

What is Machine Learning?

KCDC

August 3, 2017

David W. Body / Big Creek Software, LLC



@david_body

KCDC sponsors

TITANIUM SPONSORS



Platinum Sponsors



Gold Sponsors





Artificial Intelligence \supseteq Machine Learning \supseteq Deep Learning

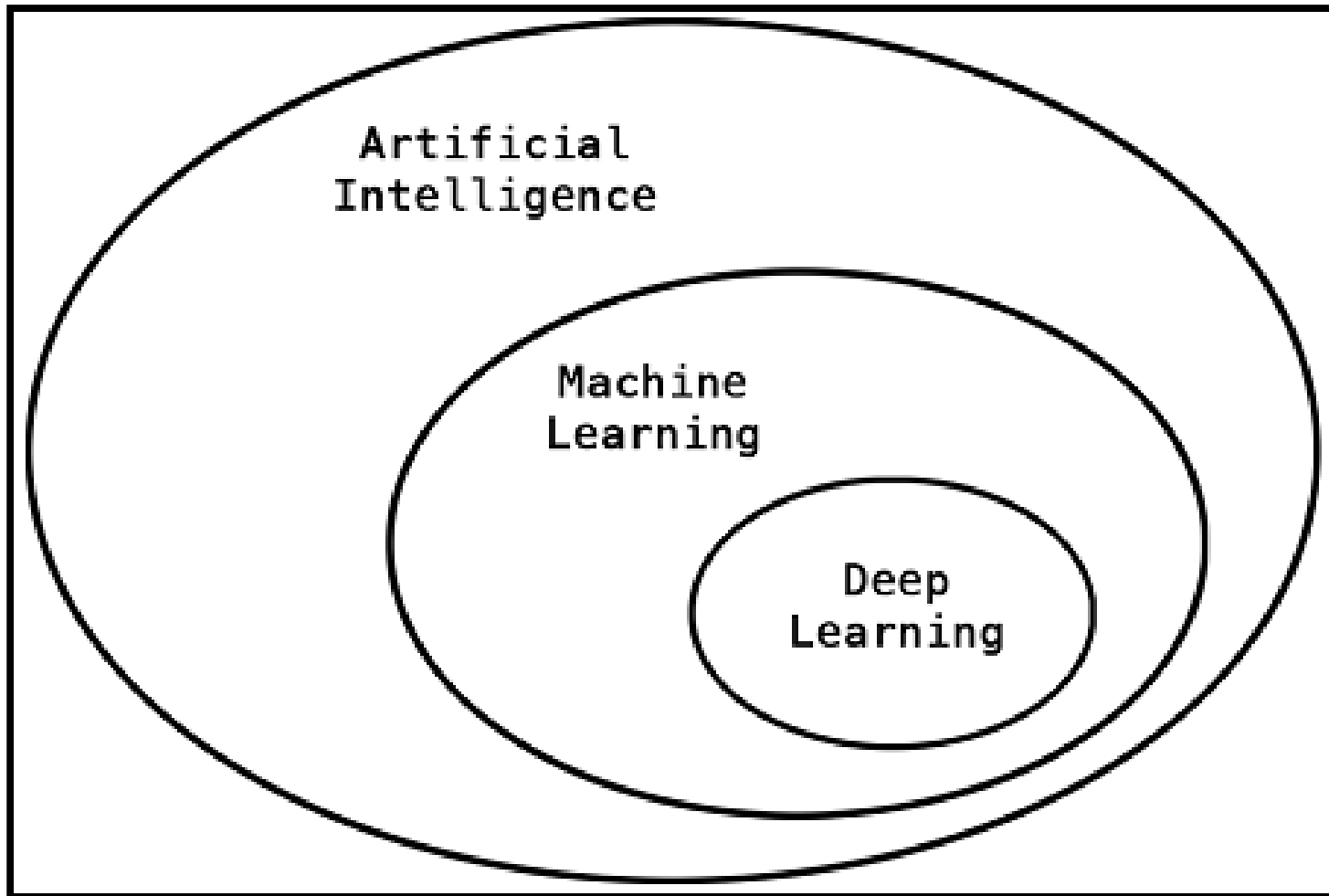


Image credit: François Chollet, Deep Learning with Python

Who am I?

- David W. Body
- Independent software developer
- Love learning & teaching
- Avid runner & fitness seeker
- Available for contract & consulting work

Who are you?

- Software developers
- Data scientists / statisticians?
- Other?



Trask

@iamtrask



Follow

if you're new to Deep Learning, be encouraged. Each part you need to learn is learnable and nobody knows it all. It's a [#lifelong](#) journey.

3:37 PM - 2 Apr 2017



83



236



Outline

- The concept of machine learning
- Supervised vs. unsupervised machine learning
- Types of machine learning problems
- Machine learning techniques
 - Linear regression
 - Logistic regression
 - Neural networks
- MNIST example
- Dangers of machine learning
- Where to go from here

In 1959 Arthur Samuel wrote a computer program to play checkers. Over time he refined it and it eventually became the first self-learning program.

Machine learning allows computers to "learn" or improve their decisions or predictions without being explicitly programmed.

Machine Learning is a new programming paradigm

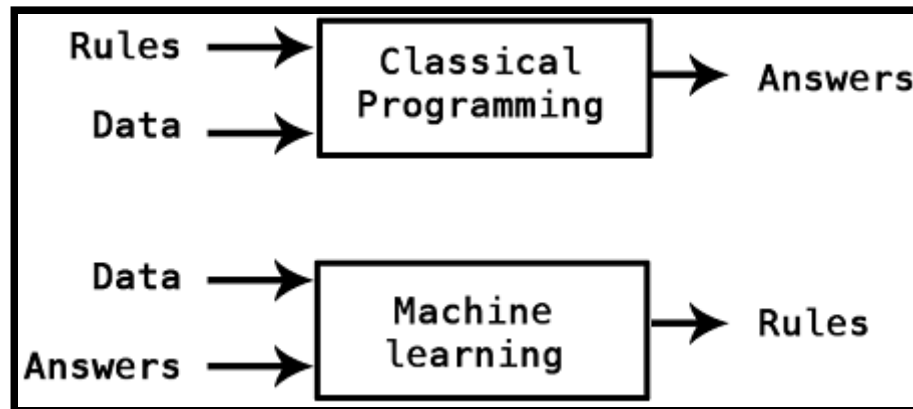


Image credit: François Chollet, Deep Learning with Python

Types of machine learning

- Supervised learning
- Unsupervised learning
- Reinforcement learning

Types of machine learning problems

- Prediction and forecasting
- Classification
- Clustering

Machine learning techniques

- Linear regression
- Logistic regression
- Neural networks



Amy Hoy

@amyhoy

 **Follow**



by today's definition, $y=mx+b$ is an artificial intelligence bot that can tell you where a line is going

9:44 AM - 29 Mar 2017

3,447 Retweets 5,608 Likes



82



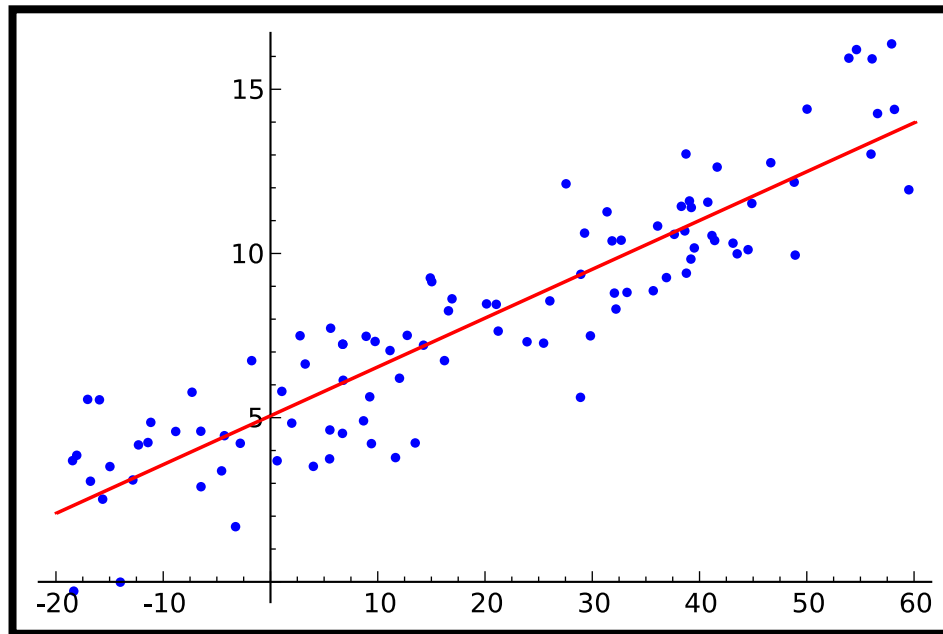
3.4K



5.6K



Linear Regression



$$y = \beta_0 + \beta_1 x$$

Linear Regression

$$y = \beta_0 + \beta_1 x$$

The problem is to find the "best" values for β_0 and β_1 given the data.

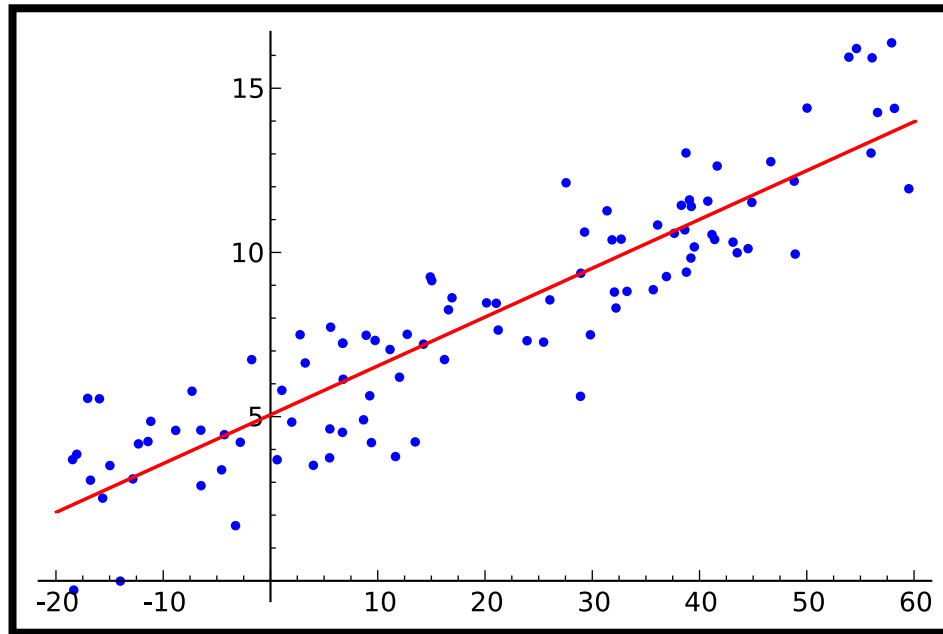
- Select loss function.
- Select minimization algorithm.
- Find values of β_0 and β_1 that minimize loss function.

Linear Regression

For linear regression, our loss function is the sum of squared errors

$$L(\beta_0, \beta_1) = \sum_{i=1}^N e_i^2$$

Linear Regression

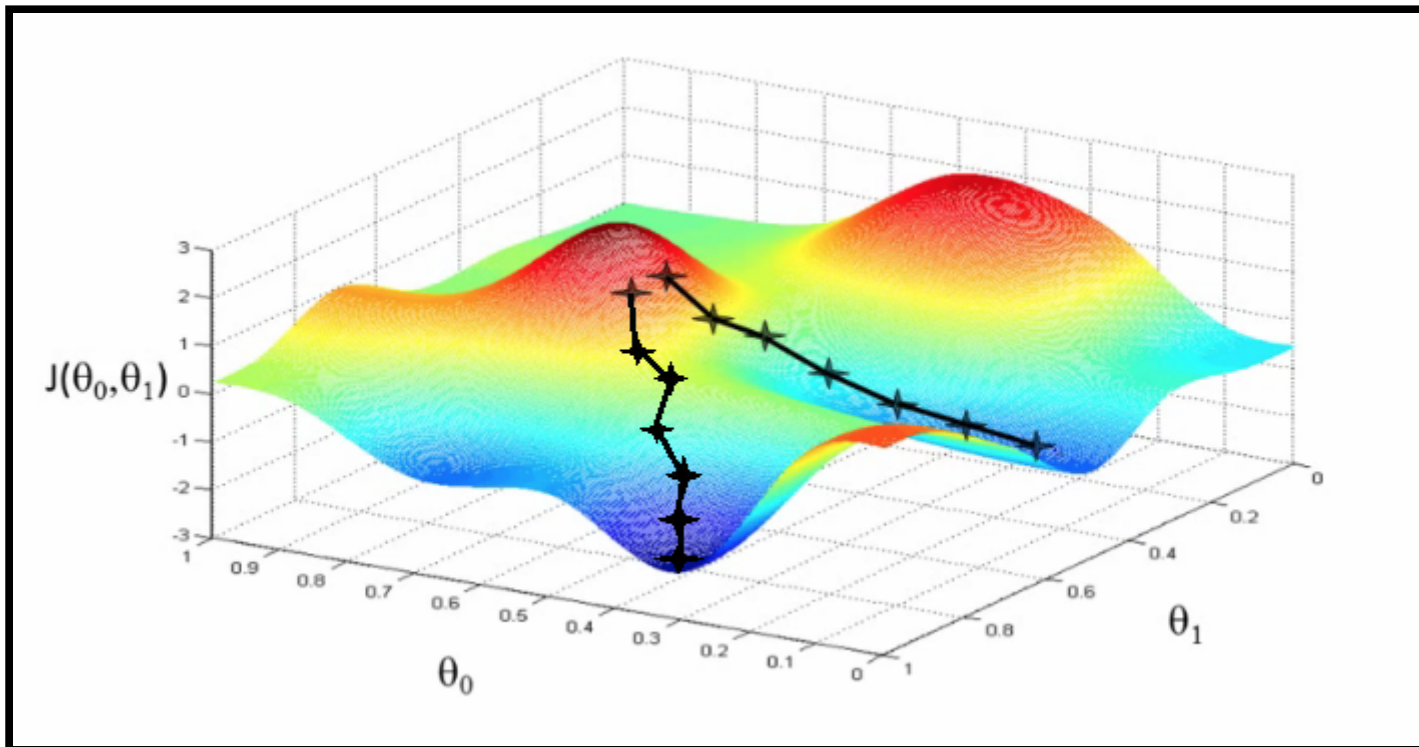


$$\hat{y}_i = \beta_0 + \beta_1 x_i$$
$$e_i = \hat{y}_i - y_i$$

Linear Regression

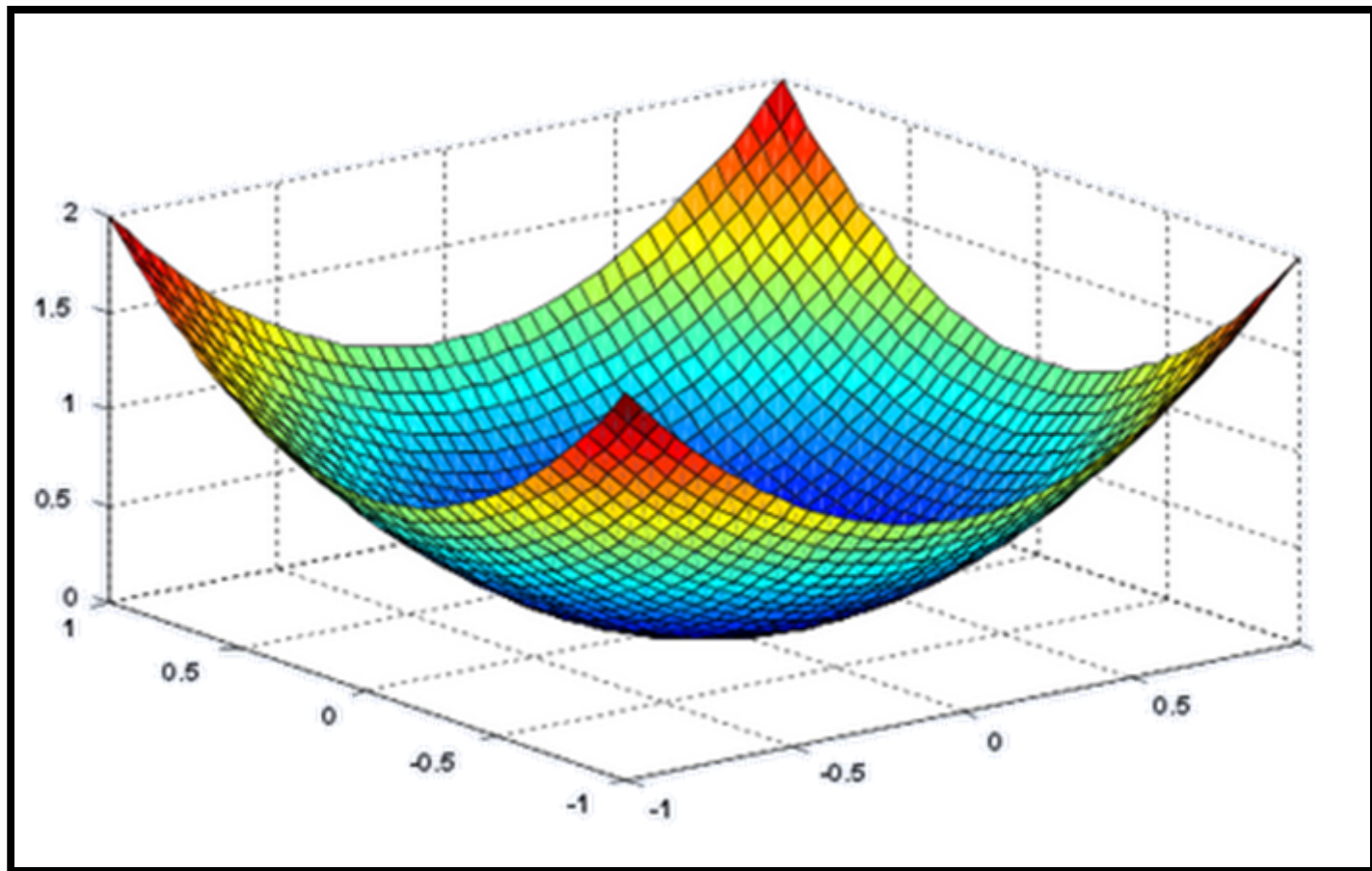
How to minimize the loss function?

Gradient Descent



Linear Regression

Actually, for linear regression with squared error loss function, the surface is a quadratic bowl.



Linear Regression

Horizontal cross-sections are ellipses. (Cross-sections in the image are not quite elliptical.)

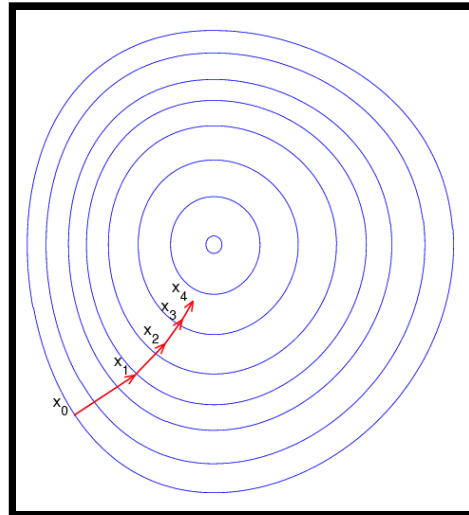


Image is public domain.

Start at any point and move in a direction perpendicular to the contour lines. Distance to move is based on steepness and the **learning rate**.

Linear Regression

The values of β_0 and β_1 that minimizes the loss function are our estimates $\hat{\beta}_0$ and $\hat{\beta}_1$.

When we get a new value of x , we can predict y using

$$\hat{y}_{new} = \hat{\beta}_0 + \hat{\beta}_1 x_{new}$$

Linear Regression

For example, the `mtcars` dataset is included with R includes data on 32 automobiles from a 1974 issue of *Motor Trend* magazine.

[illegible]

Linear Regression

We'll regress mpg on the other variables.

Call:

```
lm(formula = mpg ~ ., data = mtcars)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-3.4506	-1.6044	-0.1196	1.2193	4.6271

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	12.30337	18.71788	0.657	0.5181
cyl	-0.11144	1.04502	-0.107	0.9161
disp	0.01334	0.01786	0.747	0.4635
hp	-0.02148	0.02177	-0.987	0.3350
drat	0.78711	1.63537	0.481	0.6353
wt	-3.71530	1.89441	-1.961	0.0633 .
qsec	0.82104	0.73084	1.123	0.2739
vs	0.31776	2.10451	0.151	0.8814
am	2.52023	2.05665	1.225	0.2340
gear	0.65541	1.49326	0.439	0.6652
carb	-0.19942	0.82875	-0.241	0.8122

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.65 on 21 degrees of freedom

Multiple R-squared: 0.869, Adjusted R-squared: 0.8066

F-statistic: 13.93 on 10 and 21 DF, p-value: 3.793e-07

Binary Classification

Let's consider the problem of classifying elements of a set into two groups based on a classification rule.

Examples

- Spam detection - is an email message spam?
- Medical diagnosis - does the patient have diabetes?
- Marketing - will the customer make a purchase?
- Quality control - is a manufactured item defective?

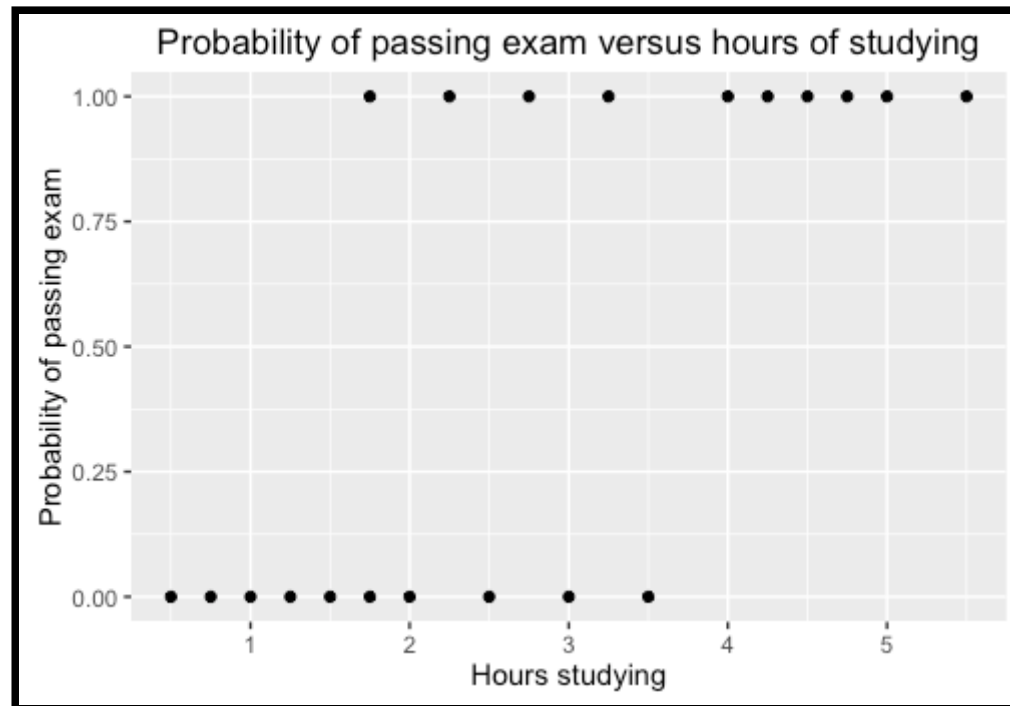
Binary Classification

Let's suppose we have a single explanatory variable x
and

$$y = \begin{cases} 1 & \text{if the example is a member of the class} \\ 0 & \text{otherwise} \end{cases}$$

Binary Classification Example

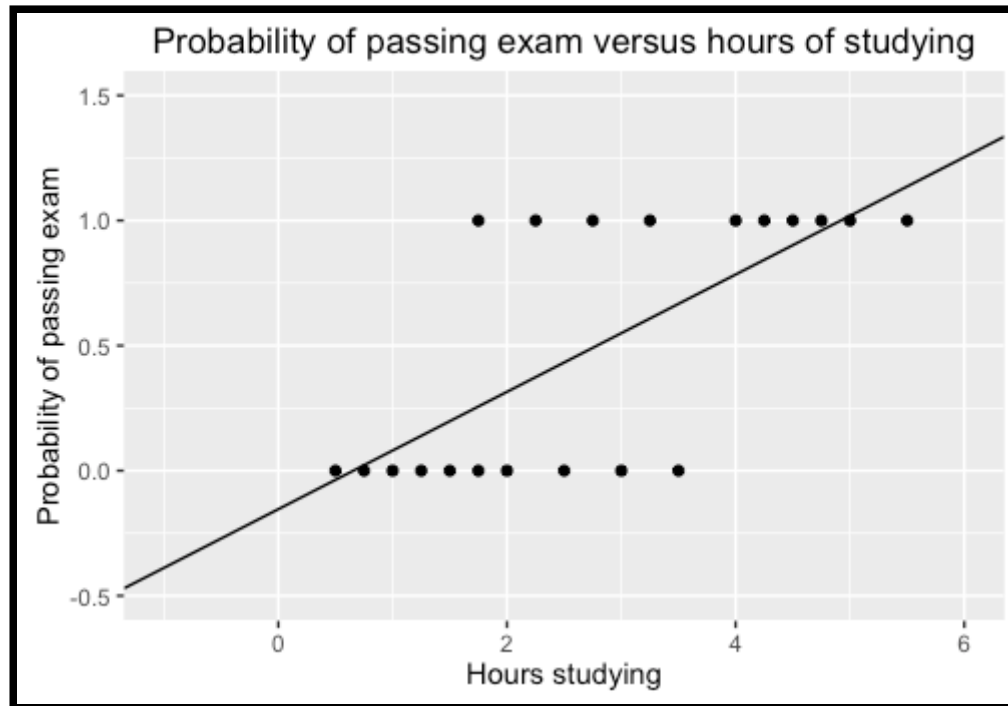
Suppose x is the number of hours a student studies for an exam, and y is whether the student passes the exam.



Binary Classification

We want $0 \leq \textit{Probability pass exam} \leq 1$.

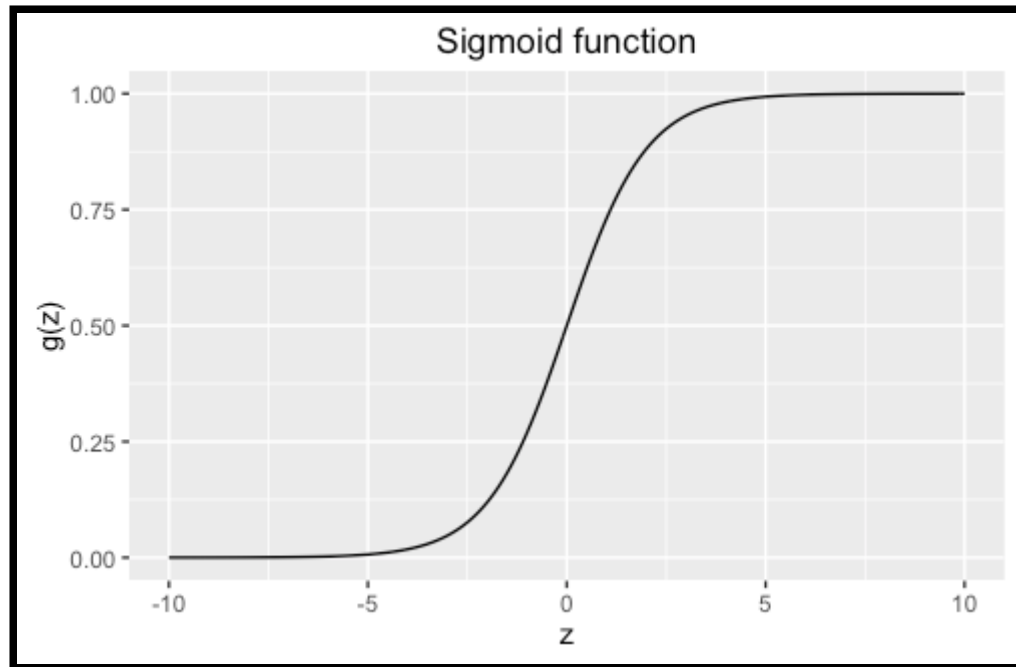
That means ordinary linear regression won't work.



Binary Classification

$$g(z) = \frac{1}{1 + e^{-z}}$$

is known as the **sigmoid** or **logistic** function.



Binary Classification

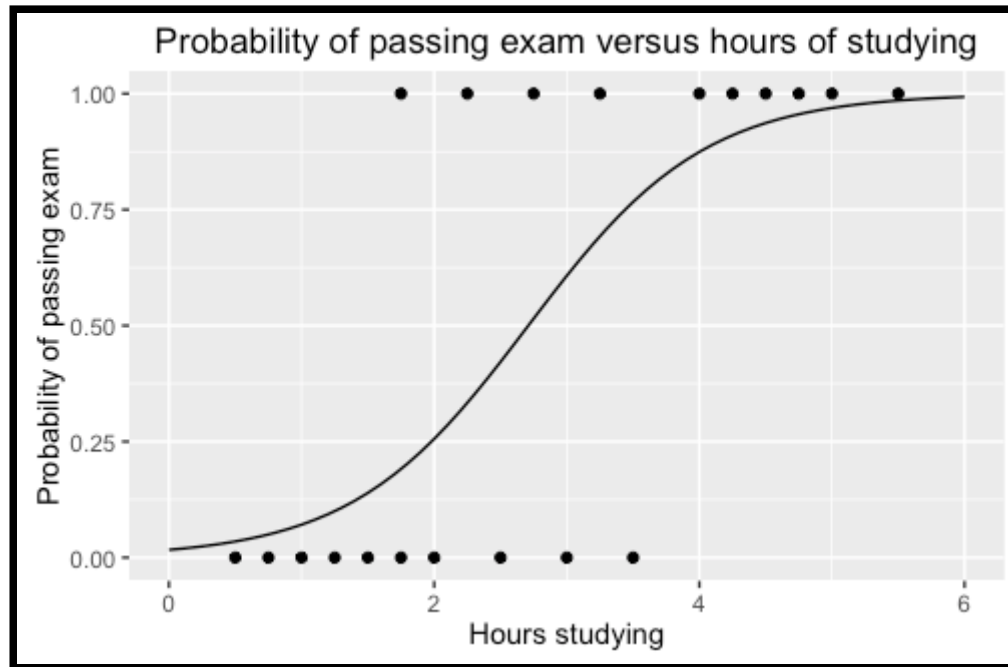
Suppose we use the following model.

$$z = \beta_0 + \beta_1 \textit{Hours}$$

$$\textit{Probability pass exam} = \frac{1}{1 + e^{-z}}$$

Binary Classification

If we estimate β_0 and β_1 using logistic regression, we get $\hat{\beta}_0 = -4.0777$ and $\hat{\beta}_1 = 1.5046$.



Binary Classification

For example, we might have a database of email messages labeled as "spam" or "email" with variables representing the relative frequencies of 50 common words and punctuation marks.

We could perform a logistic regression where

$$y = \begin{cases} 1 & \text{if the message is spam} \\ 0 & \text{otherwise} \end{cases}$$

Binary Classification

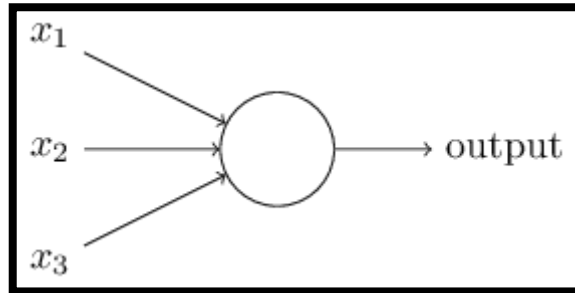
Confusion matrix

		predicted		
		F	T	
actual	F	TN 264	FP 14	
	T	FN 22	TP 158	

TN = true negatives
FN = false negatives
TP = true positives
FP = false positives

Neural Networks

Perceptron

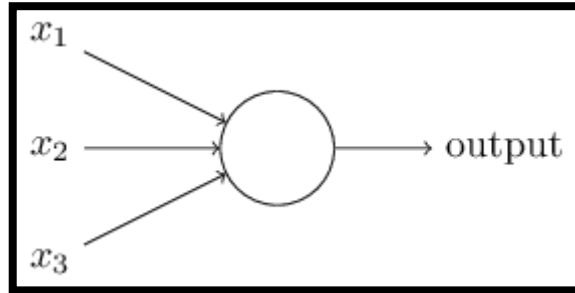


Output is binary.

Weights for each input

$$output = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{threshold} \end{cases}$$

Perceptron



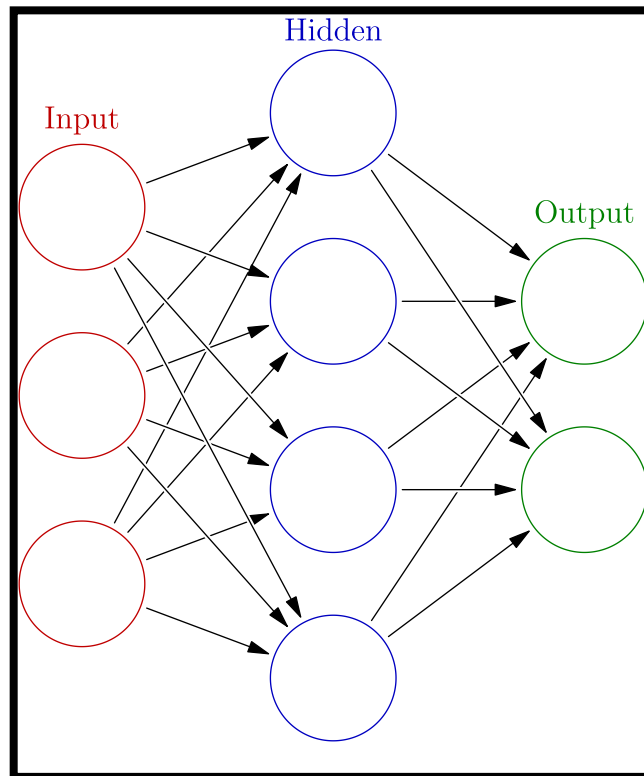
Equivalently, we can use **biases** instead of thresholds

$$output = \begin{cases} 0 & \text{if } \sum_j w_j x_j + b \leq 0 \\ 1 & \text{if } \sum_j w_j x_j + b > 0 \end{cases}$$

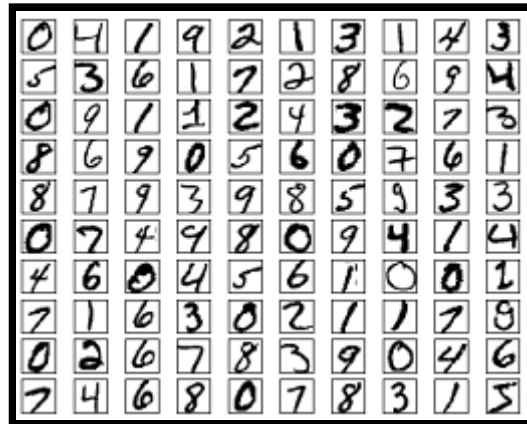
The weights (w_j) and biases (b) are the parameters that we will estimate or "learn."

Neural Networks

We can combine perceptrons together in layers to make an artificial neural network.



MNIST example



Data: <http://yann.lecun.com/exdb/mnist/>

MNIST example

Training data consists of 70,000 examples of handwritten digits.

For each example, we have

- 28 x 28 grey-level pixel image
- label indicating which digit it is (0-9)

MNIST example

In other words, our data consists of 70,000 rows with the following columns

- y = digit label (0-9)
- x_1 = value of pixel 1
- x_2 = value of pixel 2
- ...
- x_{784} = value of pixel 784

The pixel values are floating point numbers between 0.0 and 1.0.

MNIST example

Given values for x_1, x_2, \dots, x_{784} ,
we want to predict the digit or y .

One way we could do this is by running
a logistic regression for each digit.

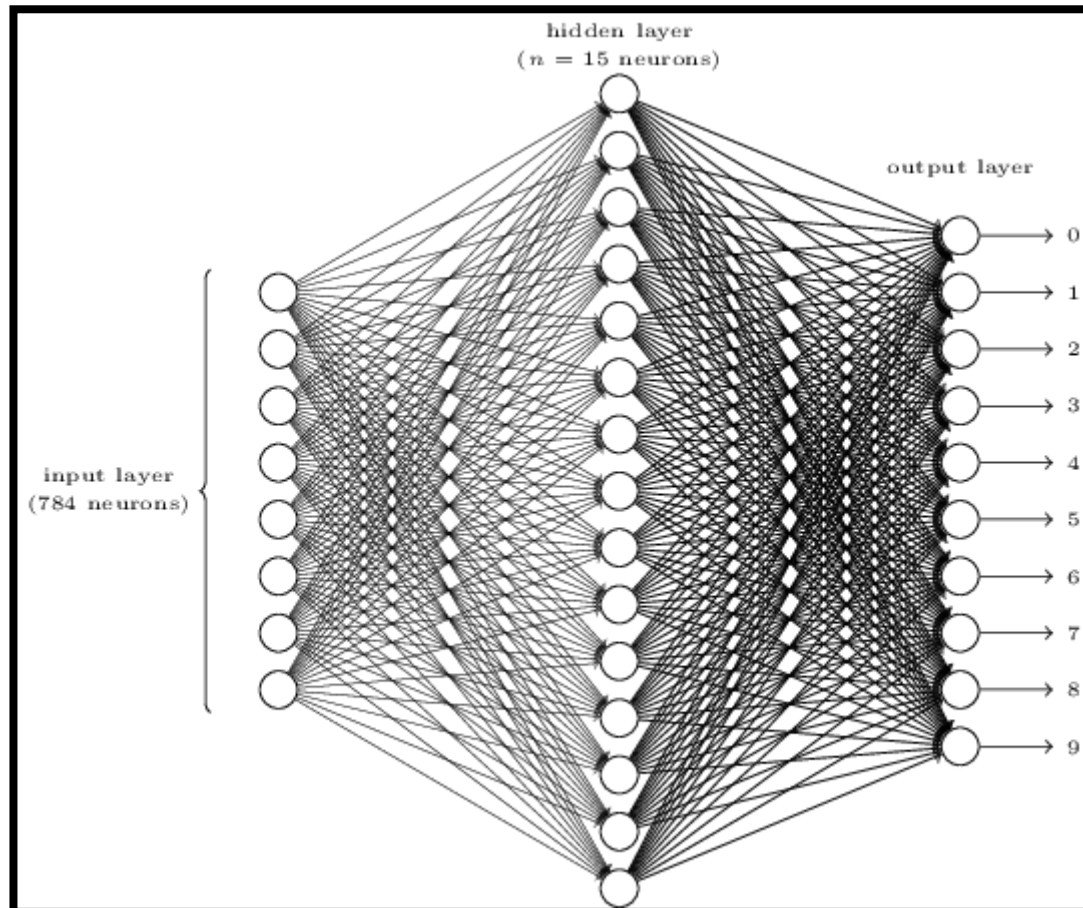
The first logistic regression would use

$$y = \begin{cases} 1 & \text{if the digit is zero} \\ 0 & \text{otherwise} \end{cases}$$

This is called **one versus rest** or **one against all**
classification.

MNIST example

We can get better results by using a neural network.



MNIST example

It turns out that perceptrons don't work very well in a multi-layer network.

Why?

Because a small change in the weights or bias can cause a neuron to flip from 0 to 1 or vice versa.

Solution?

Use neurons with sigmoid activation functions.

MNIST example

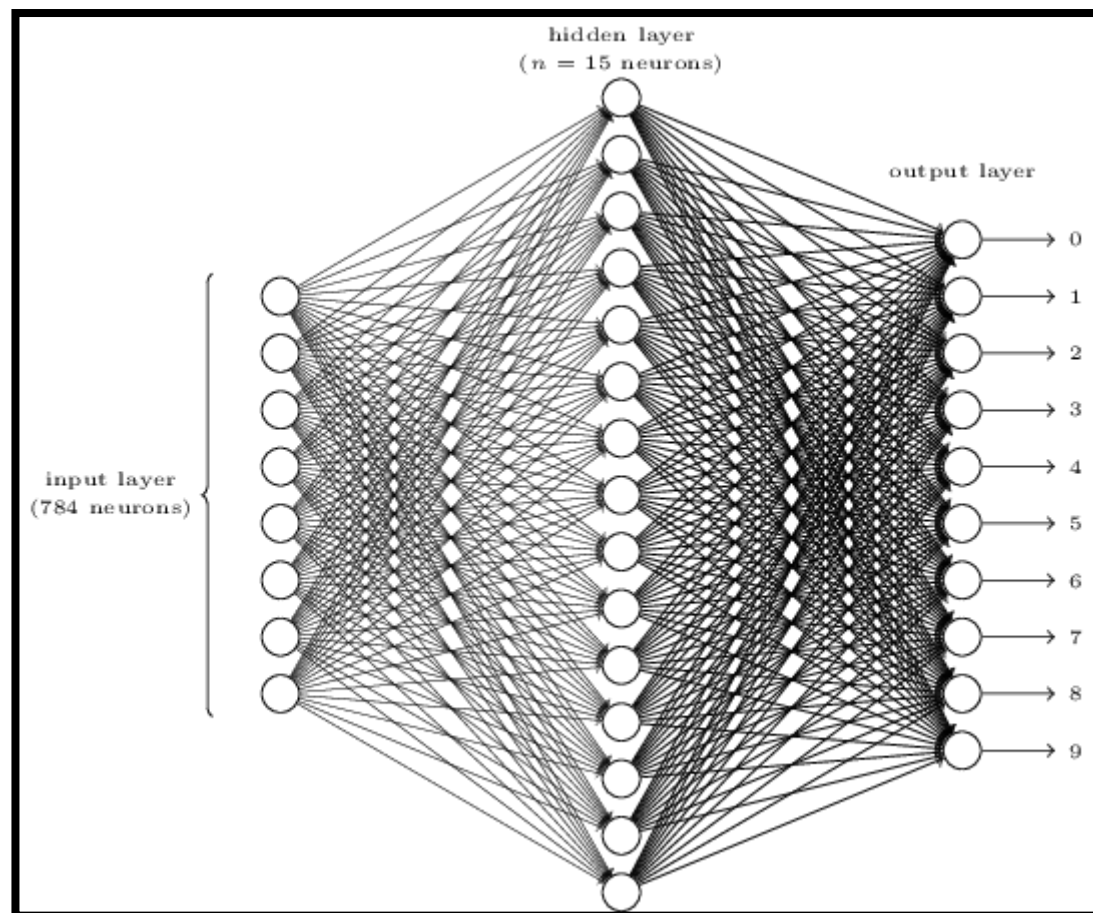
A neuron with a sigmoid activation function works like this:

$$z = b + \sum_{i=1}^M w_i x_i$$
$$output = \frac{1}{1 + e^{-z}}$$

where the x_i 's are the inputs, the w_i 's are the weights, and b is the bias.

MNIST example

All of our hidden neurons will use sigmoid activation functions.



MNIST example

TensorFlow code demo

What did we leave out?

A lot!

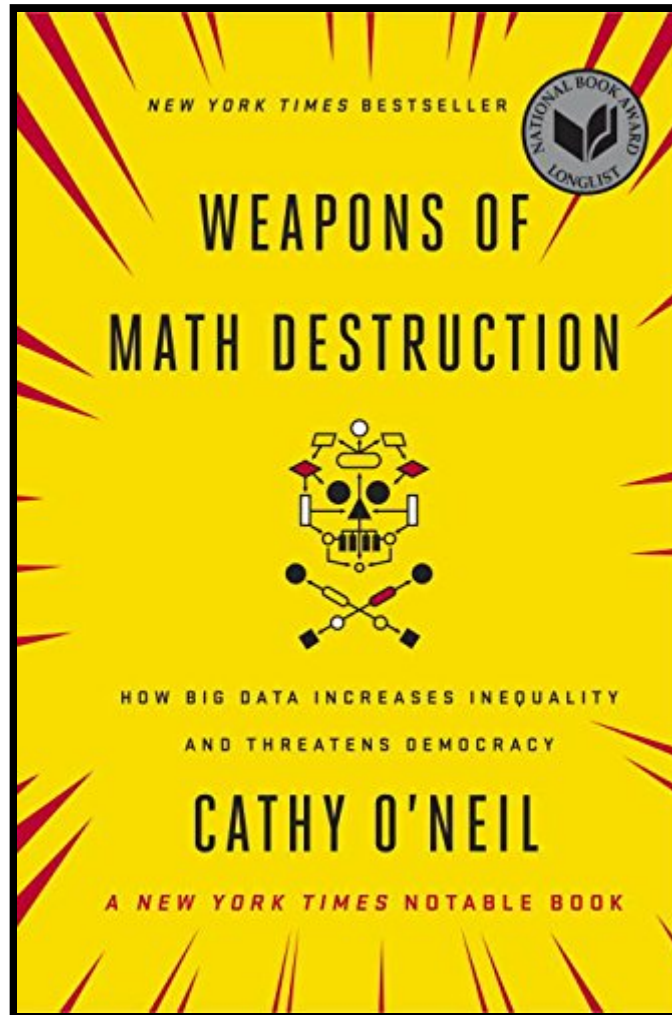
- Overfitting / regularization
- Convolutional neural networks
- Recurrent neural networks
- Much, much more

Keep in mind

- ML is susceptible to bias
- ML is hard to interpret
- ML is hard to explain
- ML is used in ways that affect people's lives

This makes ML potentially dangerous!

Recommended reading for everyone



WMD = algorithm that

- Is opaque
- Operates at scale
- Can damage people's lives

If you really want to learn ML

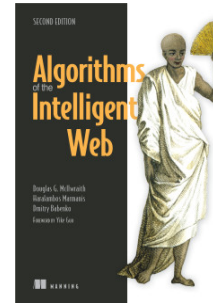
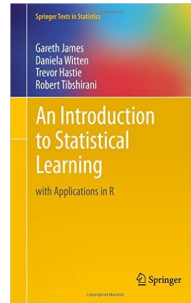
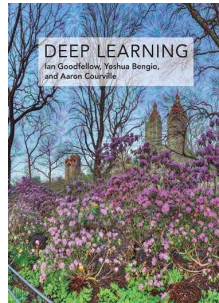
- Calculus
- Linear algebra
- Probability & statistics
- Information theory

You can learn these as you go.

No one knows everything.

Resources

Books



- neuralnetworksanddeeplearning.com
- Fast.ai
- Coursera
- Wikipedia

Questions?

THE END

Thank you



@david_body