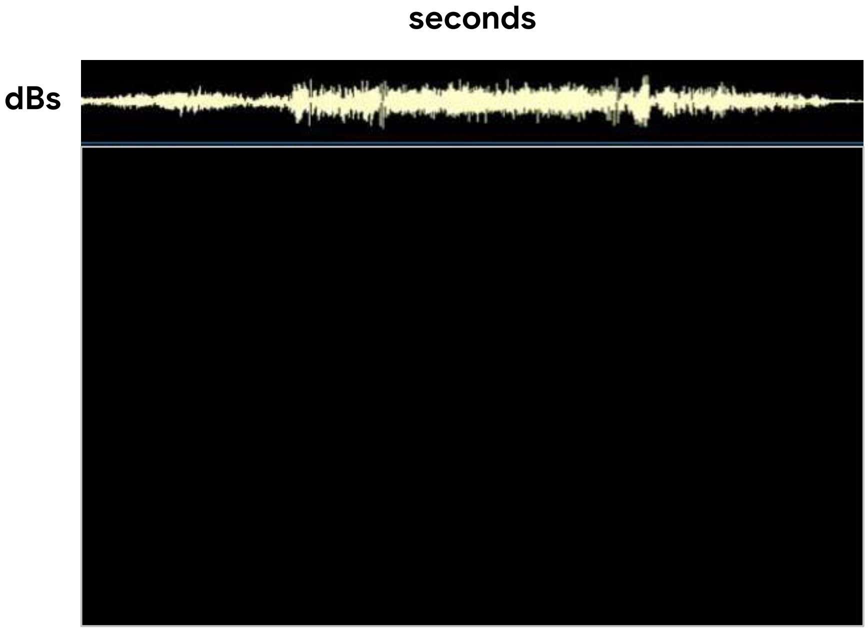


2. **Preprocess** the data and **design** your Machine Learning **model**

https://visit.figure-eight.com/rs/416-ZBE-142/images/CrowdFlower_DataScienceReport_2016.pdf

Deep Supervised Learning

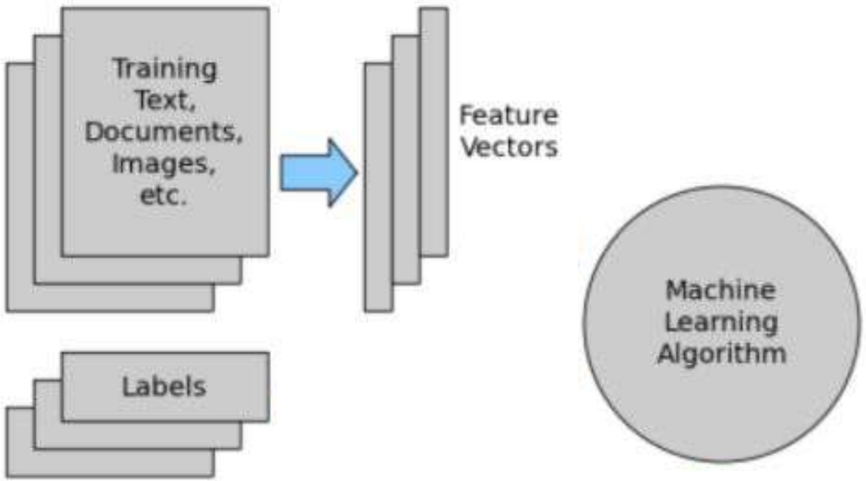




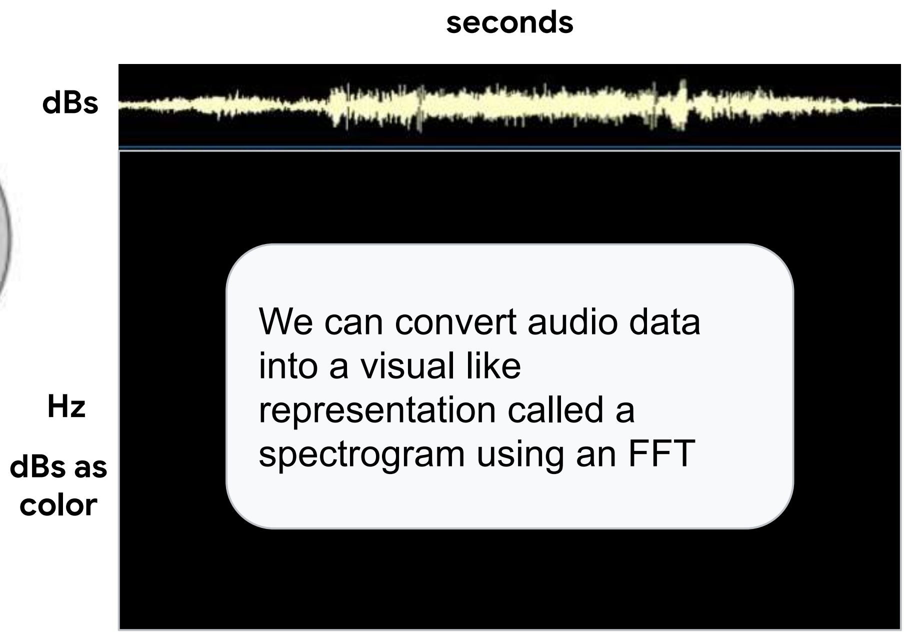
2. **Preprocess** the data and **design** your Machine Learning **model**

https://towardsdatascience.com/beginners-guide-to-speech-analysis-4690ca7a7c05

Deep Supervised Learning

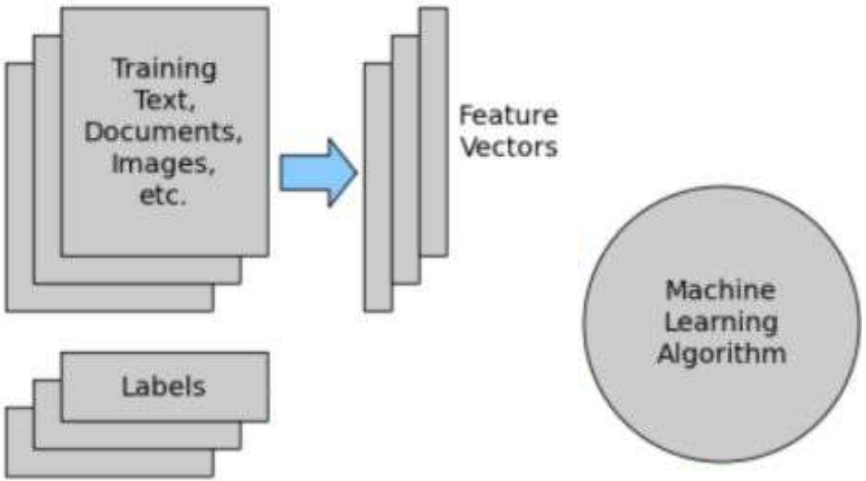


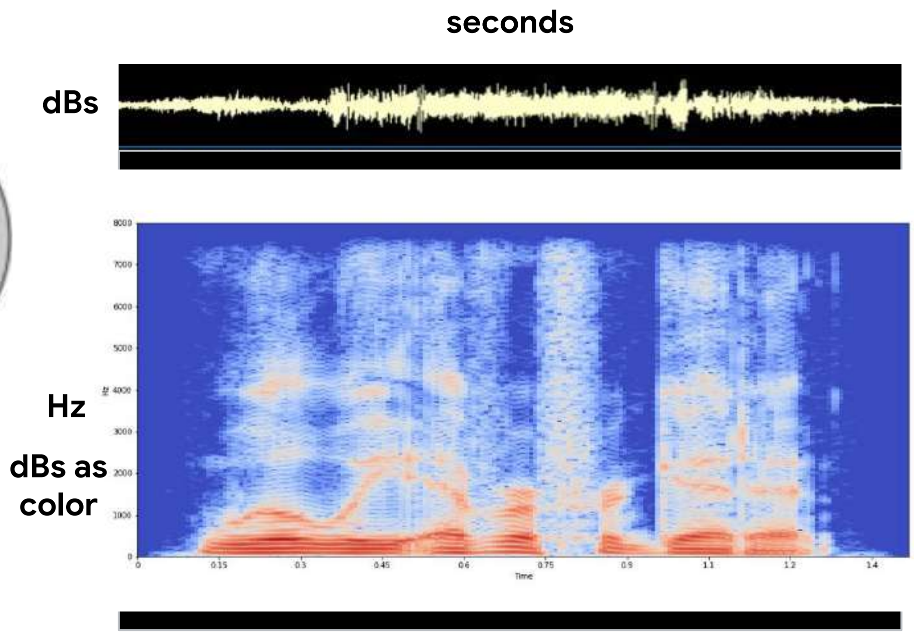
2. **Preprocess** the data and **design** your Machine Learning **model**



https://towardsdatascience.com/beginners-guide-to-speech-analysis-4690ca7a7c05

Deep Supervised Learning

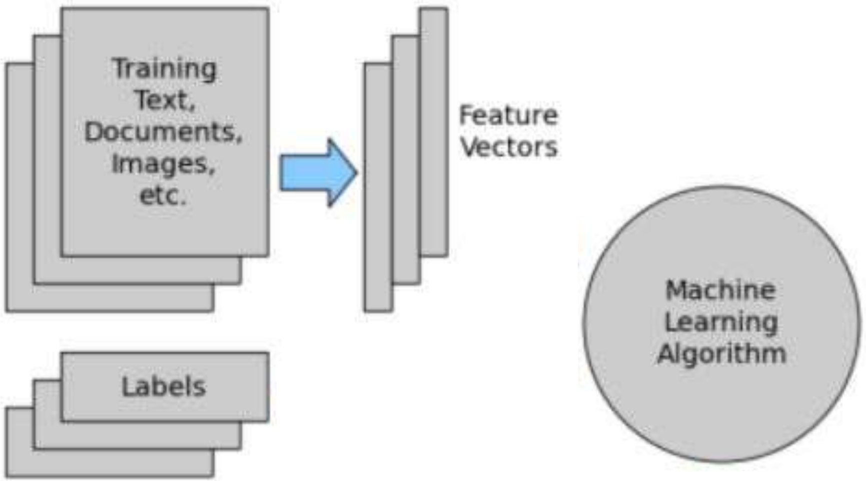




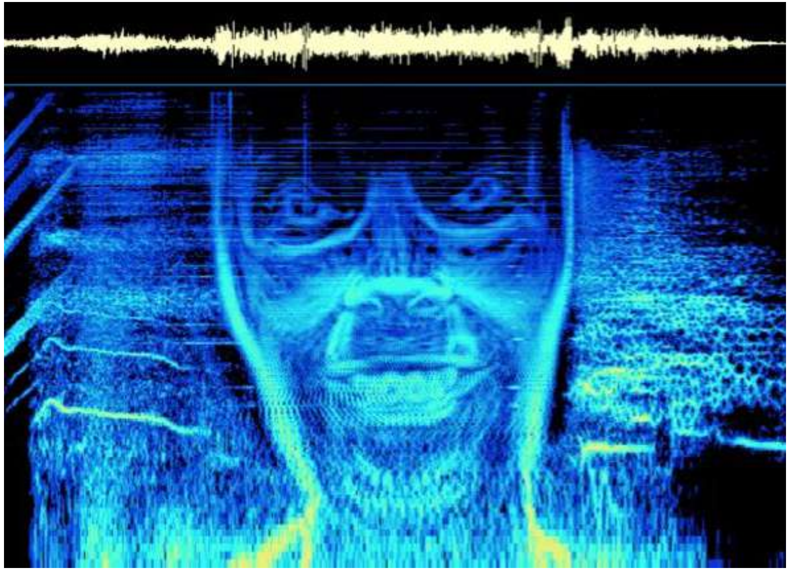
2. **Preprocess** the data and **design** your Machine Learning **model**

https://towardsdatascience.com/beginners-guide-to-speech-analysis-4690ca7a7c05

Deep Supervised Learning

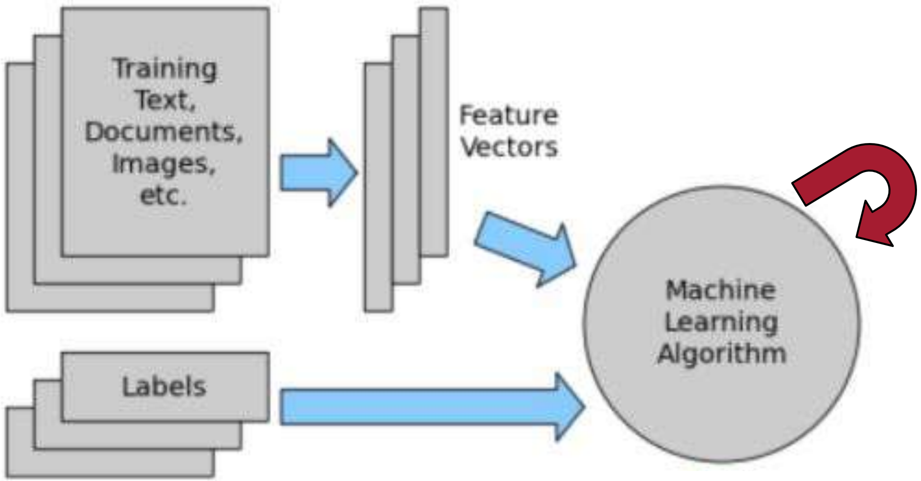


2. **Preprocess** the data and **design** your Machine Learning **model**



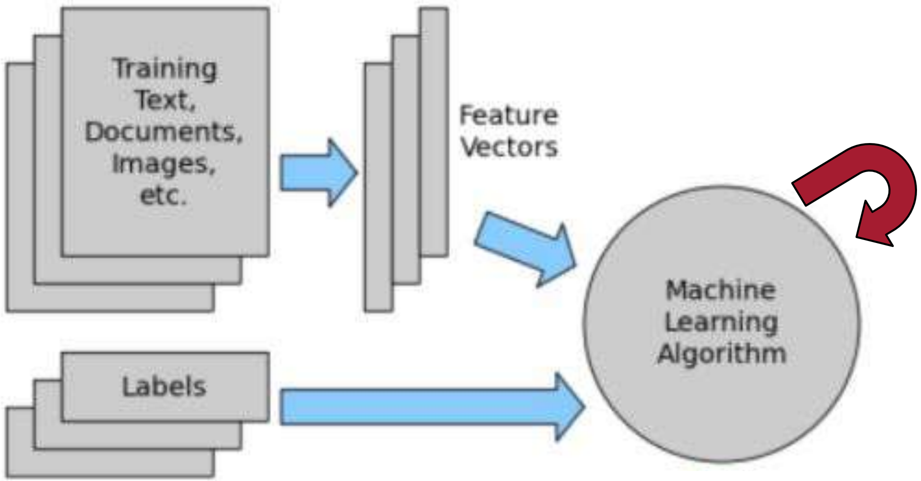
https://www.mentalfloss.com/article/61815/how-musicians-put-hidden-images-their-songs

Deep Supervised Learning



3. **Train** your model

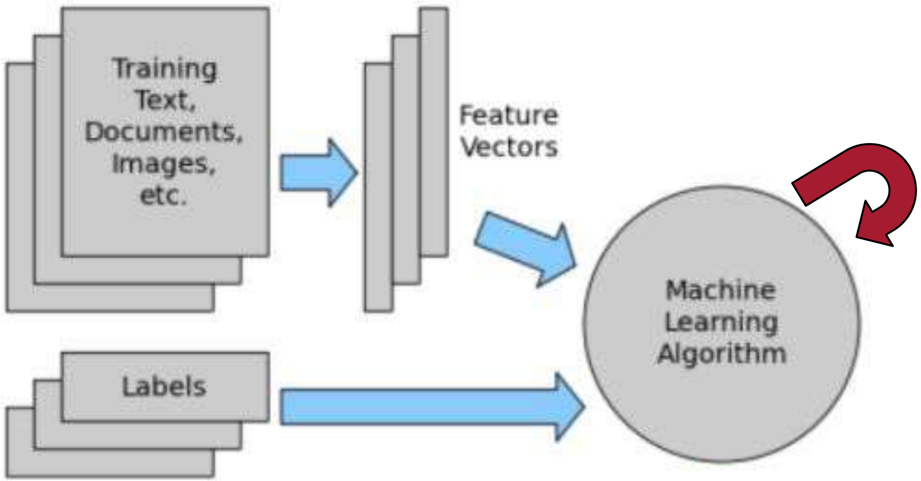
Deep Supervised Learning



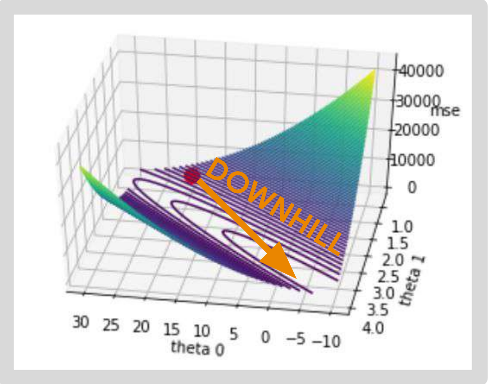
We want to **update the model parameters** based on the training data

3. **Train** your model

Deep Supervised Learning

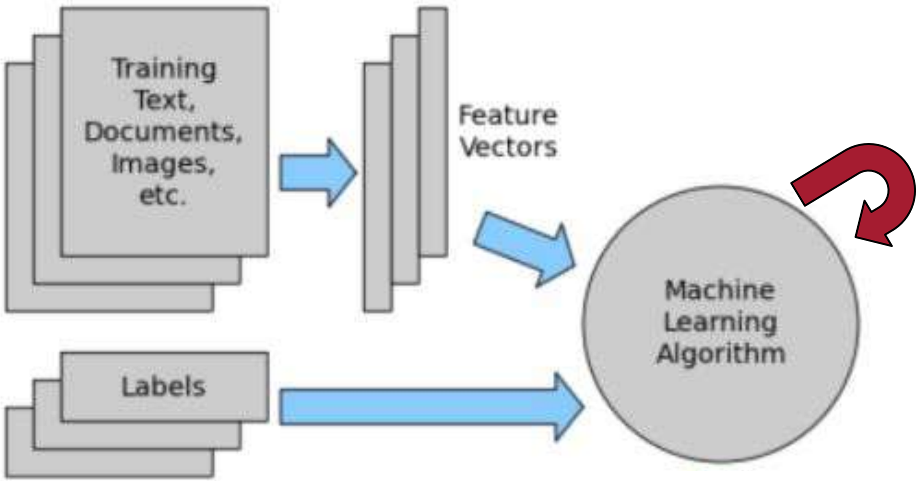


We want to **update the model parameters** based on the training data -- we use **stochastic gradient descent**!



3. **Train** your model

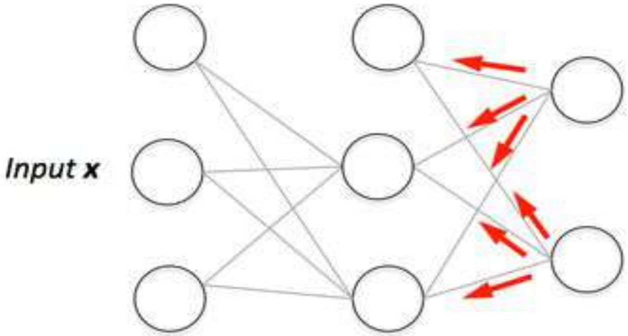
Deep Supervised Learning



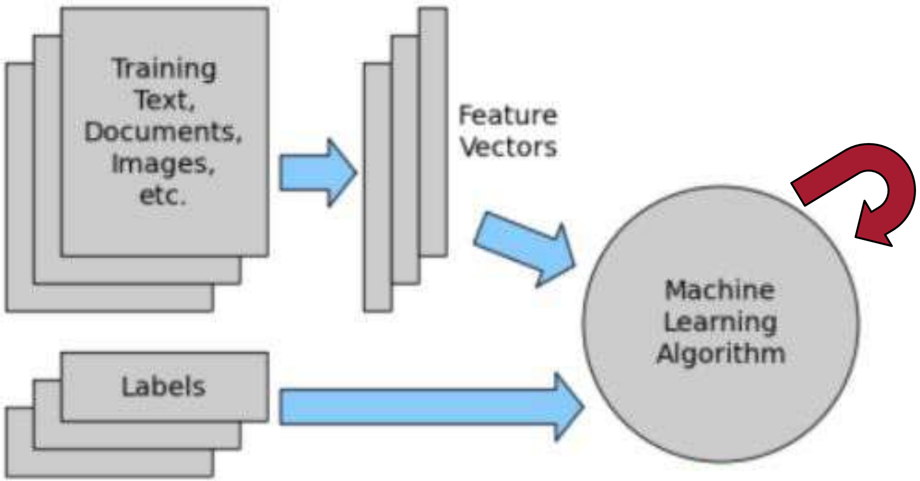
We want to **update the model parameters** based on the training data -- we use **stochastic gradient descent**!

3. **Train** your model

This is called **backpropagation**



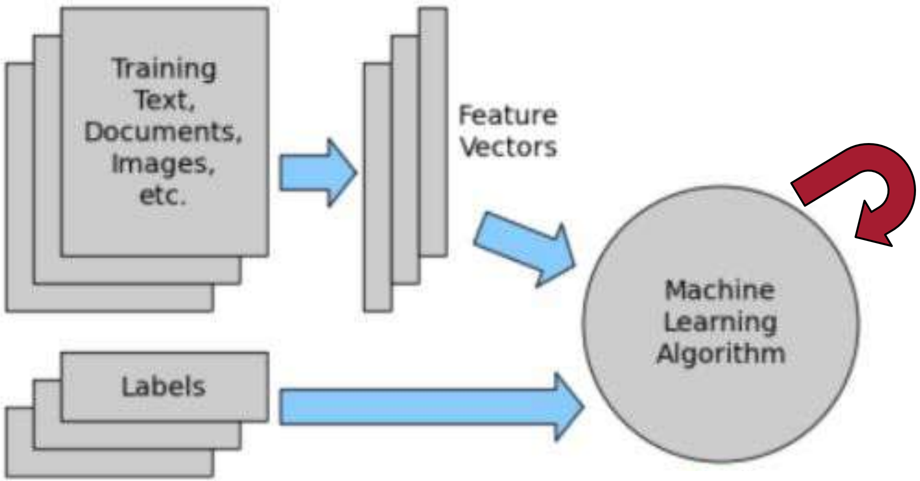
Deep Supervised Learning



3. **Train** your model

Training can take a **LONG time** so we often use the **cloud** to do this!

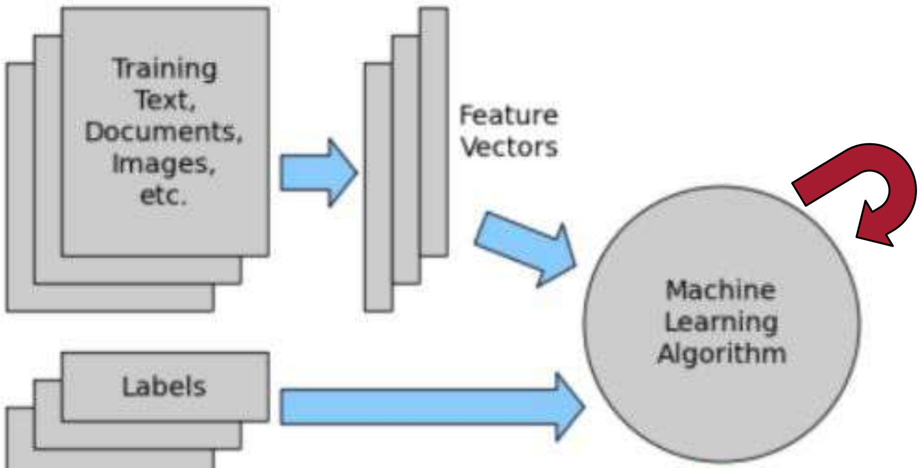
Deep Supervised Learning



3. **Train** your model

What about on device training?

Deep Supervised Learning



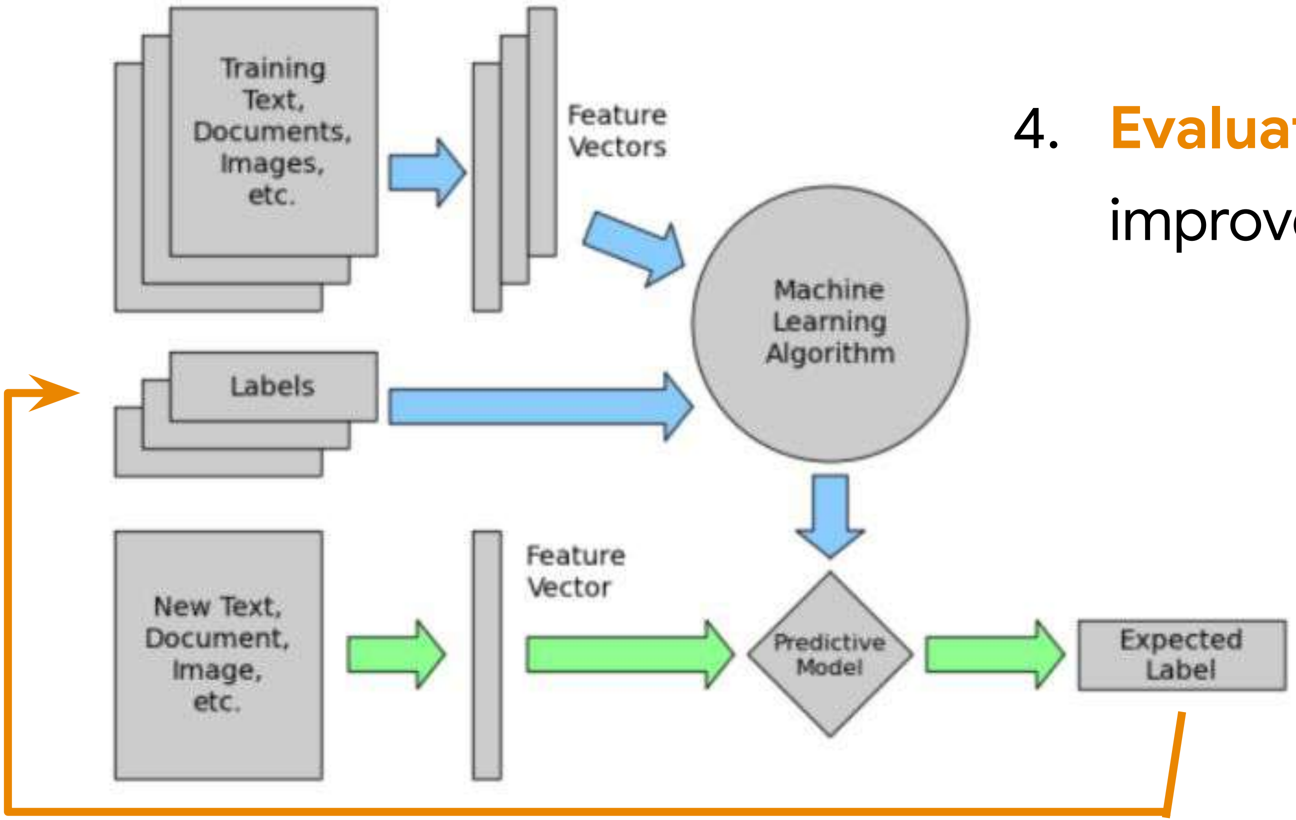
What about on device training?

Hasn't been done much yet with NNs.

Classical Learning (KNN): https://blog.arduino.cc/2020/06/18/simple-machine-learning-with-arduino-knn/

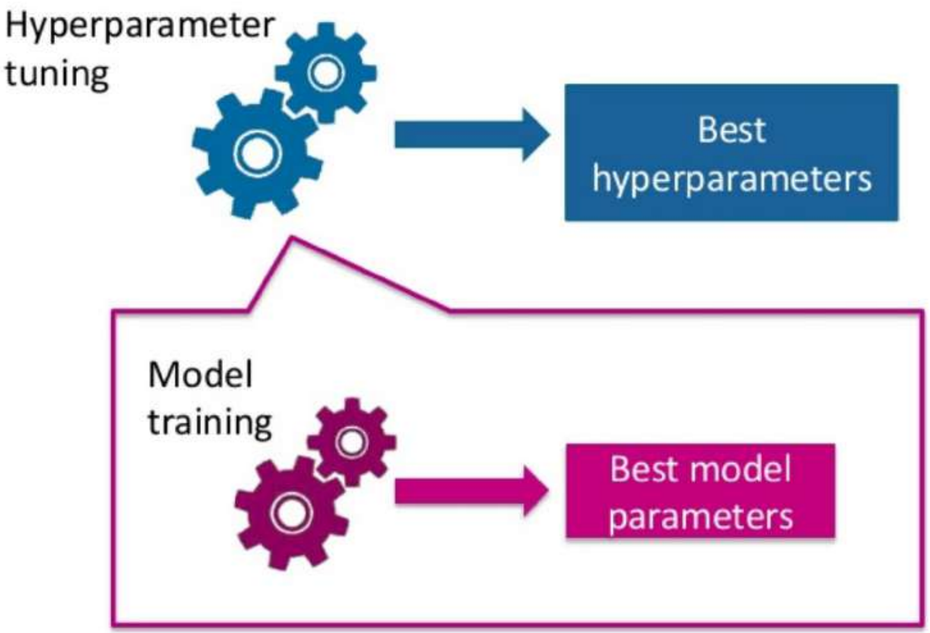
Federated Learning (using the edge devices more as sensors): https://www.researchgate.net/profile/Poonam_Yadav14/publication/341424819_CoLearn_Enabling_Federated_Learning_in_MUD-compliant_IoT_Edge_Networks/links/5ebf7fc5299bf1c09ac0b5dd/CoLearn-Enabling-Federated-Learning-in-MUD-compliant-IoT-Edge-Networks.pdf

Deep Supervised Learning



4. **Evaluate** your model and improve hyper parameters

Deep Supervised Learning



4. **Evaluate** your model and improve hyper parameters

This is a **CRUCIAL** step as models will often only learn for specific ranges of hyperparameters!

Quick Aside:
Hyperparameter Tuning

Manual Search
✓ Can jump to good solutions
✗ Requires a skilled operator and no guarantees

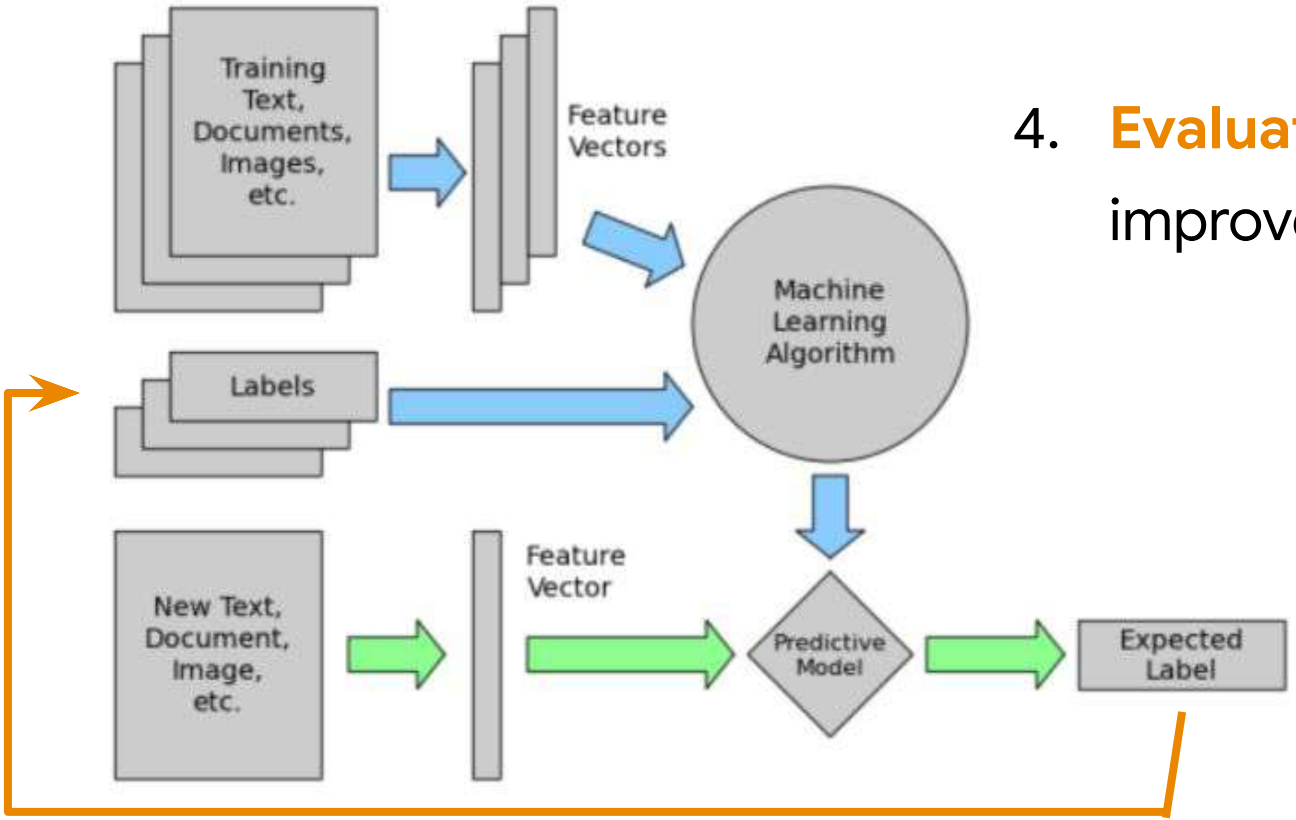
Grid Search
✓ Highly parallel and complete
✗ Very time consuming

Random Search
✓ Often works as well as others
✗ But can't be sure!

Bayesian Optimization
✓ Principled and efficient
✗ Requires a model

https://missinglink.ai/guides/neural-network-concepts/hyperparameters-optimization-methods-and-real-world-model-management/
https://dkopczyk.quantee.co.uk/hyperparameter-optimization/
https://towardsdatascience.com/hyperparameters-optimization-526348bb8e2d

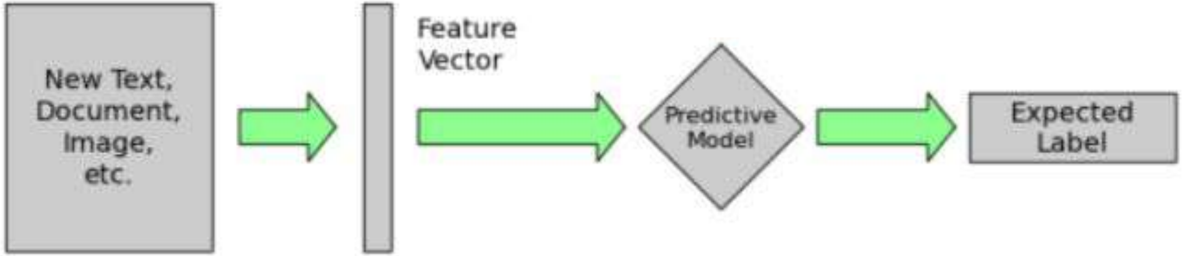
Deep Supervised Learning



4. **Evaluate** your model and improve hyper parameters

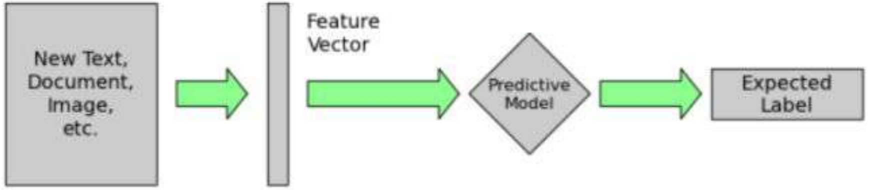
Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model



Deep Supervised Learning

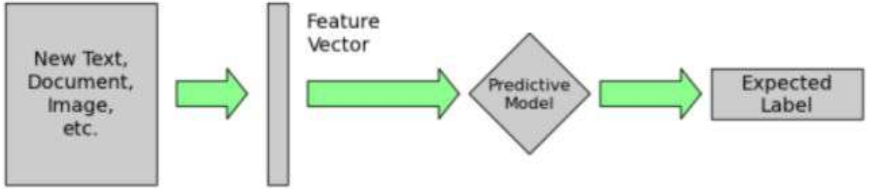
5. **Deploy** and efficient **inference** engine for your model



This is the **(ONLY) ONLINE STEP** all previous steps could have used cloud resources e.g.,

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model



This is the **(ONLY) ONLINE STEP** all previous steps could have used cloud resources e.g.,

So now we have to worry about real-world constraints:

- **Power**
- **Latency**

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

We need a compiler to generate **machine specific** optimized code!

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

We need a compiler to generate **machine specific** optimized code!

https://www.tensorflow.org/

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

TensorFlow

Model Optimization

Scripting Interface | Cloud Serving

Labeling Tools | Data Loading

Training Loop | Variable Storage

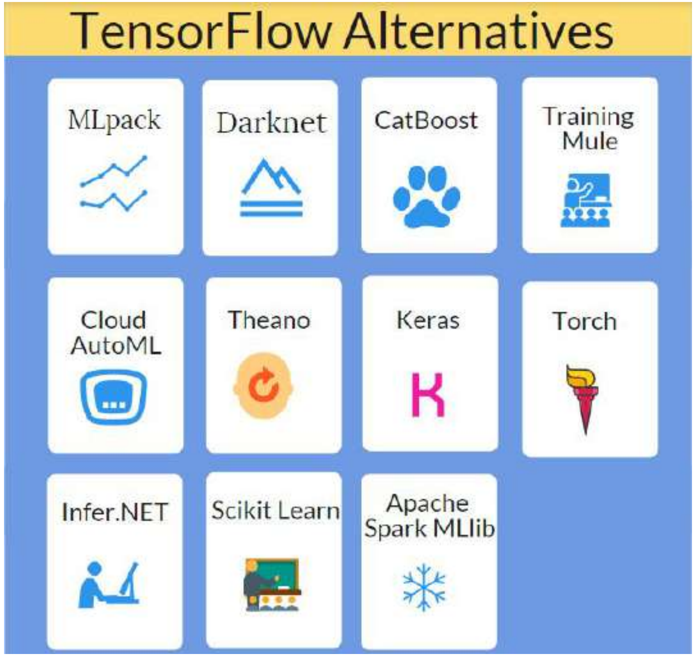
Distributed Compute | Metric Visualizer

Math Library | Feature Generation

https://www.tensorflow.org/
https://www.youtube.com/watch?v=_NcG5estXOU&list=PLtT1eAdRePYoovXJcDkV9RdabZ33H6Di0&index=4

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model



https://www.tensorflow.org/
https://www.educba.com/tensorflow-alternatives/

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

TensorFlow

Model Optimization

Scripting Interface

Cloud Serving

But for efficient inference -- do we need all of this?

Math Library

Feature Generation

https://www.tensorflow.org/
https://www.youtube.com/watch?v=_NcG5estXOU&list=PLtT1eAdRePYoovXJcDkV9RdabZ33H6Di0&index=4

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

TensorFlow

Model Optimization

Scripting Interface | Cloud Serving

Labeling Tools | Data Loading

Training Loop | Variable Storage

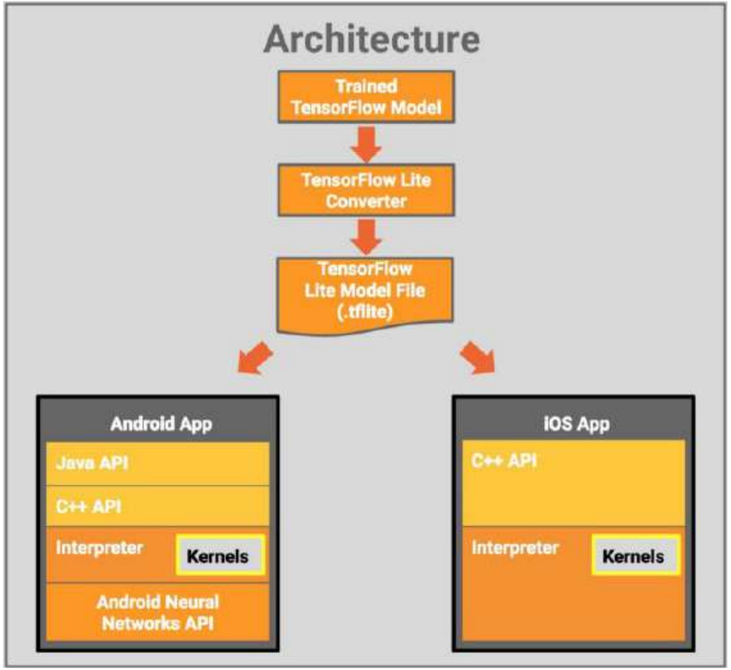
Distributed Compute | Metric Visualizer

Math Library | Feature Generation

https://www.tensorflow.org/
https://www.youtube.com/watch?v=_NcG5estXOU

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model



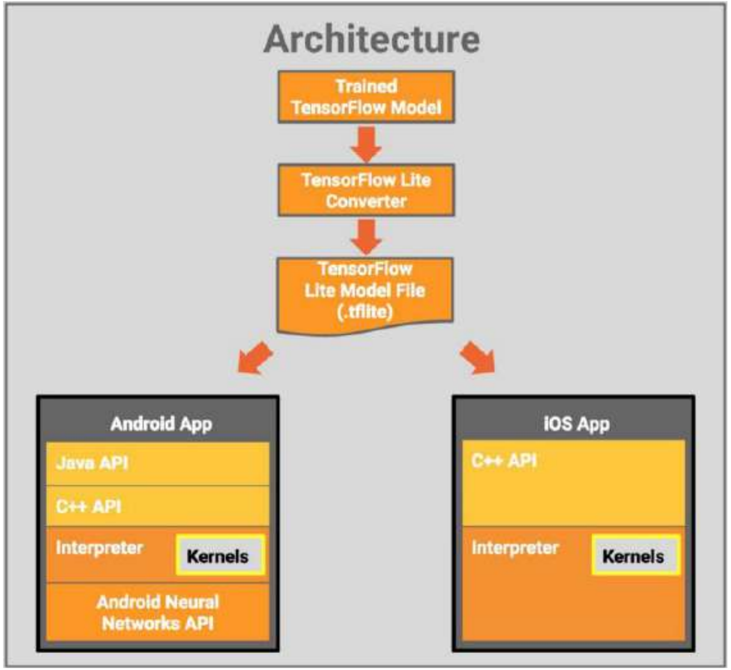
https://www.youtube.com/watch?v=_NcG5estXOU
https://medium.com/tensorflow/using-tensorflow-lite-on-android-9bbc9cb7d69d

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

TensorFlow Lite

O(500Kb)



https://www.youtube.com/watch?v=_NcG5estXOU
https://medium.com/tensorflow/using-tensorflow-lite-on-android-9bbc9cb7d69d

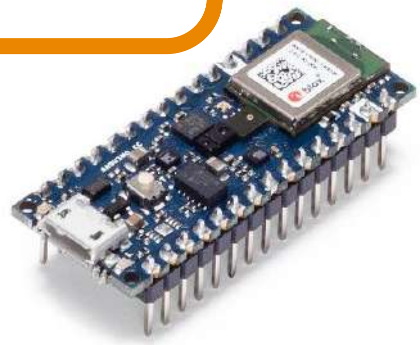
Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

TensorFlow Lite

O(500Kb)

Our board only has 256Kb of RAM!



https://www.youtube.com/watch?v=_NcG5estXOU
https://medium.com/tensorflow/using-tensorflow-lite-on-android-9bbc9cb7d69d

Deep Supervised Learning

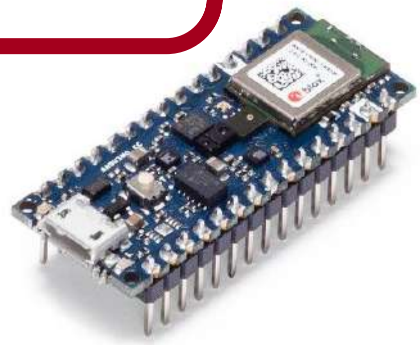
5. **Deploy** and efficient **inference** engine for your model

TensorFlow Lite

MICRO

~20KB

Only deploy what you need!



https://www.youtube.com/watch?v=_NcG5estXOU
https://medium.com/tensorflow/using-tensorflow-lite-on-android-9bbc9cb7d69d

Deep Supervised Learning

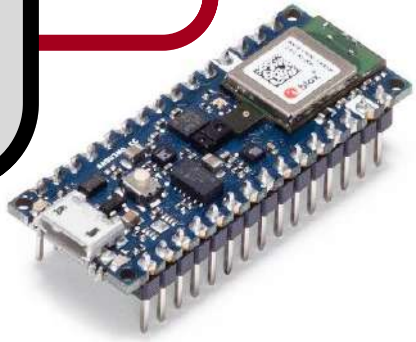
5. **Deploy** and efficient **inference** engine for your model

At the same time hardware researchers have been developing **NN accelerators**

oy
eed!

Tenso

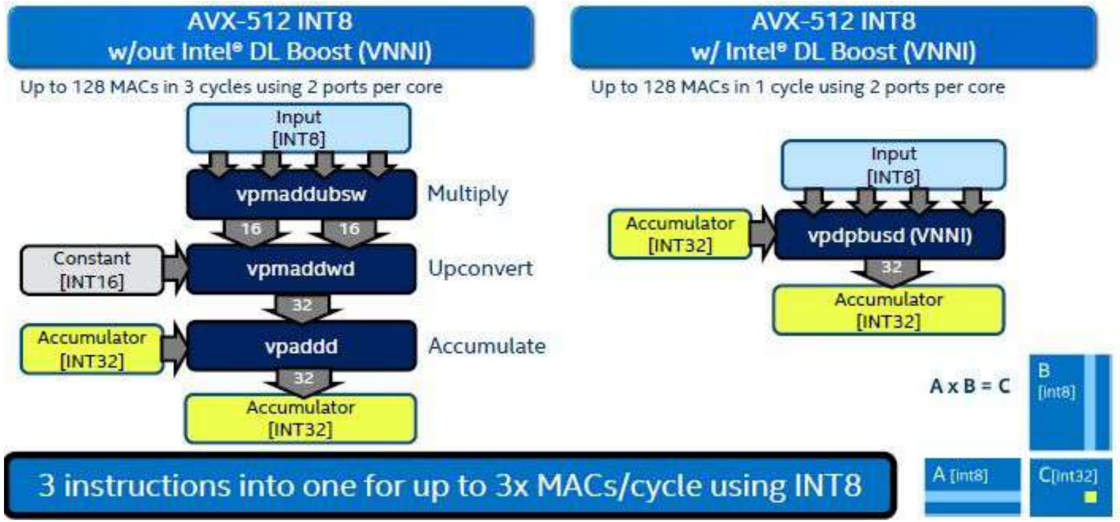
MICRO

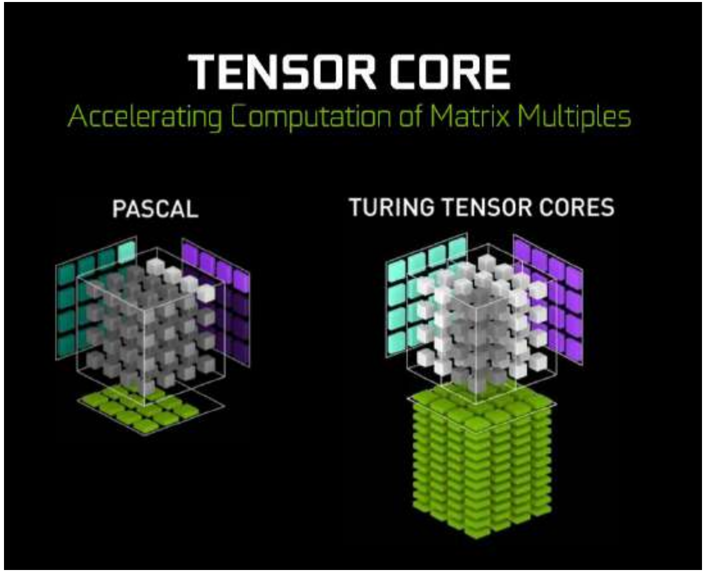


https://www.youtube.com/watch?v=_NcG5estXOU
https://medium.com/tensorflow/using-tensorflow-lite-on-android-9bbc9cb7d69d

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model

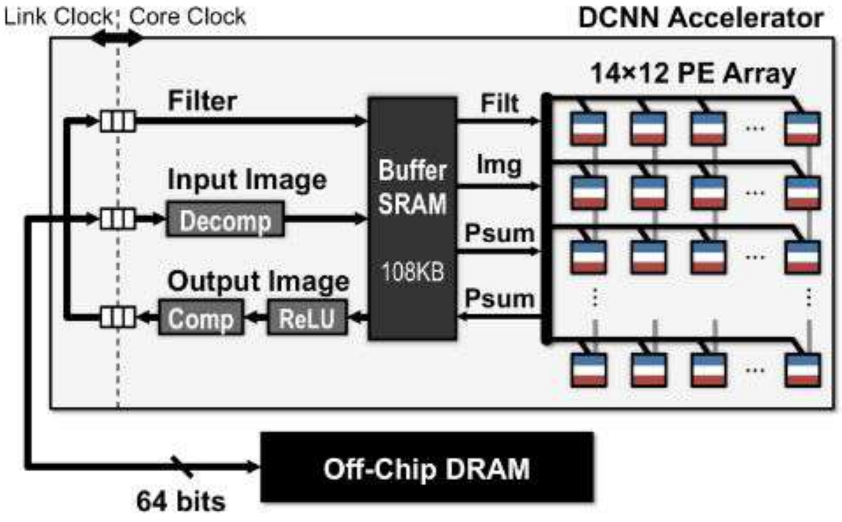




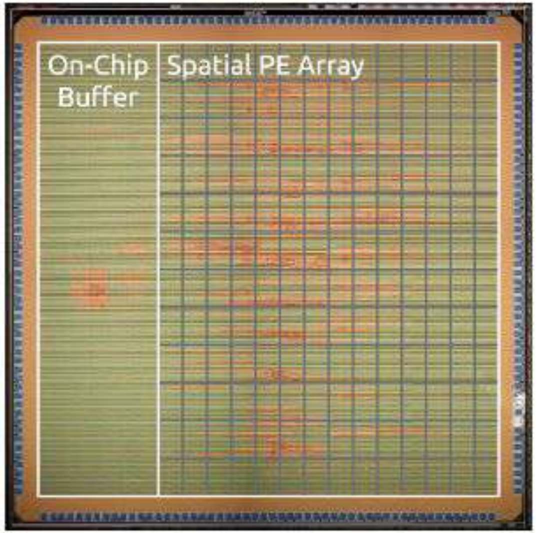
https://hexus.net/tech/reviews/graphics/122045-nvidia-turing-architecture-examined-and-explained/?page=4
https://www.nextplatform.com/2019/04/15/the-long-view-on-the-intel-xeon-architecture/

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model



Eyeriss Architecture

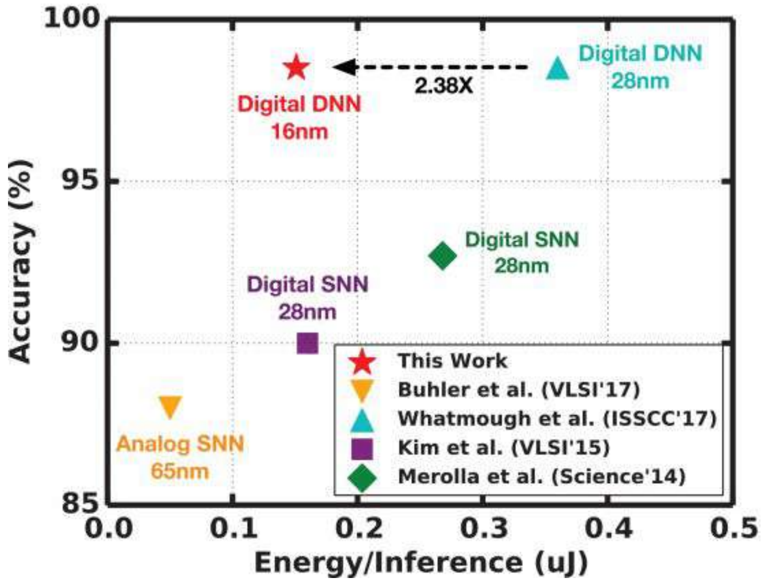


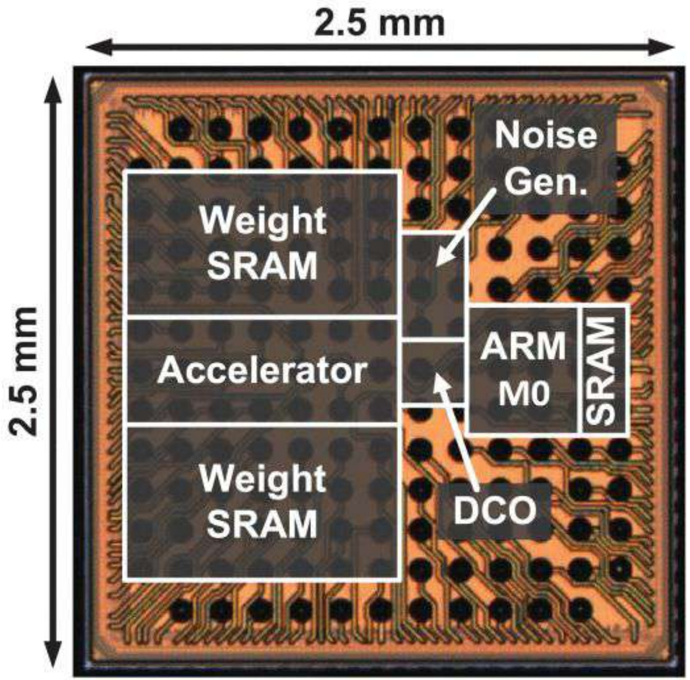
Die Photo

https://eyeriss.mit.edu/

Deep Supervised Learning

5. **Deploy** and efficient **inference** engine for your model





https://ieeexplore.ieee.org/document/8715387

Deep Supervised Learning

1. **Collect** LOTS of (unbiased) data!
2. **Preprocess** the data and **design** your model
3. **Train** your model (in the cloud)
4. **Evaluate** your model and improve hyper parameters
5. **Deploy** and efficient inference engine for your model

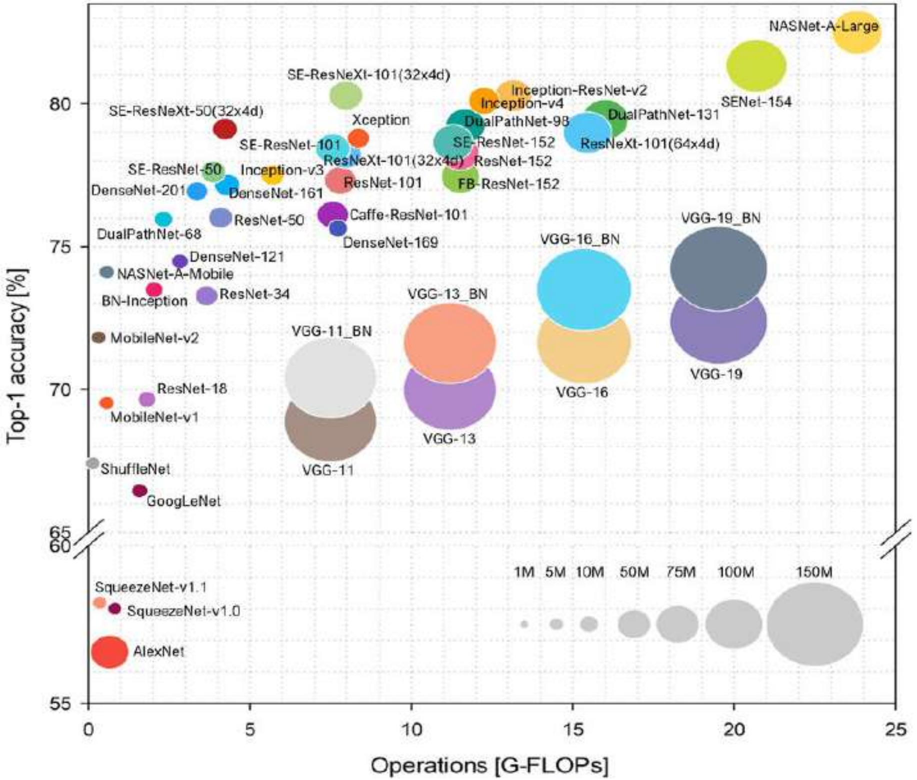
Deep Supervised Learning

1. **Collect** LOTS of (unbiased) data!
2. **Preprocess** the data and **design** your model
3. **Train** your model (in the cloud)
4. **Evaluate** your model and improve hyper parameters
5. **Deploy** and efficient inference engine for your model

Use a machine learning framework (e.g., TensorFlow)

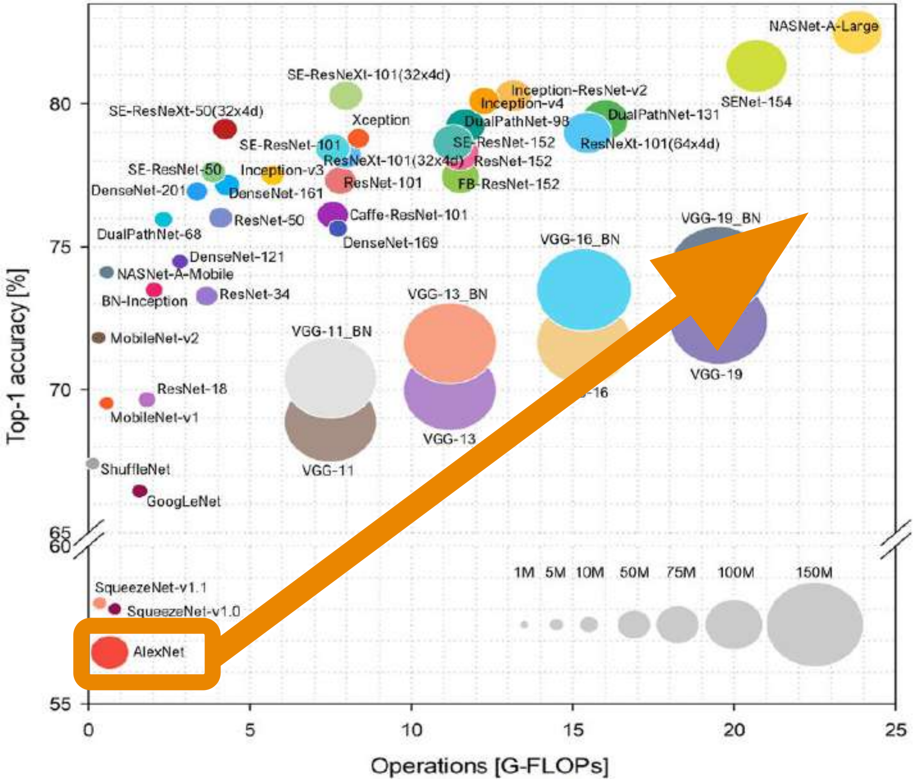
From Deep Learning to TinyML

Models come in all shapes and sizes



[Bianco et. al. Benchmark Analysis of Representative Deep Neural Network Architectures]

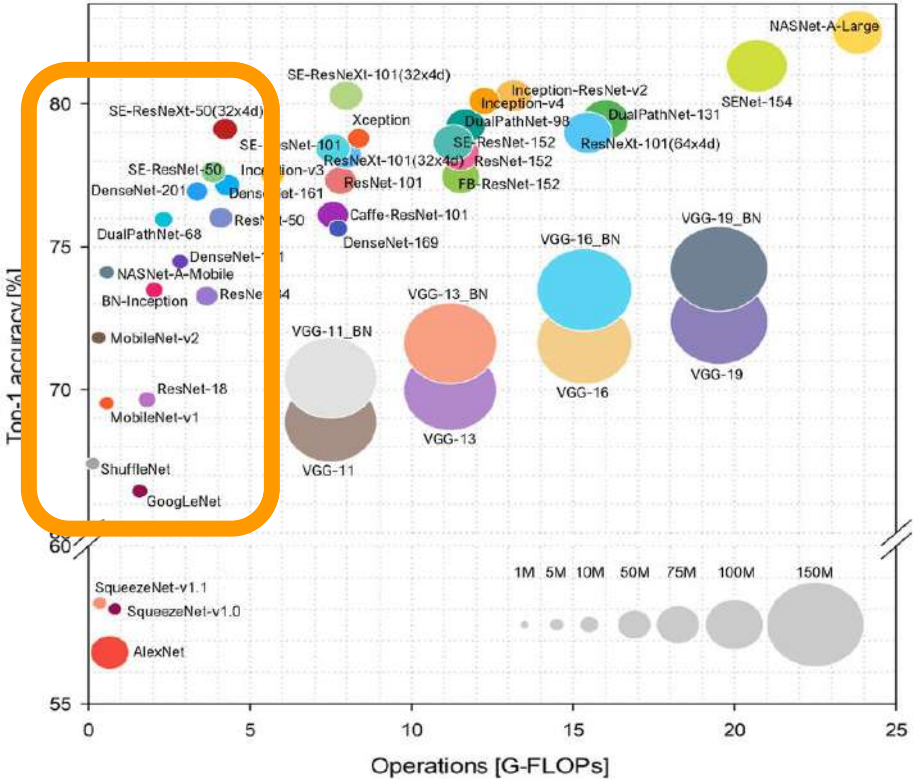
Models come in all shapes and sizes



Originally models grew in size and operations

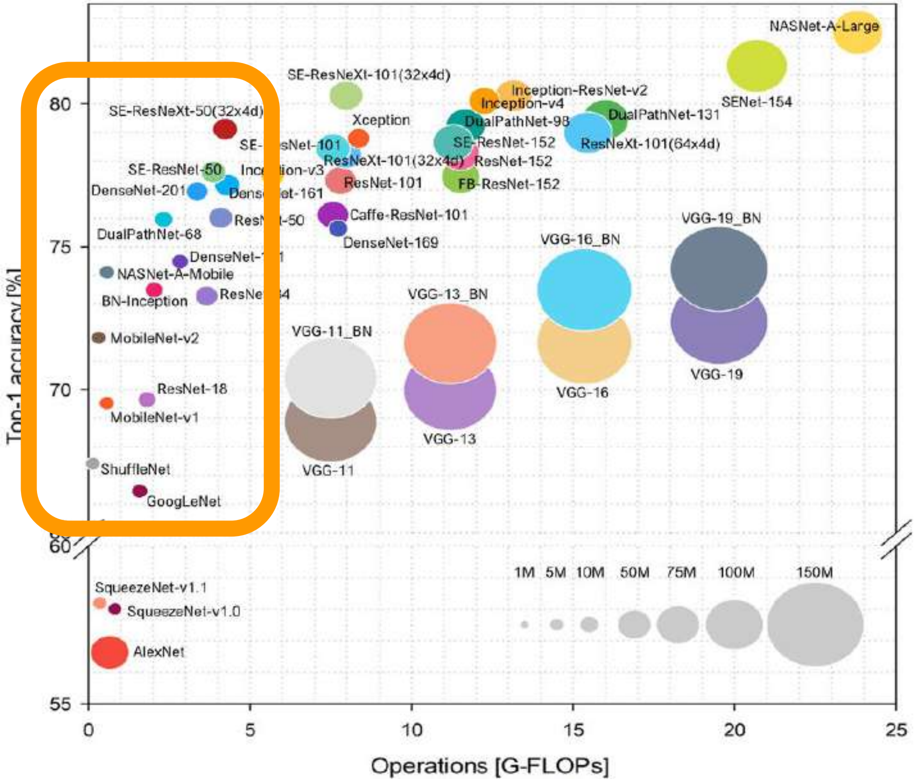
NOTE HOW SMALL AND CHEAP (in comparison) ALEXNET IS!!!!!!!!!!

Models come in all shapes and sizes



New smaller and easier to compute models have been developed that are **still very accurate**

Models come in all shapes and sizes



New smaller and easier to compute models have been developed that are still very accurate

They were designed to **target mobile devices**

How tiny is Tiny?

Table 4: Memory of CNN models on platforms (MB)

Type/Platform	AlexNet	VGGNet	GoogleNet	ResNet
Weights & Biases	233	528	26	97
Data	8	110	53	221
Workspace	11	168	46	79

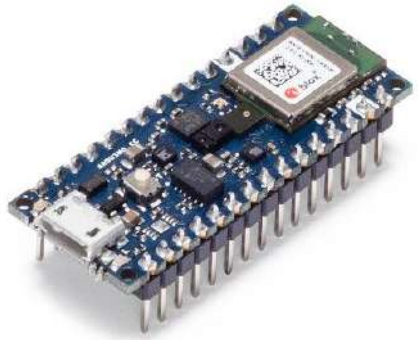
https://arxiv.org/pdf/1709.09503.pdf

How tiny is Tiny?

Table 4: Memory of CNN models on platforms (MB)

Type/Platform	AlexNet	VGGNet	GoogleNet	ResNet
Weights & Biases	233	528	26	97
Data	8	110	53	221
Workspace	11	168	46	79

Our board only has 256Kb of RAM!



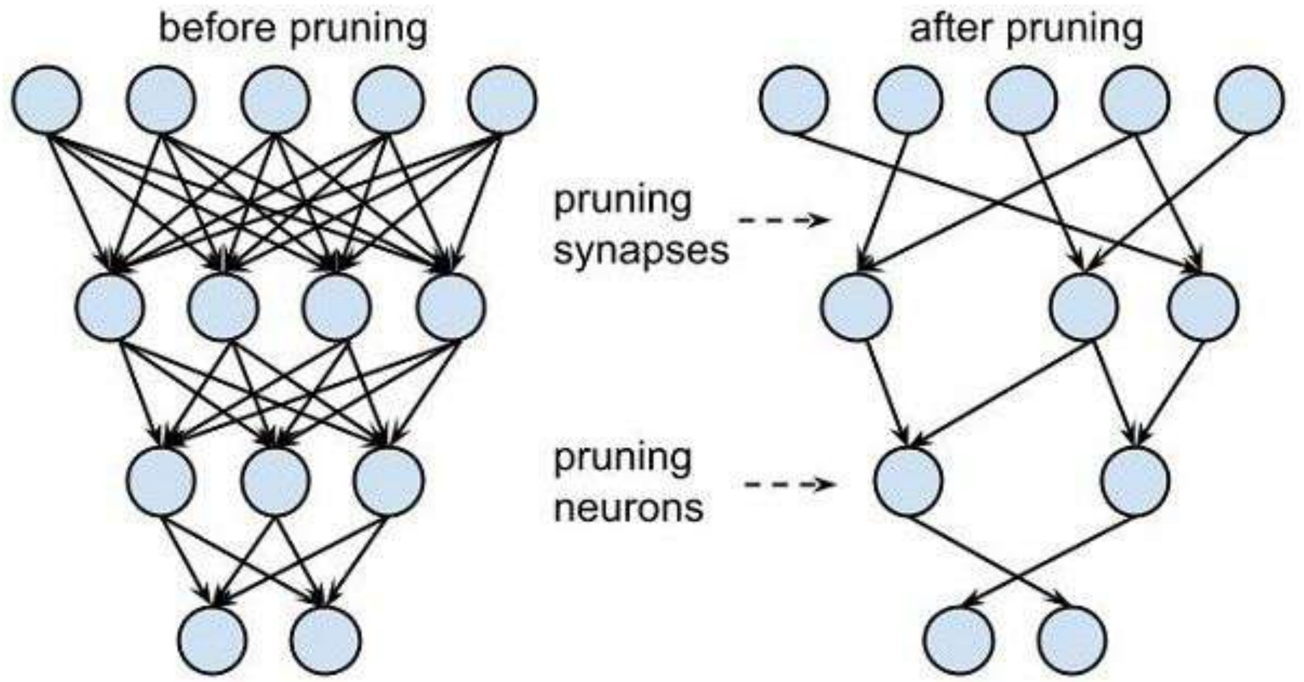
https://arxiv.org/pdf/1709.09503.pdf

How can we compress things further?

How can we compress things further?

Pruning and **Quantization** to the rescue!

Pruning removes the least important "stuff" from the model



Pruning removes the least important "stuff" from the model

Table 2: MobileNet sparse vs dense results

Width	Sparsity	NNZ params	Top-1 acc.	Top-5 acc.
1.0	0%	4.21M	70.6%	89.5%
	50%	2.13M	69.5%	89.5%
	75%	1.09M	67.7%	88.5%
	90%	0.46M	61.8%	84.7%
	95%	0.25M	53.6%	78.9%

Very small accuracy penalty!

https://openreview.net/forum?id=S1lN69AT-

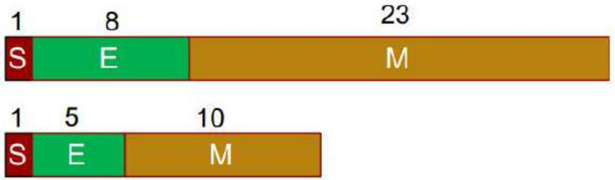
Quantization compresses the numerical representation

	Range	Accuracy
FP32	10^{-38} - 10^{38}	.000006%
FP16	$6x10^{-5}$ - $6x10^{4}$	.05%

Reduced Size
Reduced Precision

https://media.nips.cc/Conferences/2015/tutorialslides/Dally-NIPS-Tutorial-2015.pdf

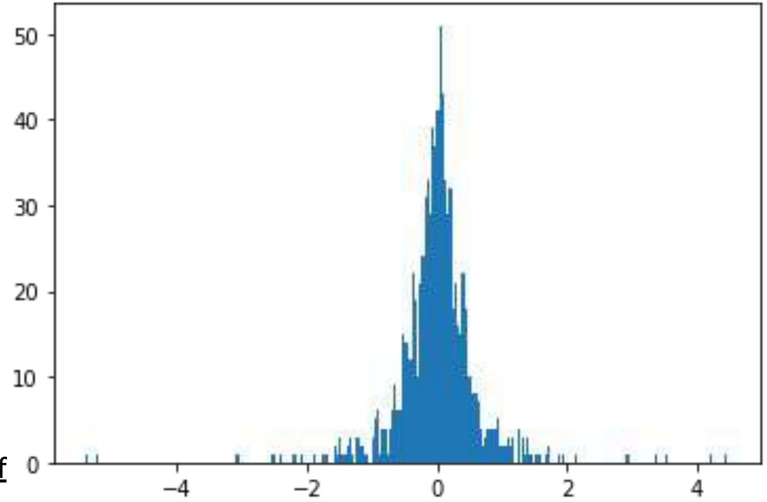
Quantization compresses the numerical representation



	Range	Accuracy
FP32	10^{-38} - 10^{38}	.000006%
FP16	$6x10^{-5}$ - $6x10^{4}$	.05%

Turns out weights are often even more densely packed than that!

https://drive.google.com/file/d/1Pn-INOty-P1fX6G66oa6wvsEn46Xyx8Y



https://media.nips.cc/Conferences/2015/tutorialslides/Dally-NIPS-Tutorial-2015.pdf

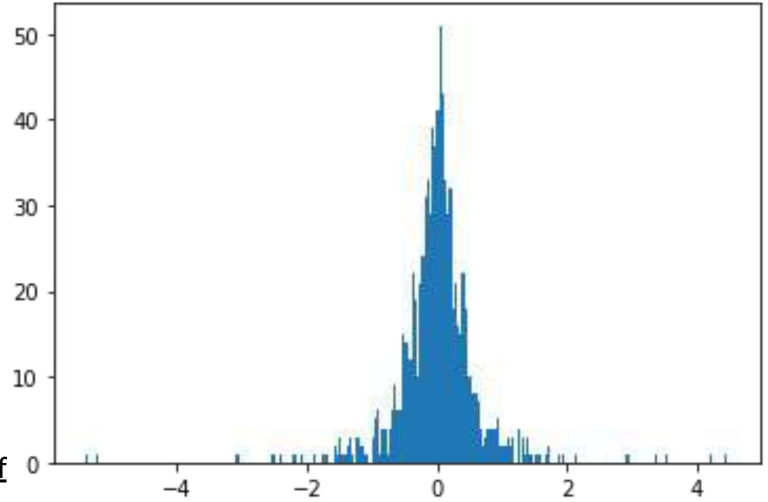
Quantization compresses the numerical representation

	Range	Accuracy
FP32	10^{-38} - 10^{38}	.000006%
FP16	$6x10^{-5}$ - $6x10^{4}$	.05%

So can we do better?

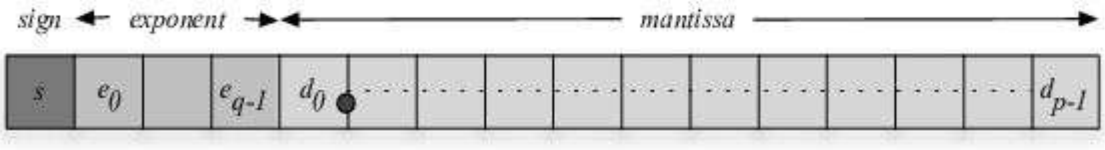
Turns out weights are often even more densely packed than that!

https://drive.google.com/file/d/1Pn-INOty-P1fX6G66oa6wvsEn46Xyx8Y



https://media.nips.cc/Conferences/2015/tutorialslides/Dally-NIPS-Tutorial-2015.pdf

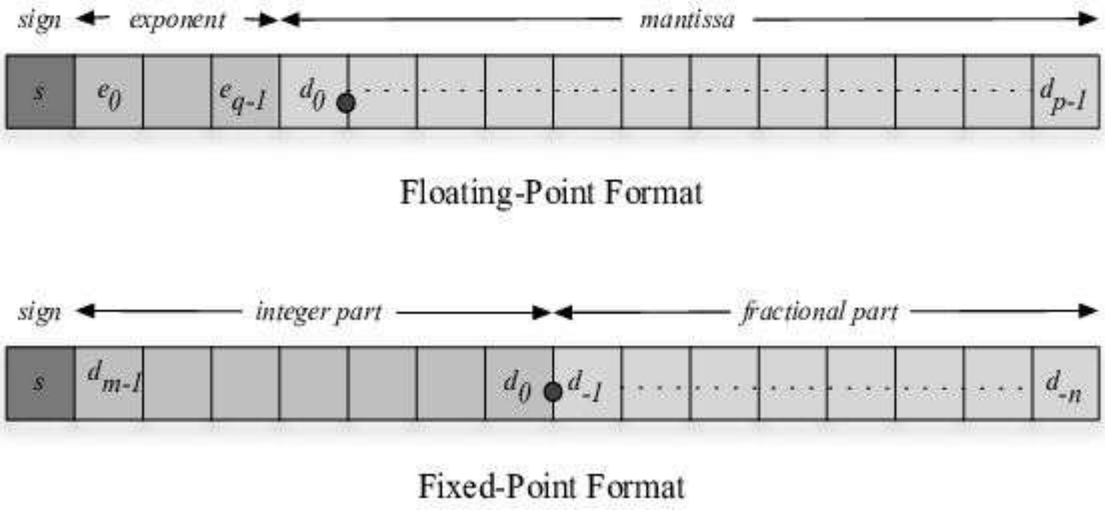
Quantization compresses the numerical representation



Floating-Point Format

mantissa * $10^{exponent}$

Quantization compresses the numerical representation

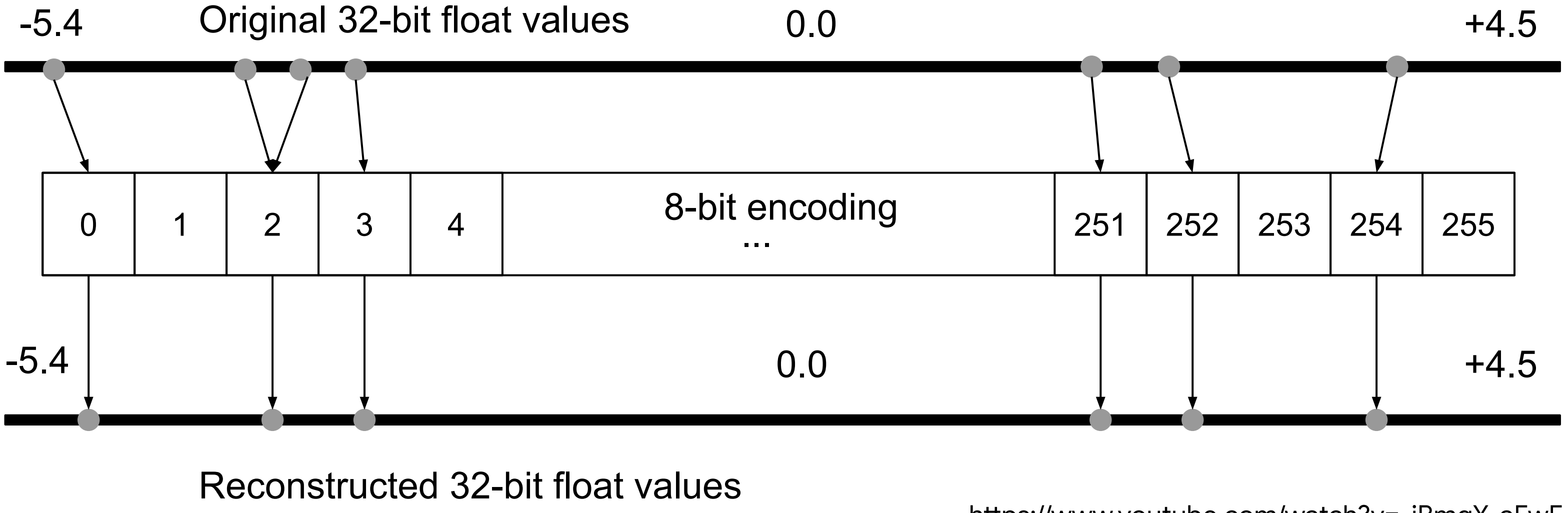


mantissa * $10^{exponent}$

Reduced Range
Reduced Complexity
We can use INTs
We can tune the size

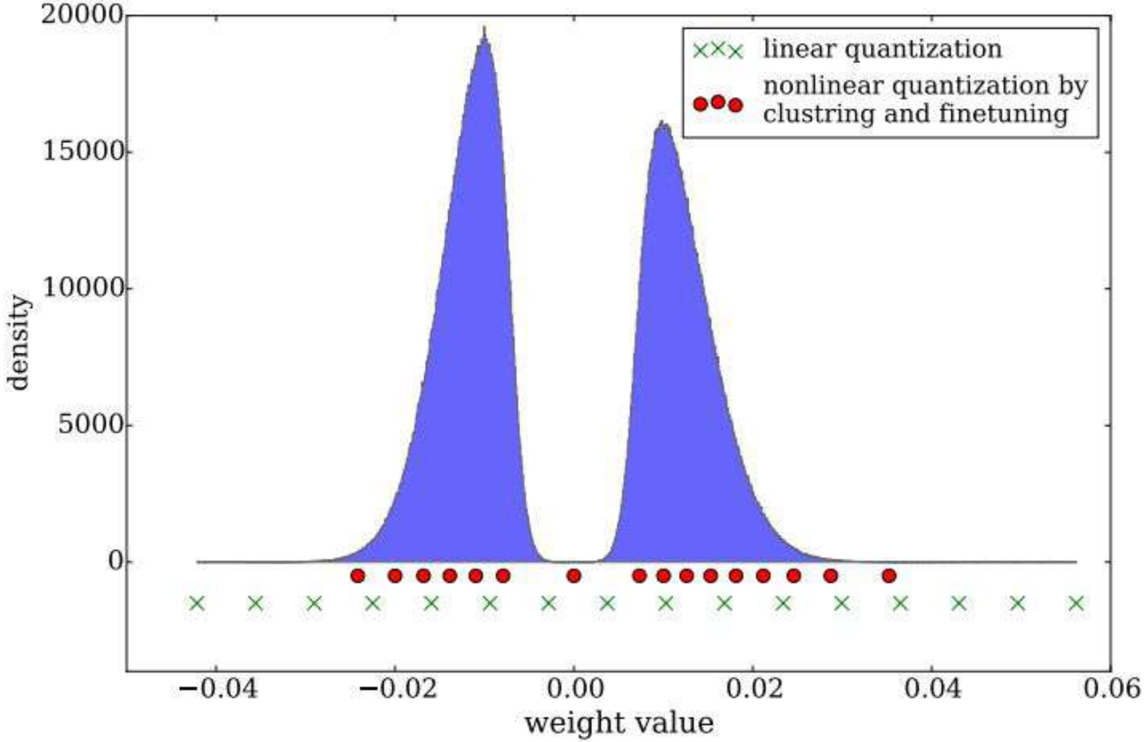
Quantization needs to be **optimized for each model**!

We can do even better with a **linear encoding** of just the range we need!



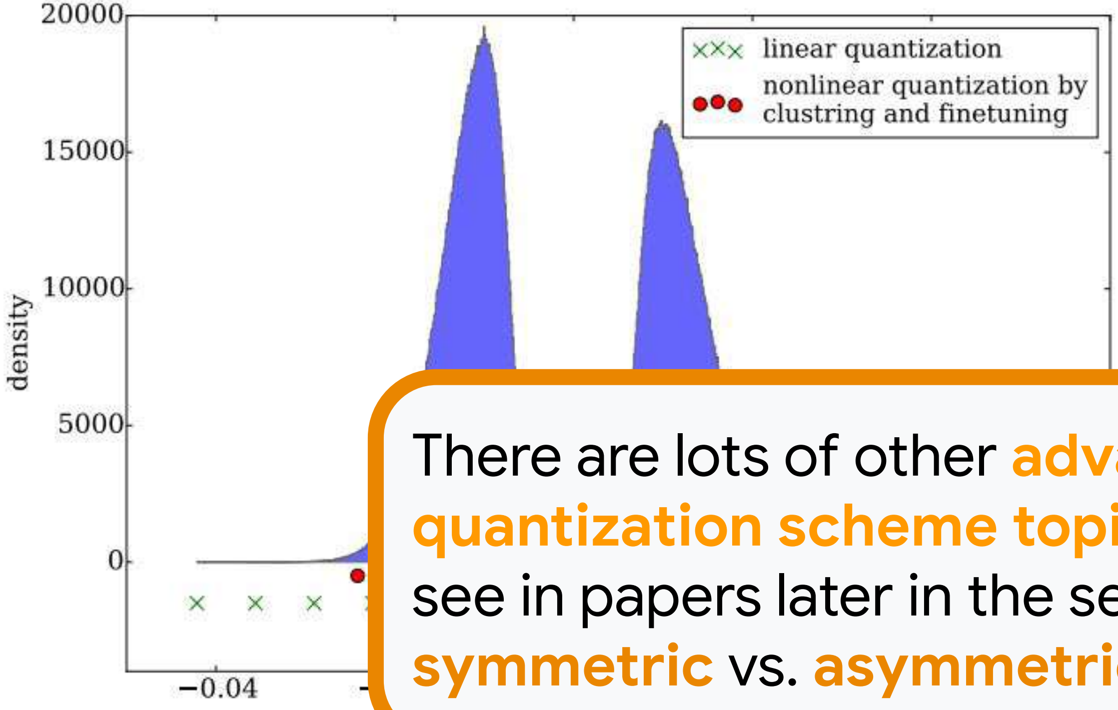
https://www.youtube.com/watch?v=-jBmqY_aFwE&feature=youtu.be

Quantization needs to be **optimized for each model**!



https://arxiv.org/pdf/1510.00149.pdf

Quantization needs to be **optimized for each model**!

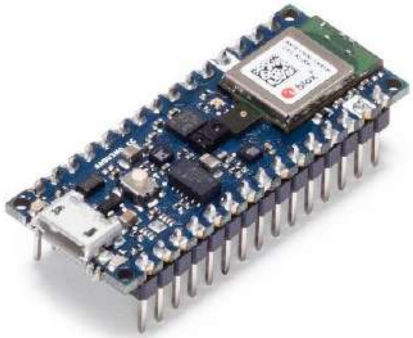


There are lots of other **advanced quantization scheme topics** that we may see in papers later in the semester (e.g., **symmetric** vs. **asymmetric**, **zero point**)

https://arxiv.org/pdf/1510.00149.pdf

Quantization, does it work?

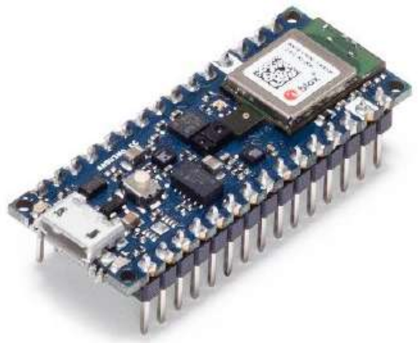
Our board only has 256Kb of RAM so compressing the model is crucial!



E.g., in the Wake Words Assignment the **quantization reduces the model size by 4x (Float32 -> INT8)**

Quantization, does it work?

Our board only has 256Kb of RAM so compressing the model is crucial!

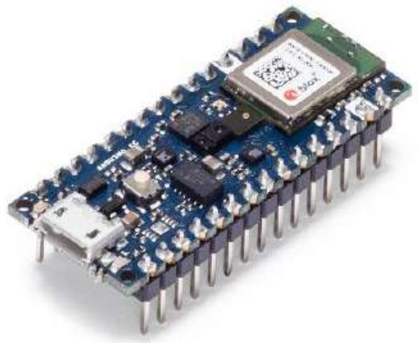


E.g., in the Wake Words Assignment the **quantization reduces the model size by 4x (Float32 -> INT8)**

But is there an accuracy penalty?

Quantization, does it work?

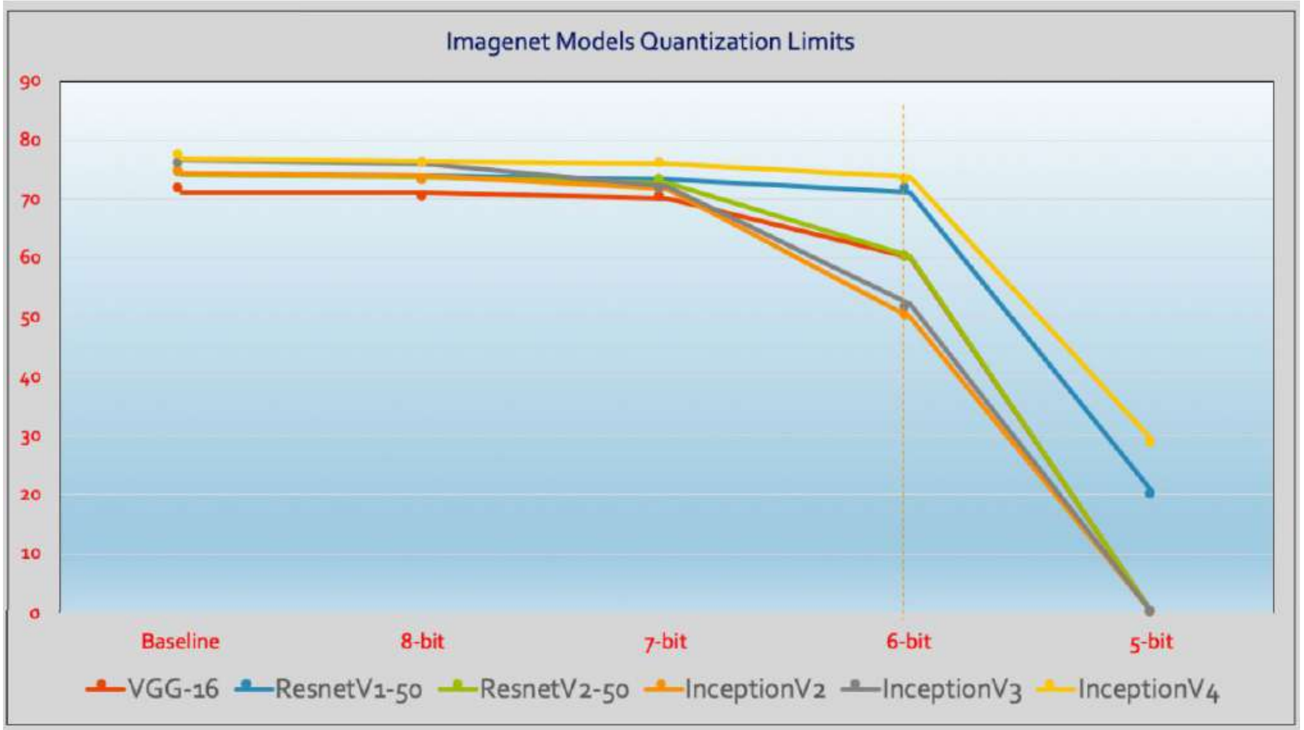
Our board only has 256Kb of RAM so compressing the model is crucial!



E.g., in the Wake Words Assignment the **quantization reduces the model size by 4x (Float32 -> INT8)**

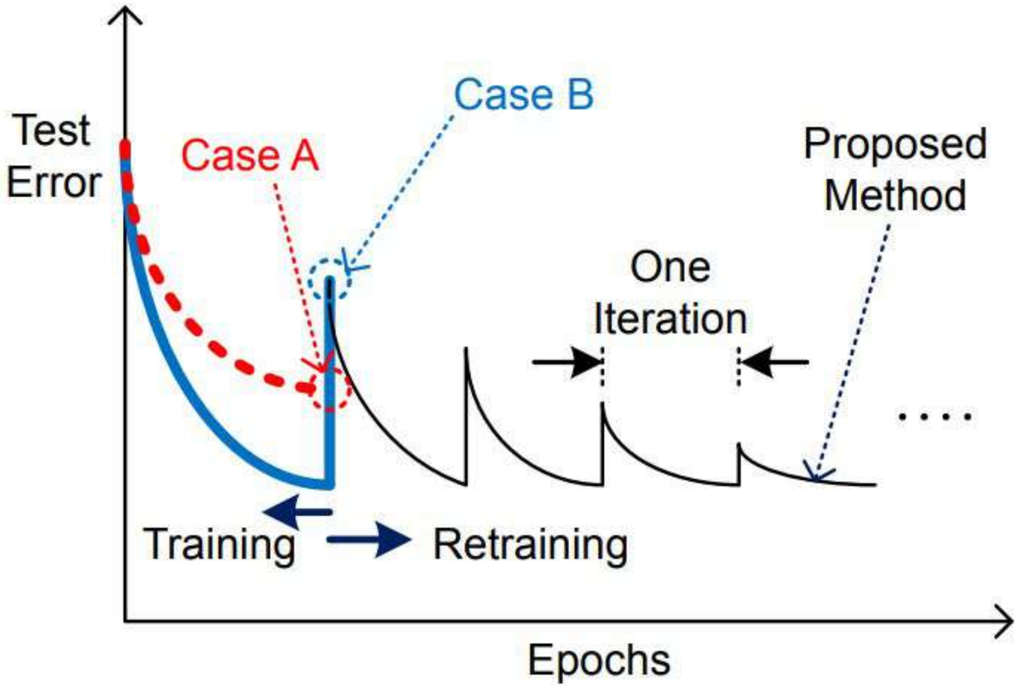
For Wake Words it actually improves!
91.91% to 91.99%

Quantization, does it work?



https://arxiv.org/pdf/1910.04877.pdf

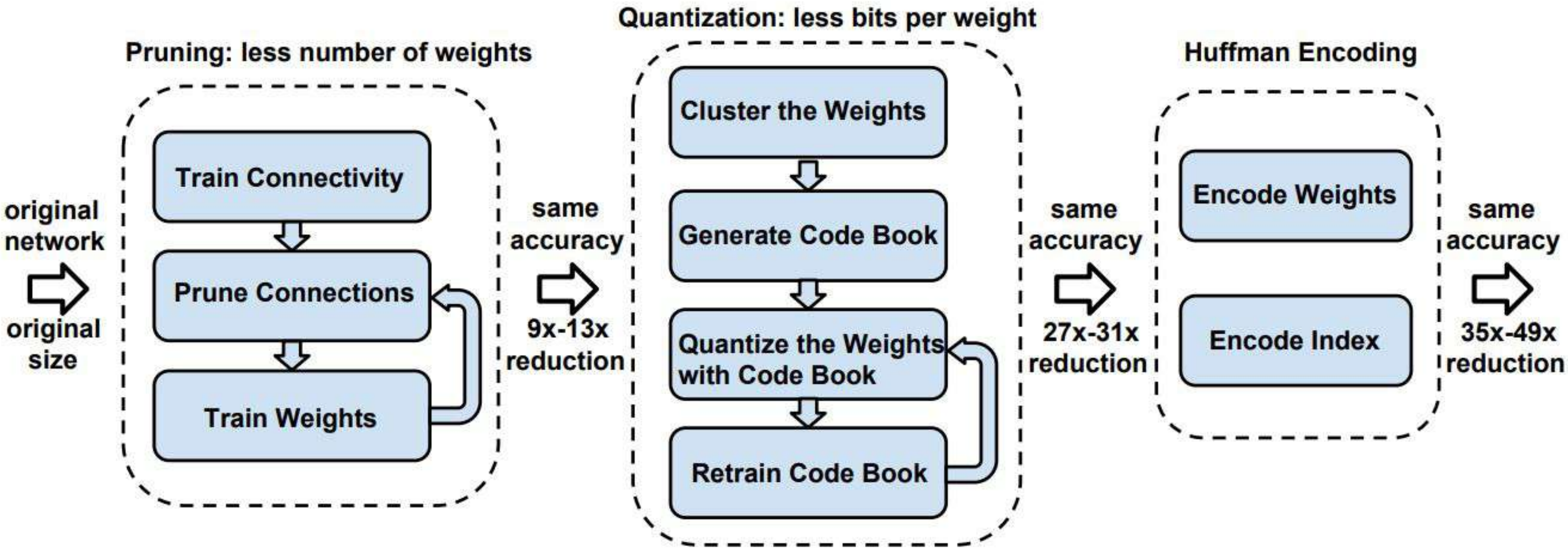
Quantization needs to be **optimized for each model**!



https://arxiv.org/pdf/1805.11233.pdf

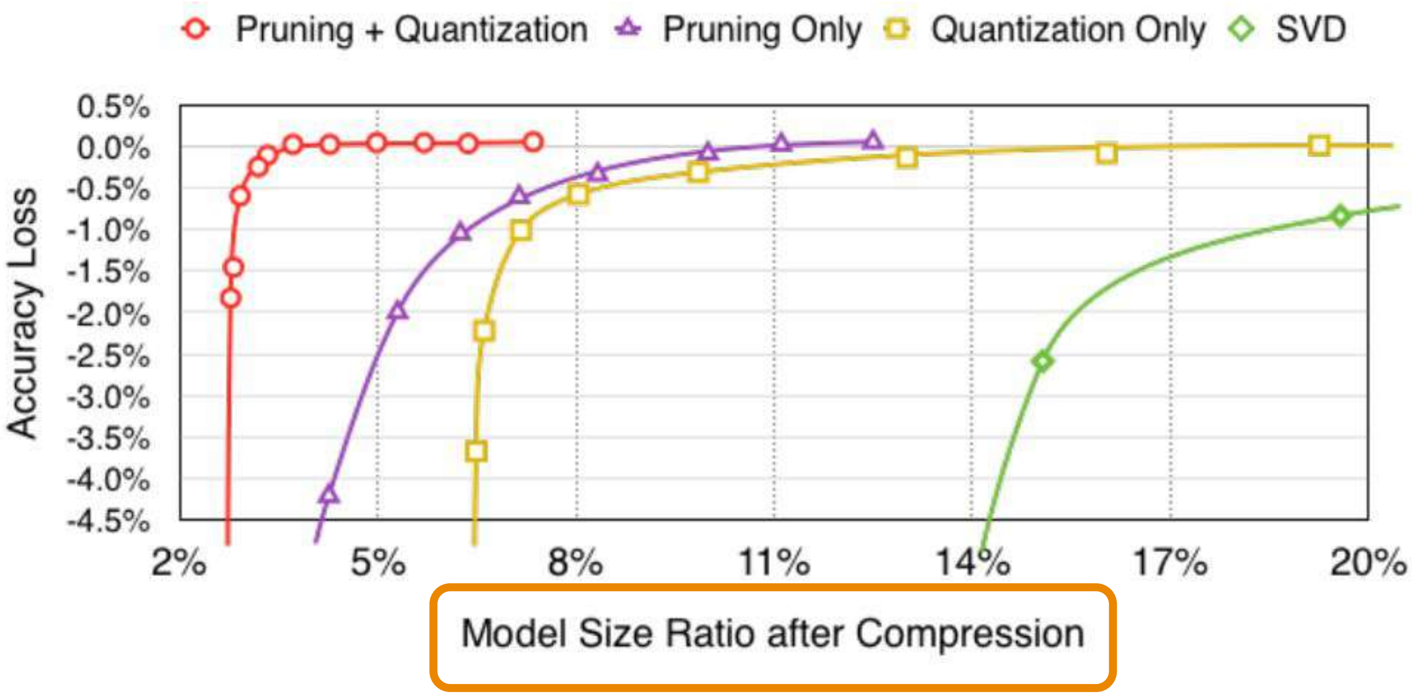
Retraining improves accuracy after quantization

Quantization needs to be **optimized for each model**!



https://arxiv.org/pdf/1510.00149.pdf

Pruning and Quantization to the rescue!

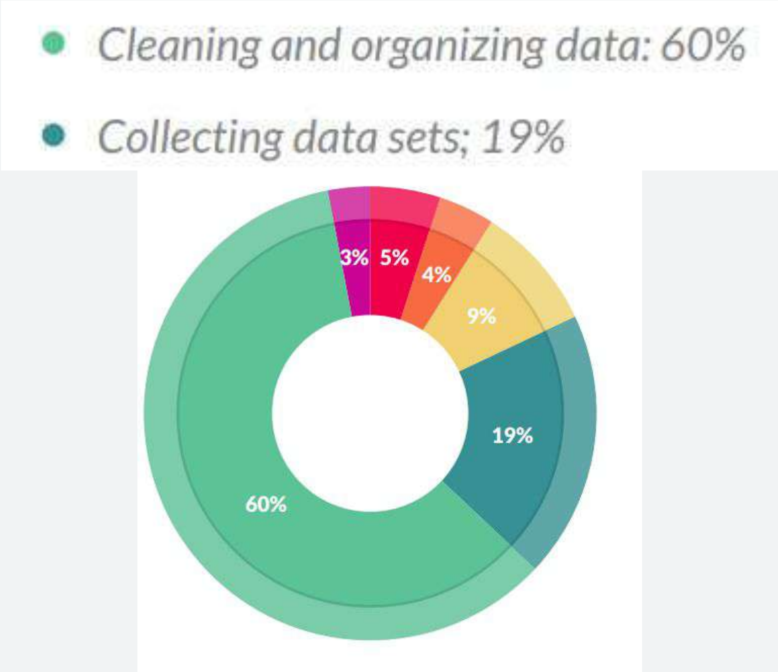


https://arxiv.org/pdf/1510.00149.pdf

And that’s all folks!

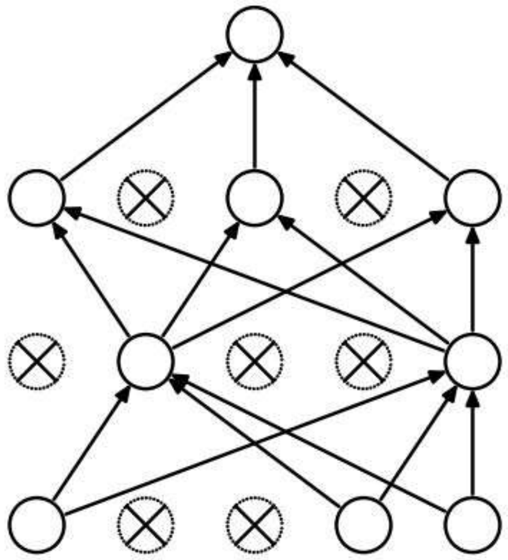
Quick Summary:

- The Supervised Deep Learning flow in practice is: **Collect, Preporcess/Design, Train, Evaluate,** and **Deploy**



Quick Summary:

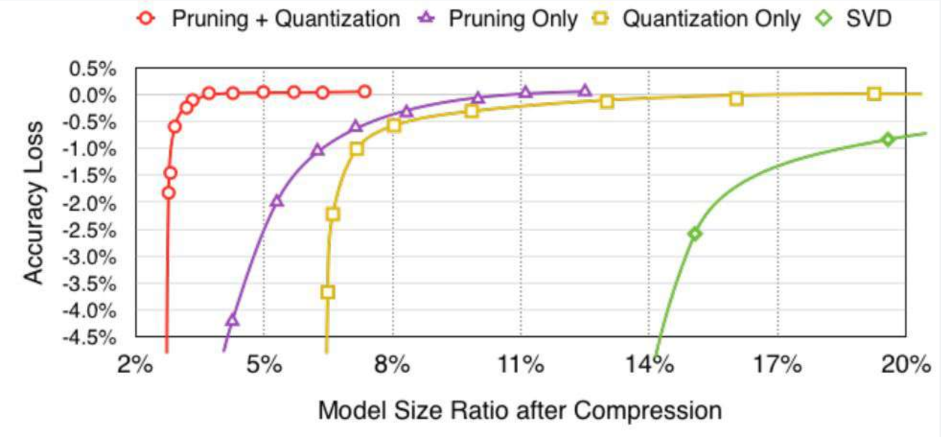
- The Supervised Deep Learning flow in practice is: Collect, Preporcess/Design, Train, Evaluate, and Deploy
- Effective learning requires **lots of unbiased data**, regularization (**Dropout**, **Data Augmentation**), efficient training (**Batch Updates**, **Adaptive Learning Rates**), and hyperparameter tuning



(b) After applying dropout.

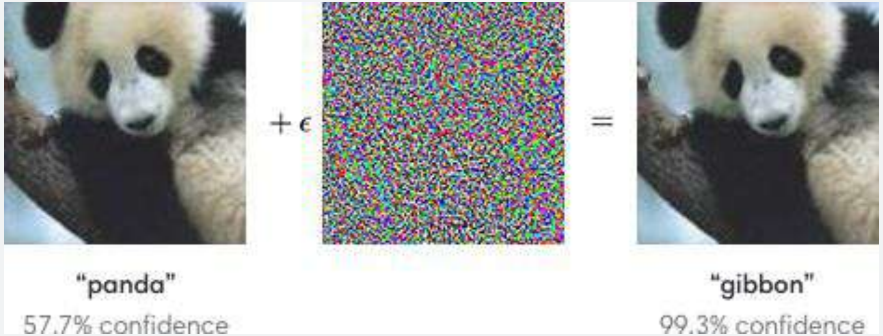
Quick Summary:

- The Supervised Deep Learning flow in practice is: Collect, Preporcess/Design, Train, Evaluate, and Deploy
- Effective learning requires lots of unbiased data, regularization (Dropout, Data Augmentation), efficient training (Batch Updates, Adaptive Learning Rates), and hyperparameter tuning
- **TinyML** is enabled by **inference optimizations** including **pruning** and **quantization**



Quick Summary:

- The Supervised Deep Learning flow in practice is: Collect, Preporcess/Design, Train, Evaluate, and Deploy
- Effective learning requires lots of unbiased data, regularization (Dropout, Data Augmentation), efficient training (Batch Updates, Adaptive Learning Rates), and hyperparameter tuning
- TinyML is enabled by inference optimizations including pruning and quantization
- ML practitioners need to consider the **ethics** and **security** of their applications



Please fill out the feedback poll!