Daily Inspiration

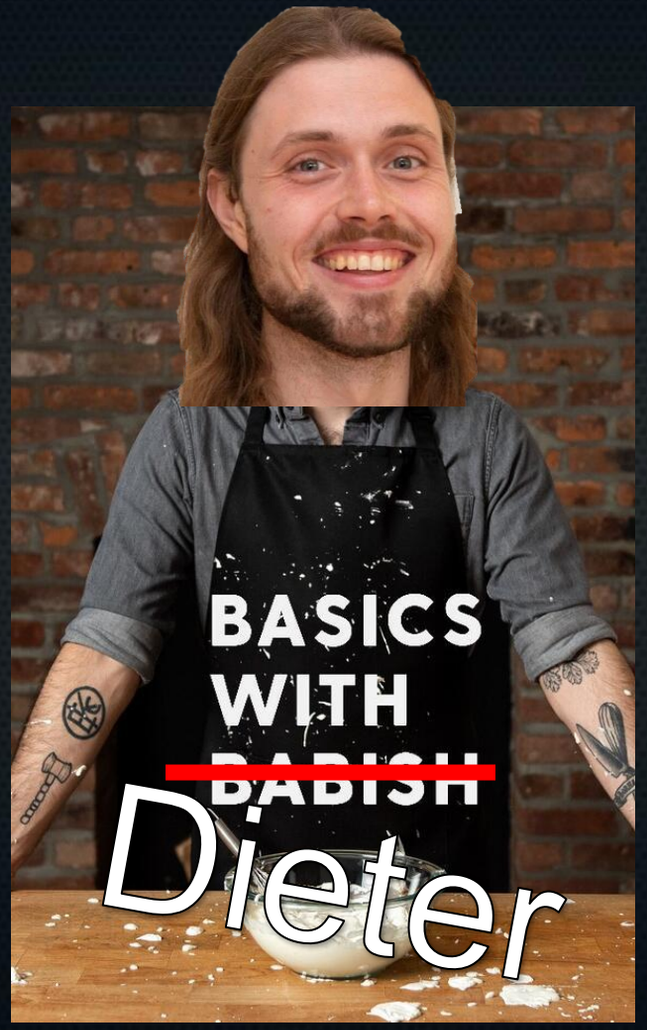
Today

- Intro
- Linear regression, cost function, gradient descent
- Bias and variance, cross-validation, learning curves
- Linear algebra primer

Intro

So uhh…what now?

Intro

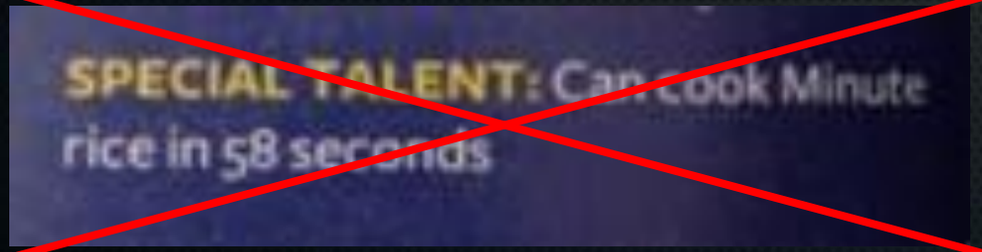


Who am I?

- ~~Your worst nightmare~~ a friendly *homo sapiens*
- Studied Biology, then Molecular and Cellular Life Sciences with Bioinformatics profile, doing mainly ML. Homegrown at UU.

Who am I?

- ~~Your worst nightmare~~ a friendly *homo sapiens*
- Studied Biology, then Molecular and Cellular Life Sciences with Bioinformatics profile, doing mainly ML. Homegrown at UU.
- Hidden talent:



Can sing Tom Lehrer's Elements Song (used to, anyway)

- I can also tie my own shoes →

Why am I teaching this?

UMC Utrecht
Center for Molecular Medicine | *De Ridder Group*

Research Publications Group members Vacancies Contact

Dieter Stoker

PhD-Student

Dieter completed both his Bachelor in Biology and his Master in Molecular and Cellular Life Sciences (Computational Track, with Bioinformatics profile) at Utrecht University, focussing on integrative omics/ML in both internships, one of them on DNA loop prediction in the de Ridder lab. After his Master, he assisted with (developing practicals for) courses in the new Bioinformatics and Biocomplexity Master, tried his hand at a grand writing project on the history of Computational Biology with Prof. Dr. Paulien Hogeweg (a bit of an overreach, in hindsight) and developed and taught the Basic Machine Learning for Bioinformatics course. He is now back for more ML, focussing chiefly on transfer learning or domain adaptation to be able to leverage the exponential progress in ML also for rare diseases or small patient cohorts. In his free time, he likes reading Terry Pratchett and various non-fiction works, pondering his place in the Universe (here on Earth, it turns out), walking and playing the occasional video game.

D.G.G.Stoker-6@umcutrecht.nl

Real MVPs

Andrew Ng,
Coursera ML guru

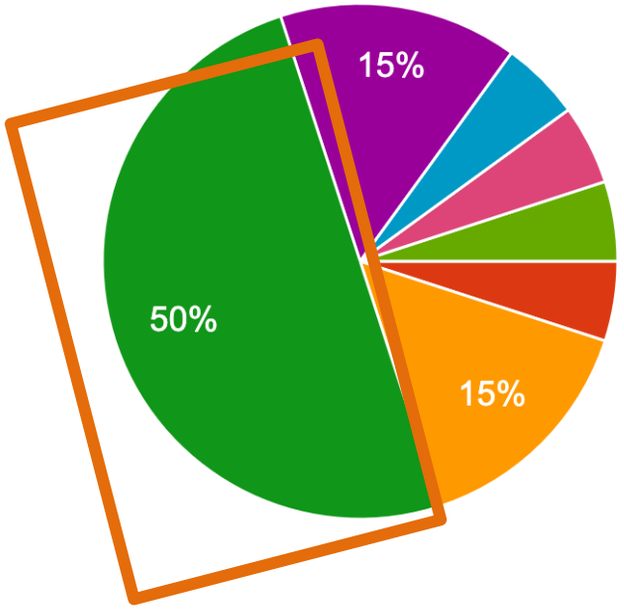
Jeroen de Ridder,
resident ML
maestro UMCU

Dieter Stoker,
PhD candidate
man

Who are you?

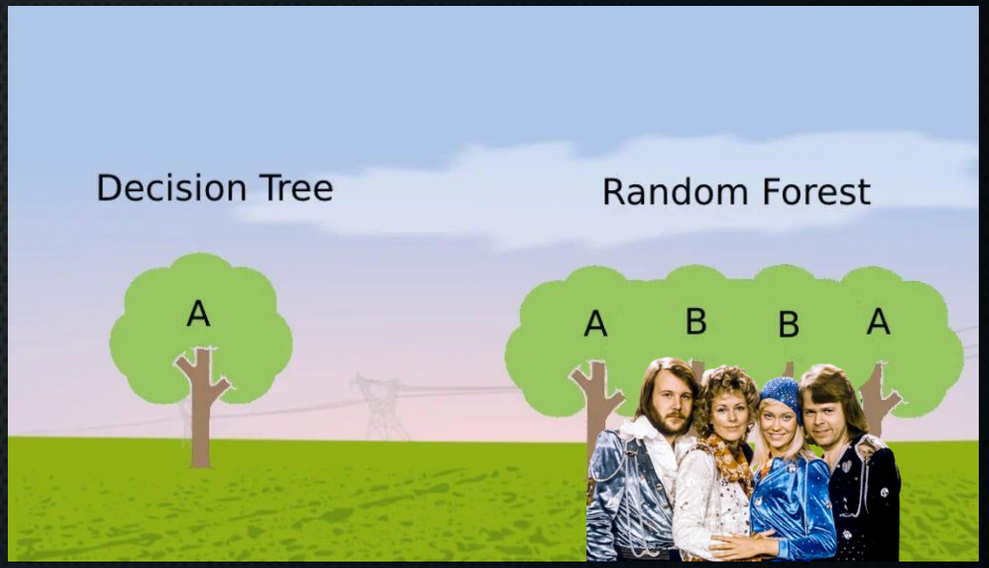
What's your level of ML experience?

20 responses



- I am literally Andrej Karpathy/Alec Radford/Andrew Ng/one of the DL god…
- I have an anime profile pic on X, tweet daily progress about my reimplementa…
- I have trained neural networks and/or…
- Kevin Kenna talked at me menacingly…
- Well my buddy ChatGPT/OpenAI Cod…
- I was blessed by the Omnissiah to lea…
- Well, I don't think my buddies know "e…
- I have worked with the Snowphlake m…

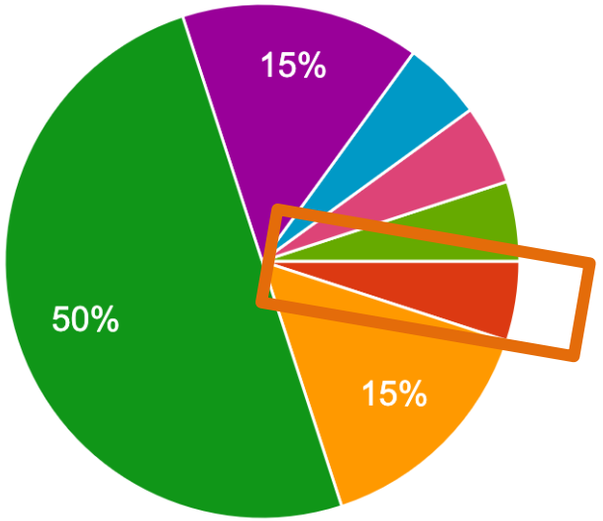
Who are you?



Who are you?

What's your level of ML experience?

20 responses



- I am literally Andrej Karpathy/Alec Radford/Andrew Ng/one of the DL god…
- I have an anime profile pic on X, tweet daily progress about my reimplementa…
- I have trained neural networks and/or…
- Kevin Kenna talked at me menacingly…
- Well my buddy ChatGPT/OpenAI Cod…
- I was blessed by the Omnissiah to lea…
- Well, I don't think my buddies know "e…
- I have worked with the Snowphlake m…

Who are you?

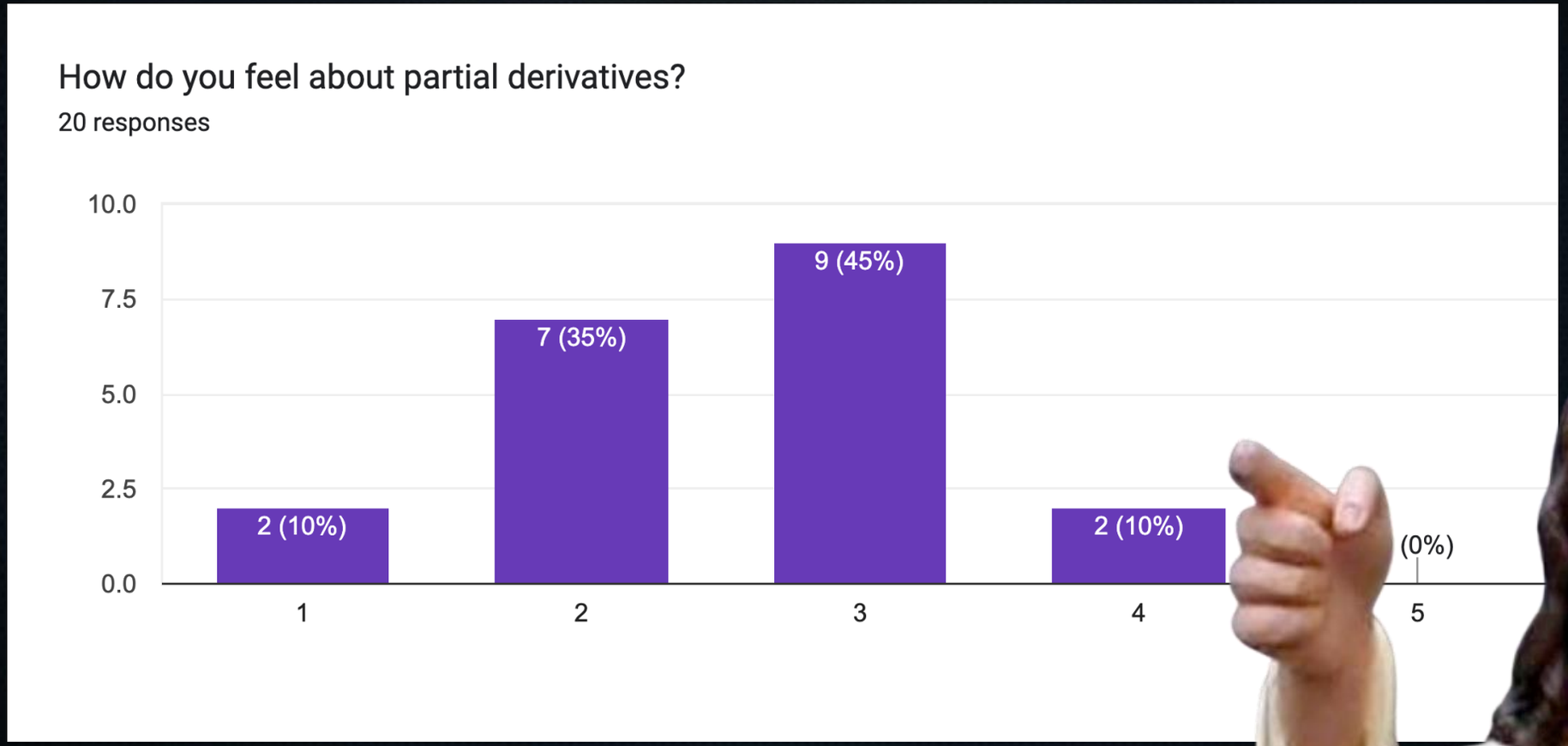
One of you satisfies the following:

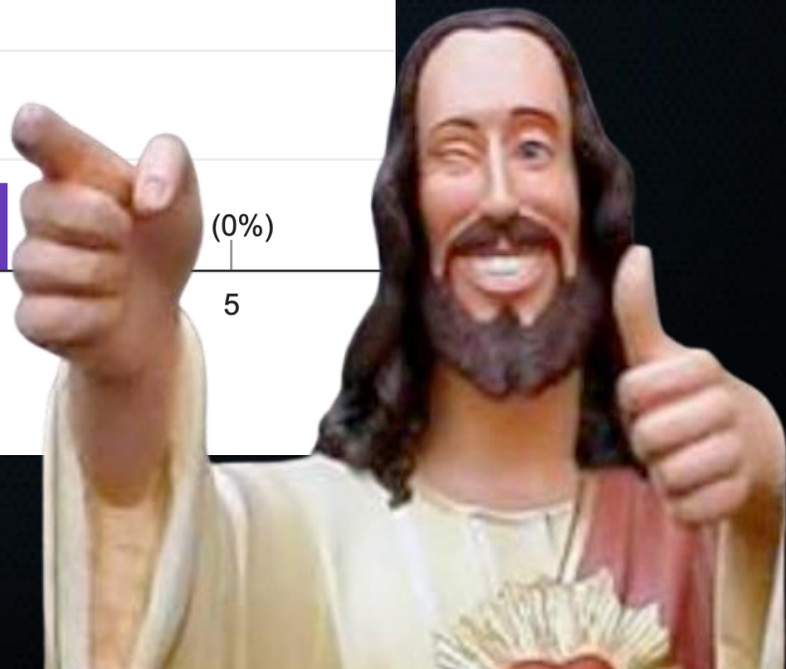
I have an anime profile pic on X, tweet daily progress about my reimplementation of Triton kernels and how I'm agencymaxxing and love effective acceleration

Please hmu after class to be my mentor.

Who are you?

Who are you?


How do you feel about partial derivatives?
20 responses
10.0
7.5
5.0
2.5
0.0
2 (10%)
7 (35%)
9 (45%)
2 (10%)
(0%)
1
2
3
4
5



Who are you?

Article | Published: 14 March 2018

DNA methylation-based classification of central nervous system tumours

David Capper, David T. W. Jones, Martin Sill, Volker Hovestadt, Daniel Schrimpf, Dominik Sturm, Christian Koelsche, Felix Sahm, Lukas Chavez, David E. Reuss, Annekathrin Kratz, Annika K. Wefers, Kristin Huang, Kristian W. Pajtler, Leonille Schweizer, Damian Stichel, Adriana Olar, Nils W. Engel, Kerstin Lindenberg, Patrick N. Harter, Anne K. Braczynski, Karl H. Plate, Hildegard Dohmen, Boyan K. Garvalov, ... Stefan M. Pfister + Show authors

Nature **555**, 469–474 (2018) | Cite this article

113k Accesses | **2542** Citations | **490** Altmetric | Metrics

'To train the RF classifier, the randomForest R package 40 was used. First, the most important features (probes) were selected by applying the Random Forest algorithm to the beta-values of all filtered 428,799 probes. For efficient computation, the probes were split into 43 sets of approximately 10,000 probes. For each set, 100 trees were fitted using 654 randomly sampled candidate features at each split (mtry parameter, square root of 428,799, as would be used by default when not splitting into sets). To take the imbalanced methylation class sizes into account a downsampling strategy was followed that ensures an identical number of samples per class (parameter sampsize=rep(8, 91)), eight reflecting the minimum number of cases in the 91 classes) 41. For all other parameters the default settings were used. This procedure was repeated 100 times, essentially fitting 10,000 trees per probe. Finally, features are selected by the permutation-based variable importance measure as implemented in the randomForest R package40.'

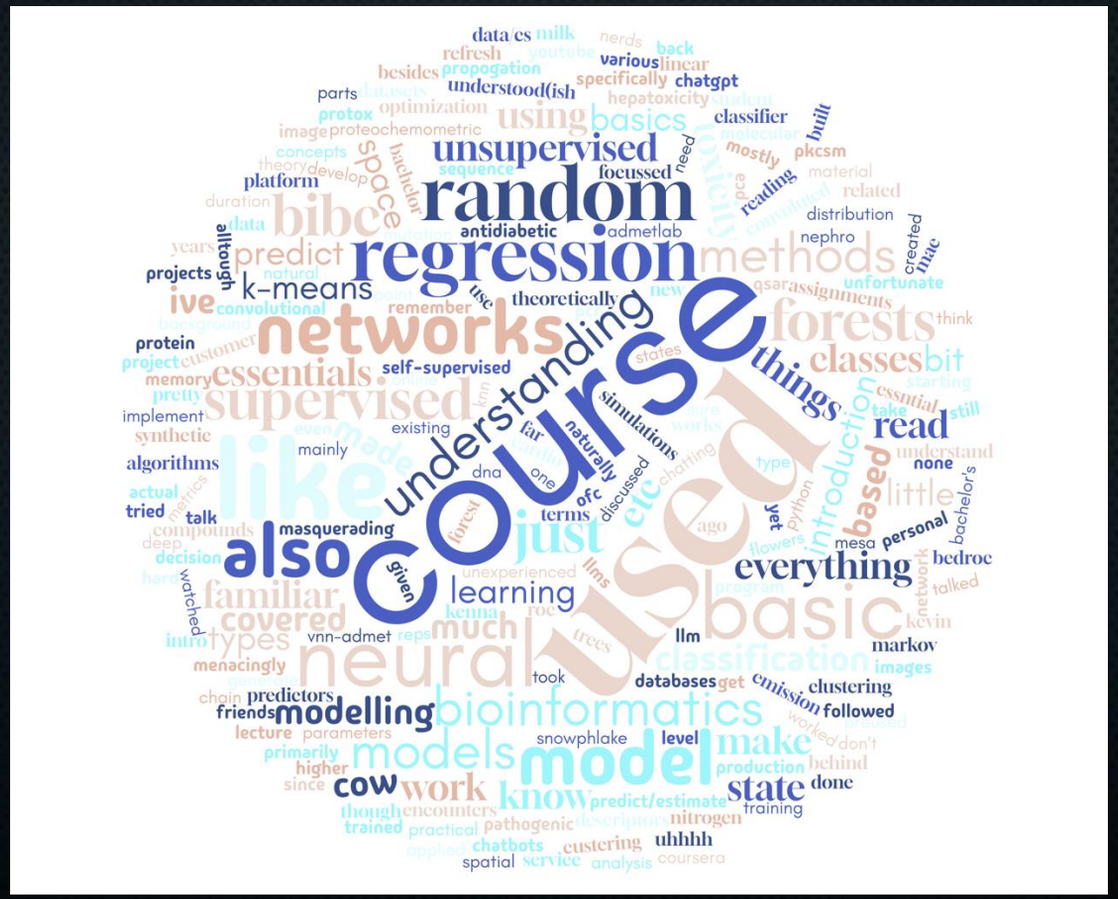
Who are yo

Who are you?

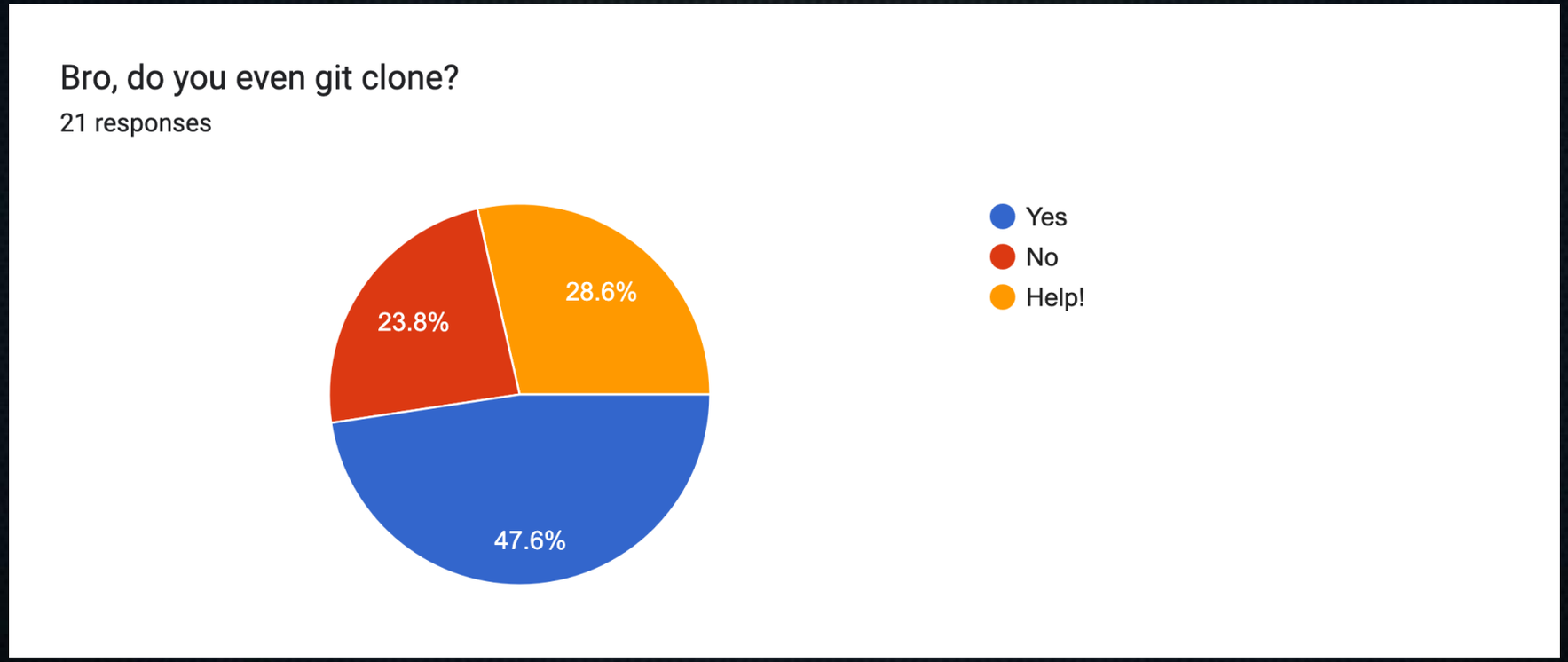
What do you hope to learn?

Who are you?

What do you already know?



Git or git gud?



Who are you?

Deep learning?

W

19

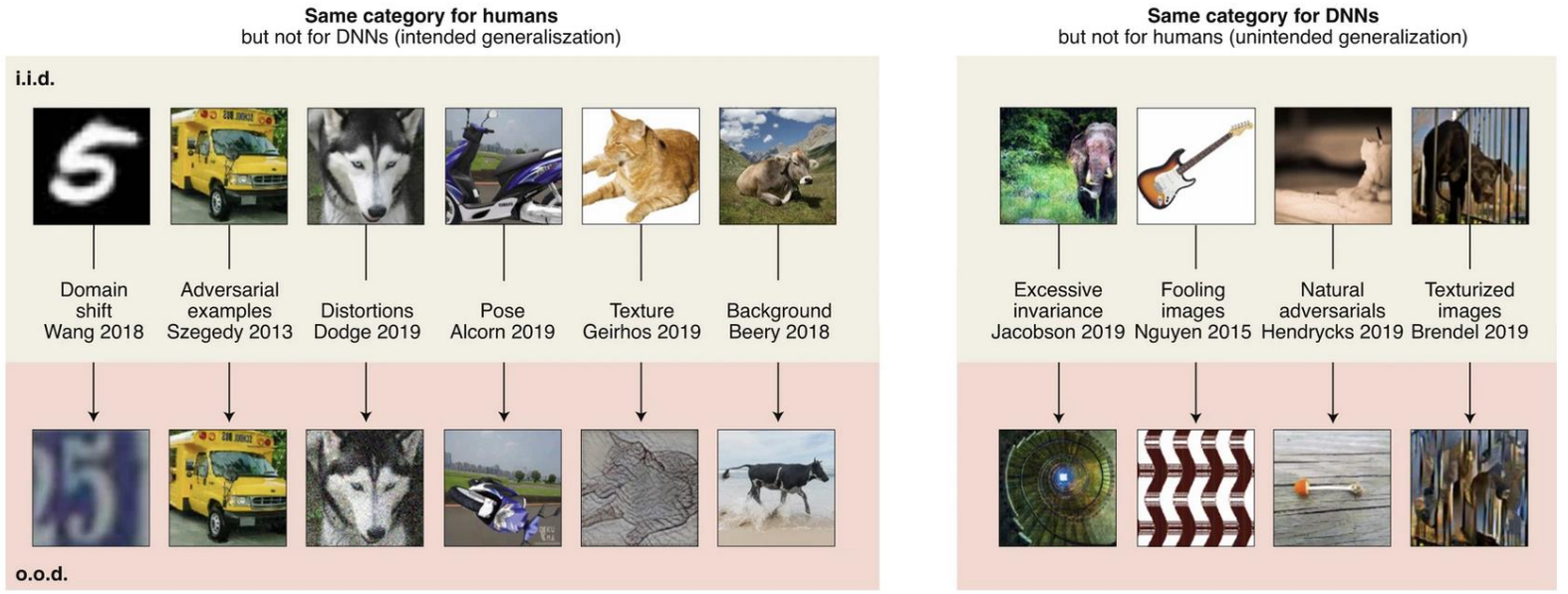
Figure 1: brr

Your questions

What about bias?

Fig. 4: Humans and DNNs both generalize, but they generalize very differently.

From: Shortcut learning in deep neural networks



Left: image pairs that belong to the same category for humans, but not for DNNs. Right: image pairs assigned to the same category by a variety of DNNs, but not by humans. Figure adapted with permission from ref. [47], Elsevier (5 images); ref. [8], ICLR (trucks); ref. [38], ICLR (bottom cat); ref. [9], Springer (bottom cow); ref. [45], IEEE (curved pattern).

https://www.nature.com/articles/s42256-020-00257-z

Your questions

What about bias?

Figure 1: Images of household items across the world, and classes recognized in these images by five publicly available image-recognition systems. Image-recognition systems tend to perform worse in non-Western countries and for households with lower incomes. See supplemental material for license information.

https://arxiv.org/abs/1906.02659

Your questions

What about bias?

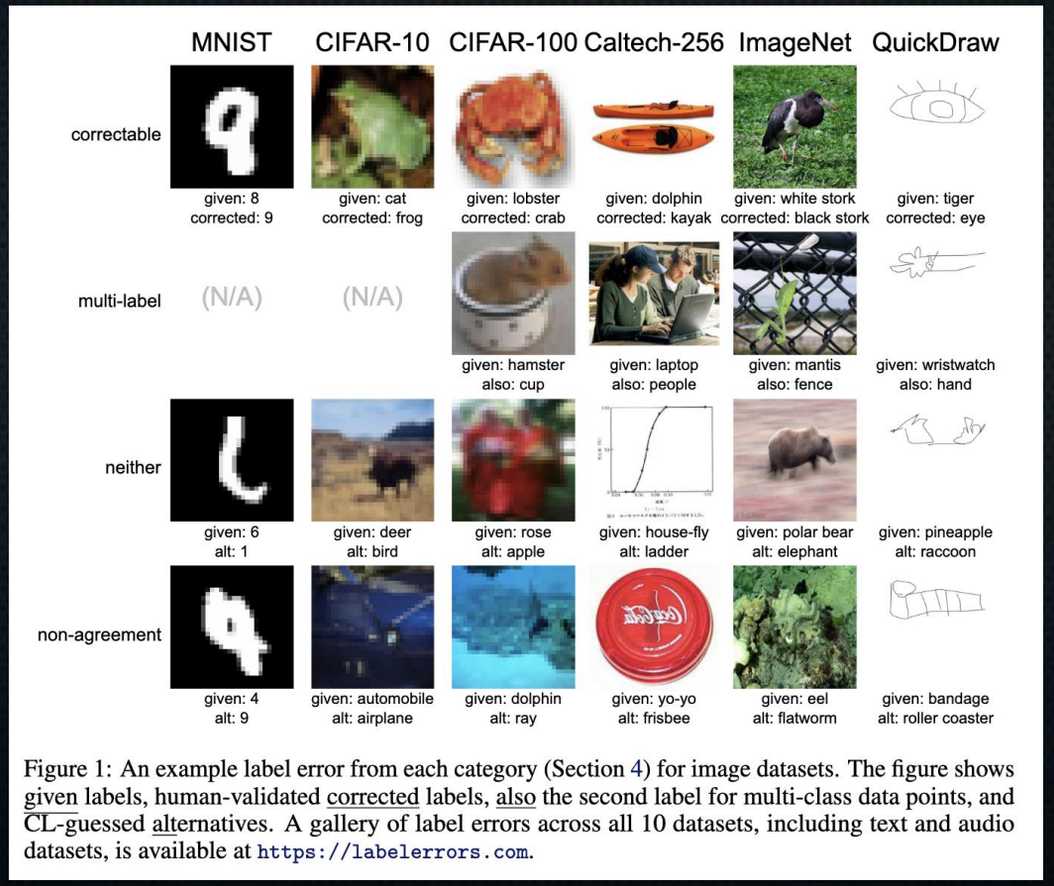


Figure 1: An example label error from each category (Section 4) for image datasets. The figure shows given labels, human-validated corrected labels, also the second label for multi-class data points, and CL-guessed alternatives. A gallery of label errors across all 10 datasets, including text and audio datasets, is available at https://labelerrors.com.

https://arxiv.org/abs/2103.14749

Your questions

Why is ChatGPT amazing
but also an idiot that struggles
with addition?


Language Models Use Trigonometry to Do Addition

Subhash Kantamneni [1] Max Tegmark [1]

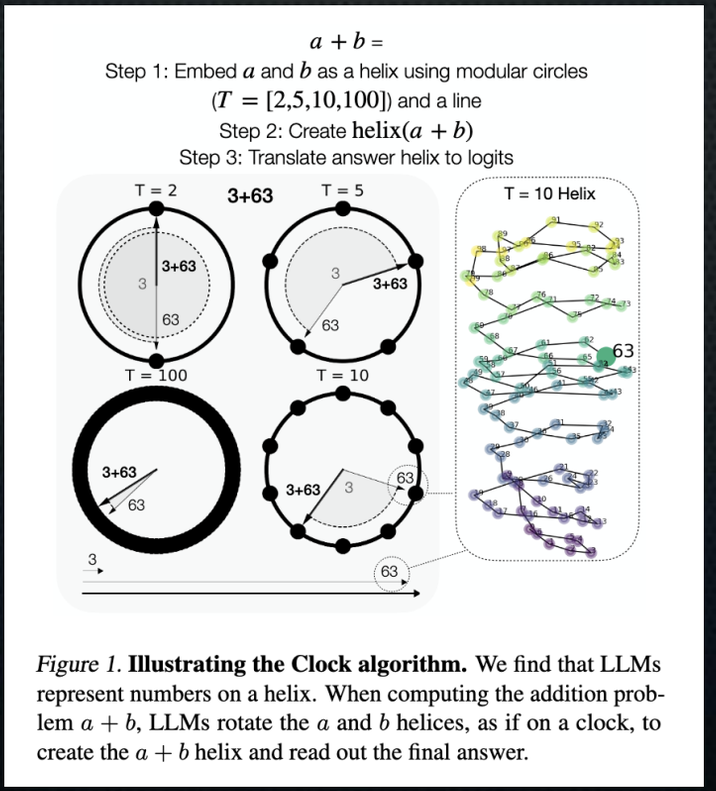


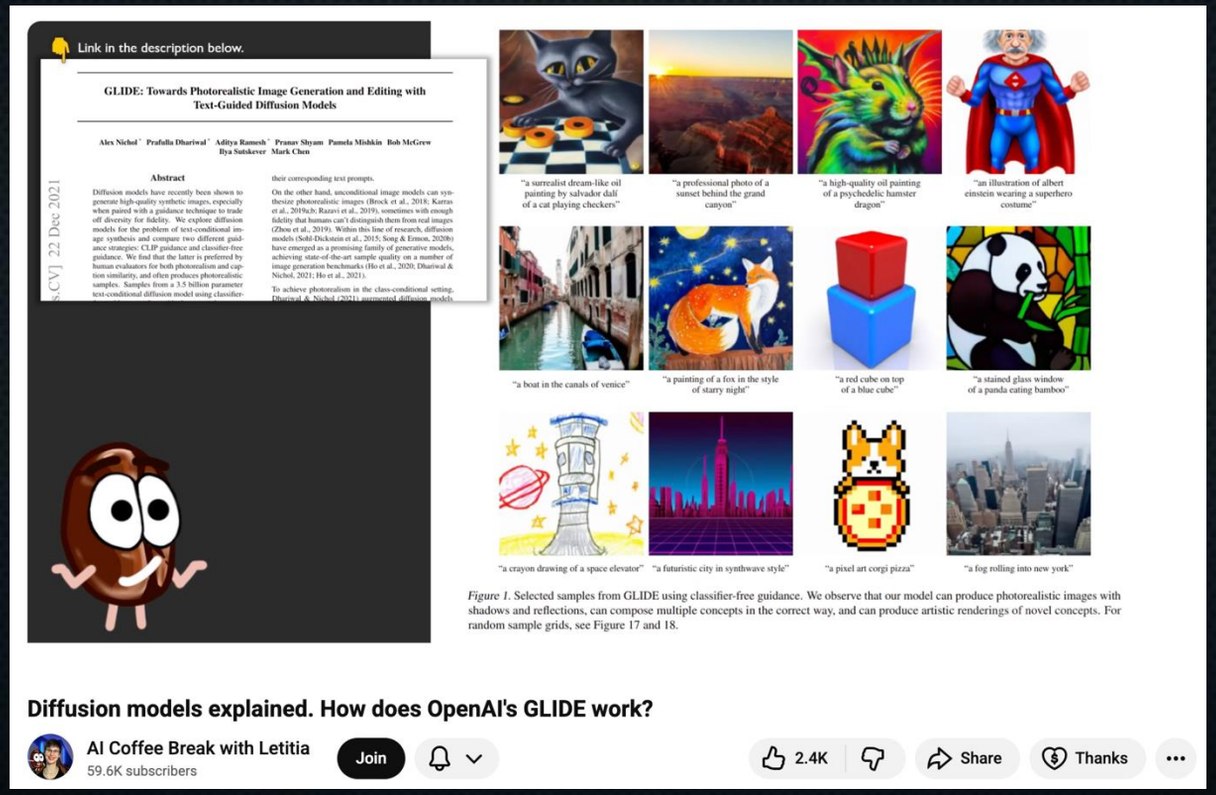
Figure 1. **Illustrating the Clock algorithm.** We find that LLMs represent numbers on a helix. When computing the addition problem $a + b$, LLMs rotate the a and b helices, as if on a clock, to create the $a + b$ helix and read out the final answer.

"How can an LLM trained on next-discrete token prediction, that does not see numbers as logical sequences at all, that is then subjected to SFT and RLHF, even learn to do addition kinda well sometimes? Just from text and examples, really, without further logic?"

https://arxiv.org/abs/2502.00873

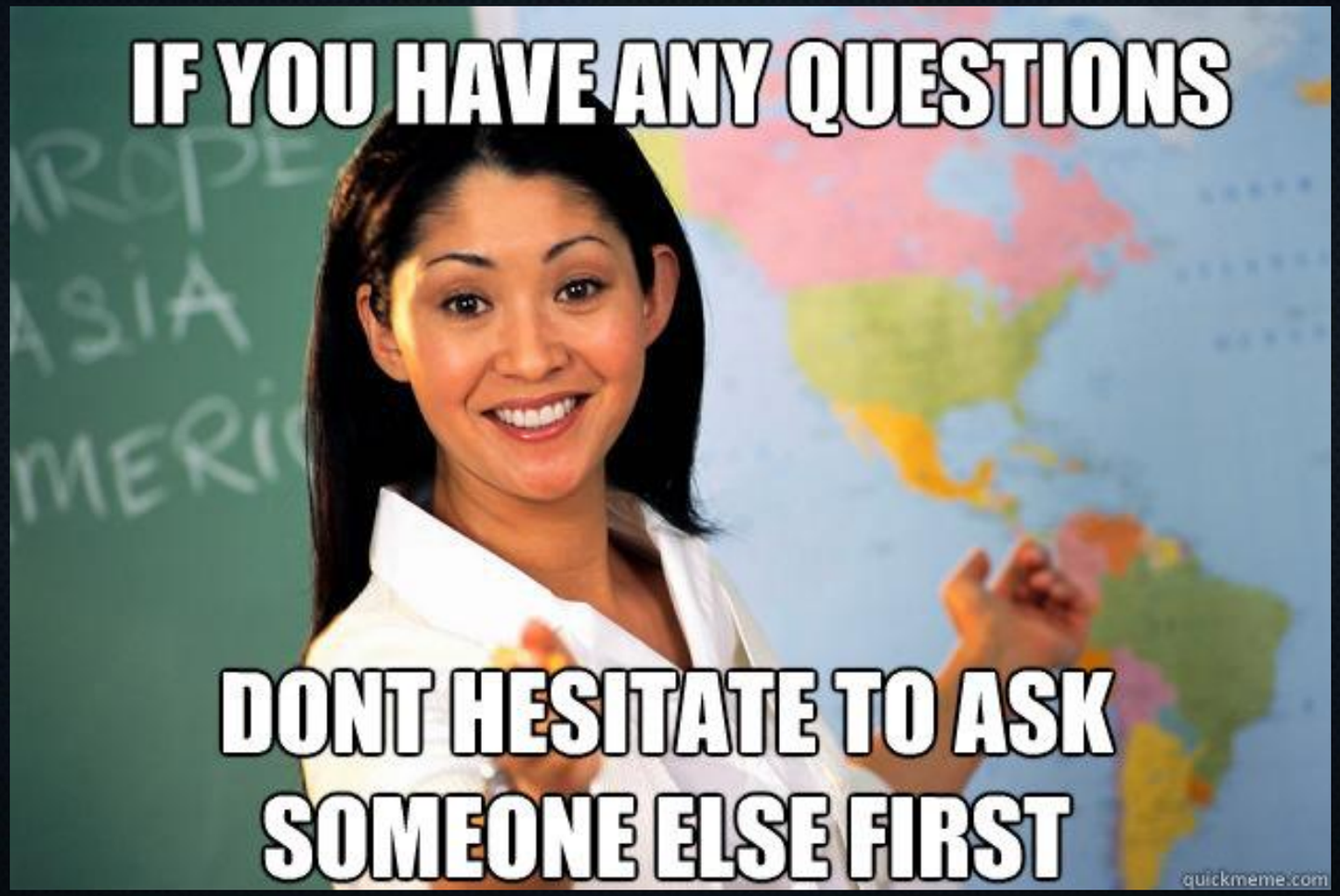
Your questions

Hello yes diffusion models plz?



https://www.youtube.com/watch?v=344w5h24-h8
https://www.youtube.com/watch?v=J87hffSMB60

More?


IF YOU HAVE ANY QUESTIONS
DONT HESITATE TO ASK
SOMEONE ELSE FIRST
quickmeme.com

Course content: what I hope to teach you

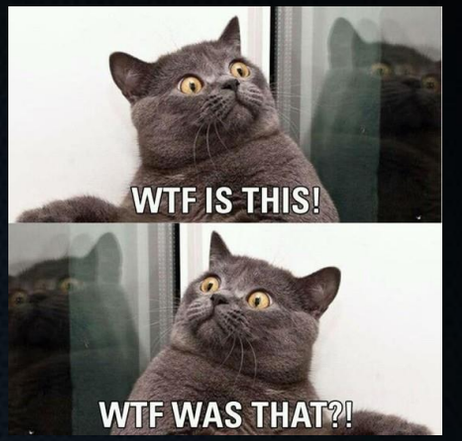
- What is ML? Cost functions, gradient descent, generalisation, bias and variance.
- Week 1: low-level understanding: able to implement linear regression, logistic regression, neural networks, ~~clustering~~ and PCA yourself using numpy in Python.
- Week 2: modern ML library (scikit-learn/Keras) workflow (one day), + introduction of a hands-on project (~2 days; finish after course). Written exam about lecture and computer lab concepts **on the 6th**.

Setup per day

- Morning/early afternoon:
  - Lectures of ~45-60 minutes, interspersed with (2) short practical(s).
- Rest of the day:
  - Somewhat longer afternoon practical
- Taken together:
  - Lecture (09:00-09:45)
  - Short practical 1 (09:45-10:45)
  - Lecture (10:45-11:30)
  - Short practical 2 (11:30-12:30)
  - Lecture (13:15-14:00)
  - Afternoon practical (14:00-~16:45)
  - Final takeaway summary (~16:45-17:00)

Lunch somewhere here
(12:30-13:15 is the idea)
Might shift times a bit.

Warning: difficulty ahead



- Last years: day 3 (neural networks) is very difficult, so we skip implementing clustering yourself on day 4 because there simply isn't enough time.

Questions

- Besides this, feel free to raise your hand and ask questions when something is unclear.

- If there’s no hands raised right now then we’ll dive right in!

Aha, I can ask you questions

What is ML?

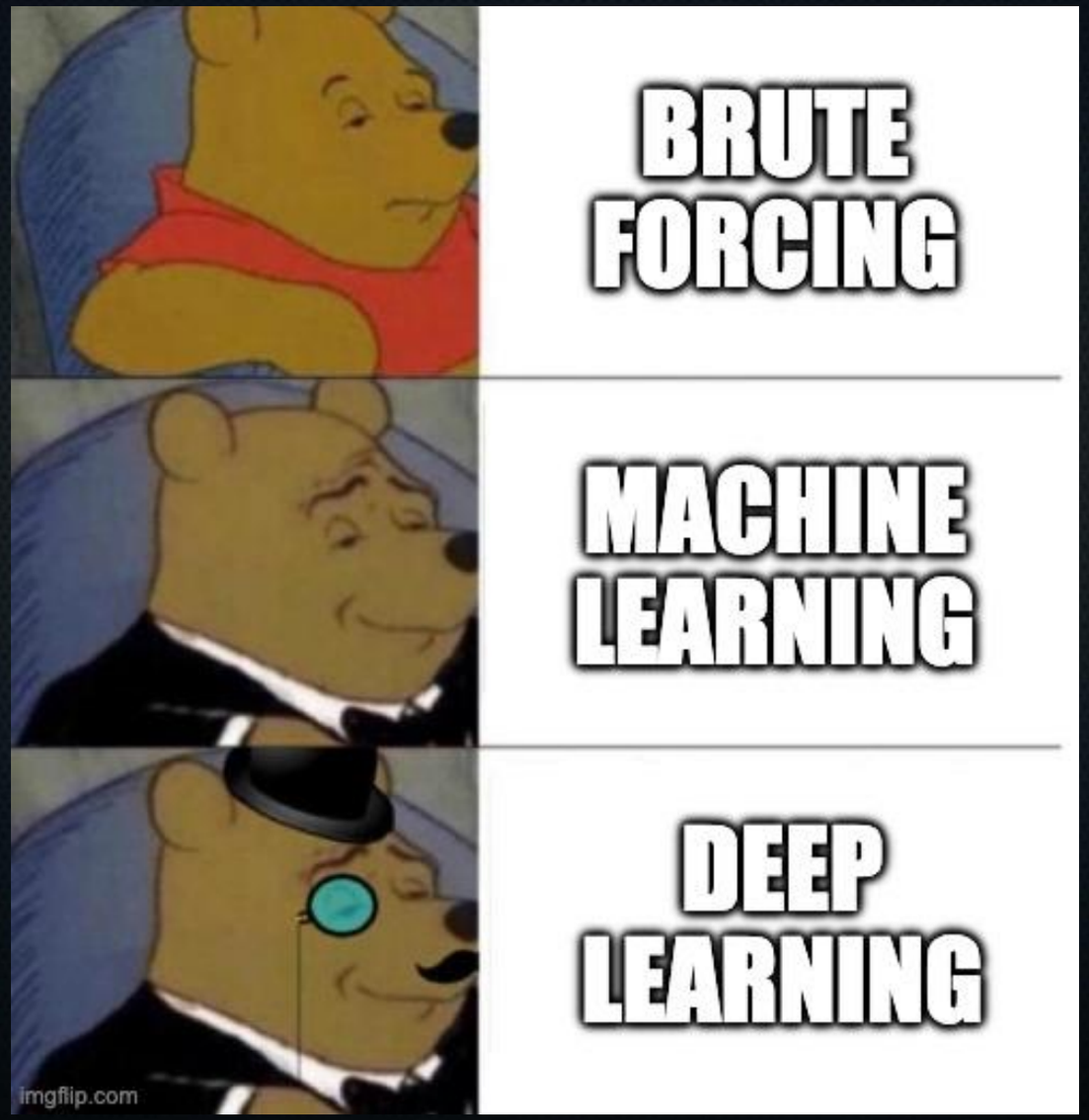
What is deep learning?

What is the goal of ML?

What is a ChatGPT and how does it work?

What is dimension reduction?

TL;DR Machine Learning



TL;DR Machine Learning

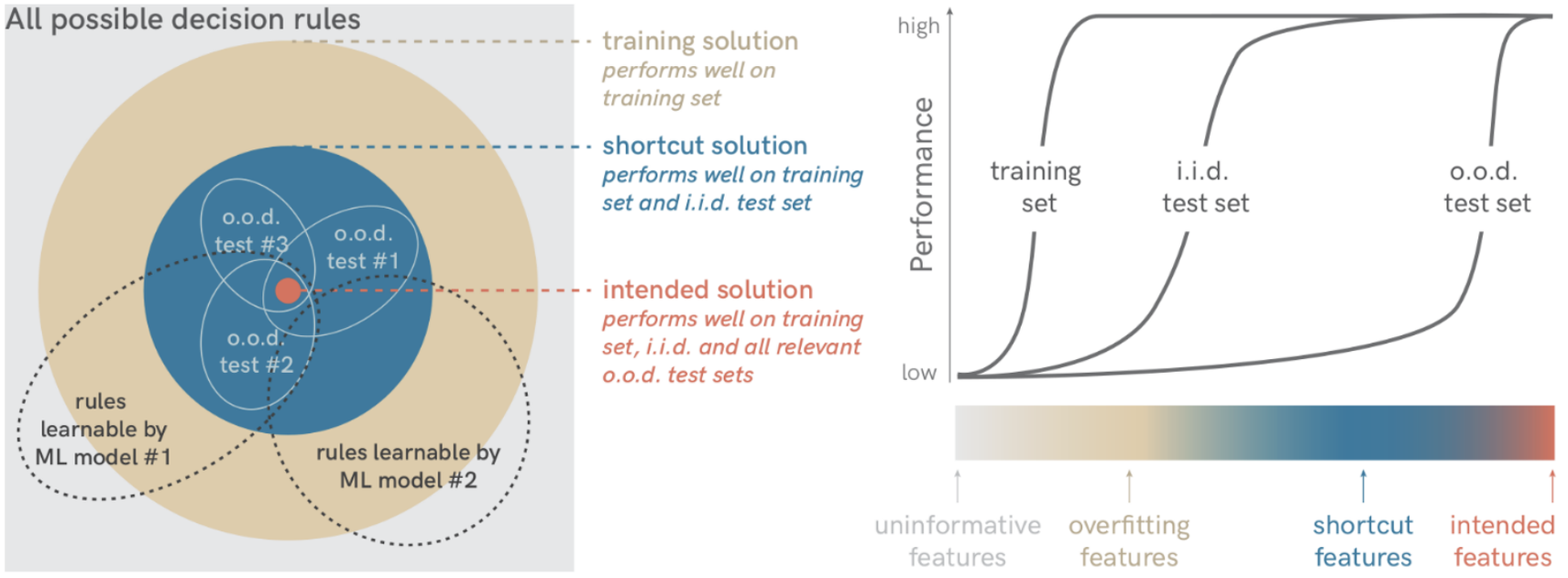


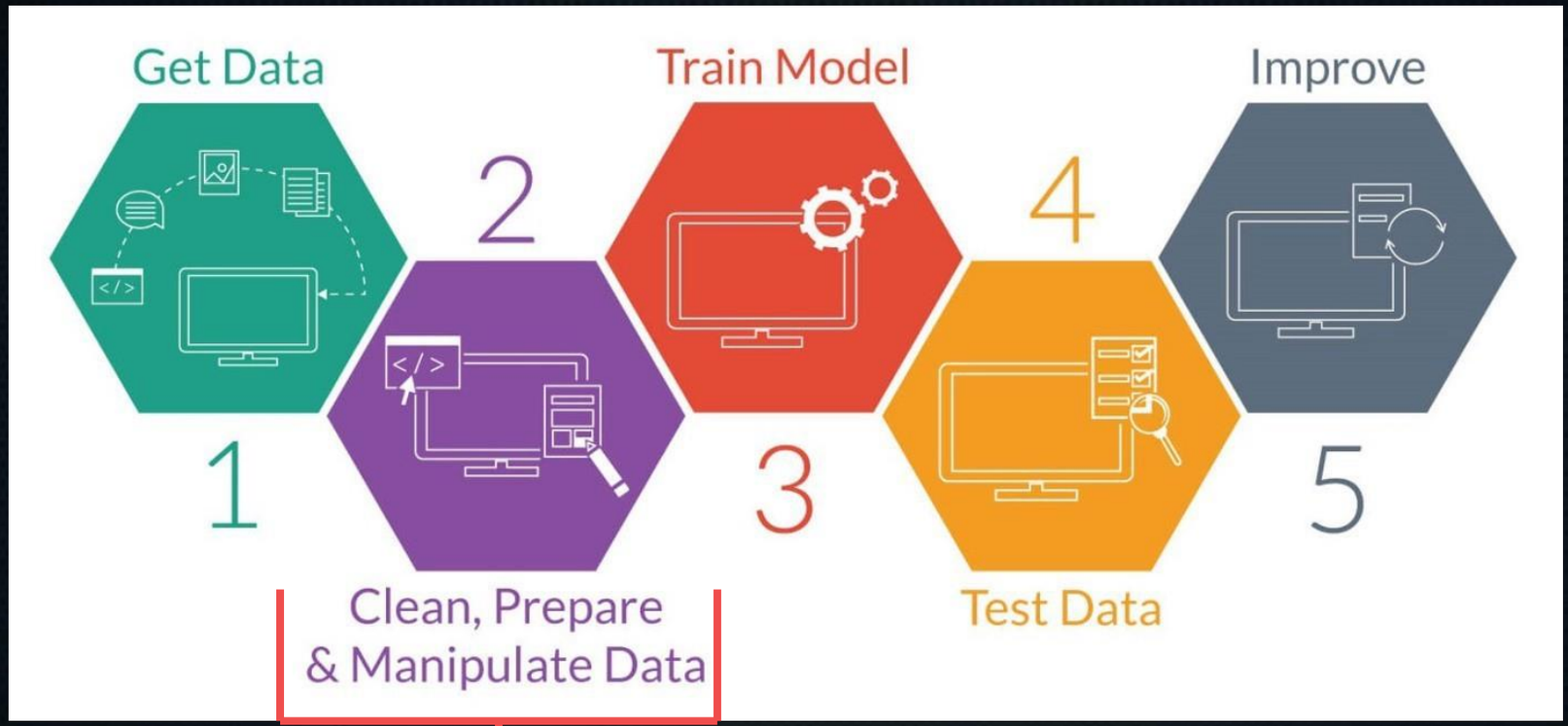
Figure 3. Taxonomy of decision rules. Among the set of all possible rules, only some solve the training data. Among the solutions that solve the training data, only some generalise to an i.i.d. test set. Among those solutions, shortcuts fail to generalise to different data (o.o.d. test sets), but the intended solution does generalise.

https://www.nature.com/articles/s42256-020-00257-z

This presentation

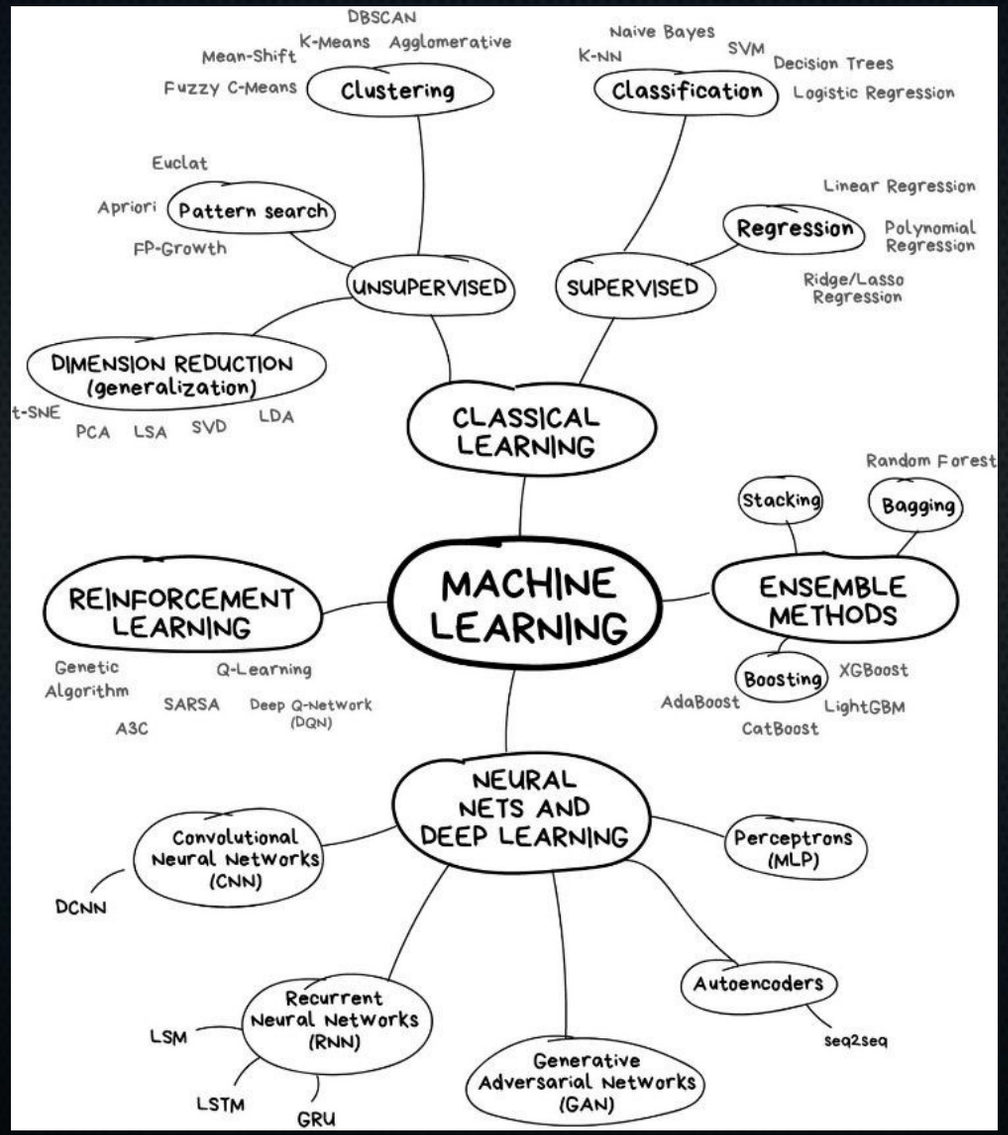
- Two branches of ML: supervised and unsupervised
- Terminology
- Linear regression & cost function
- Gradient descent & partial derivatives

The ugly truth about ML

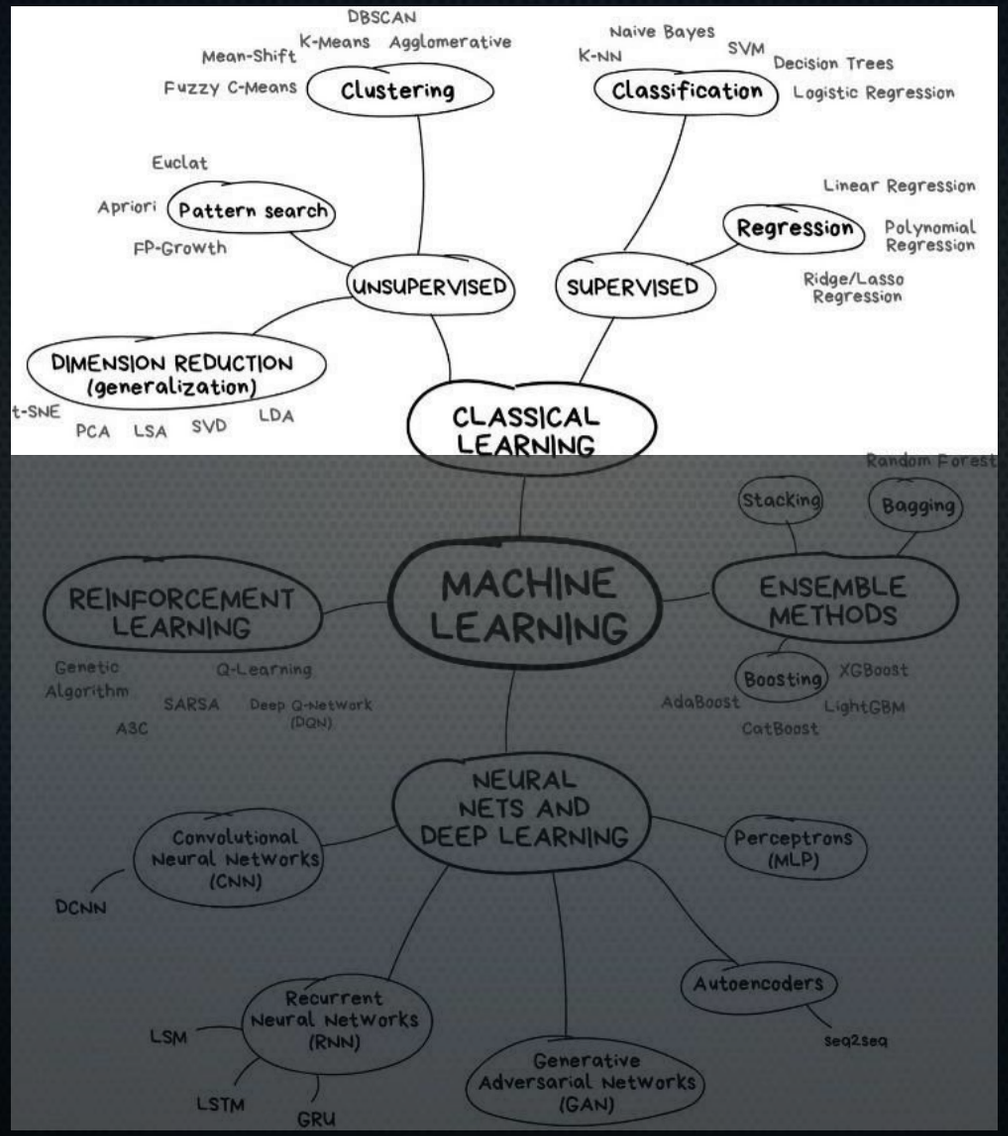


80% of your time

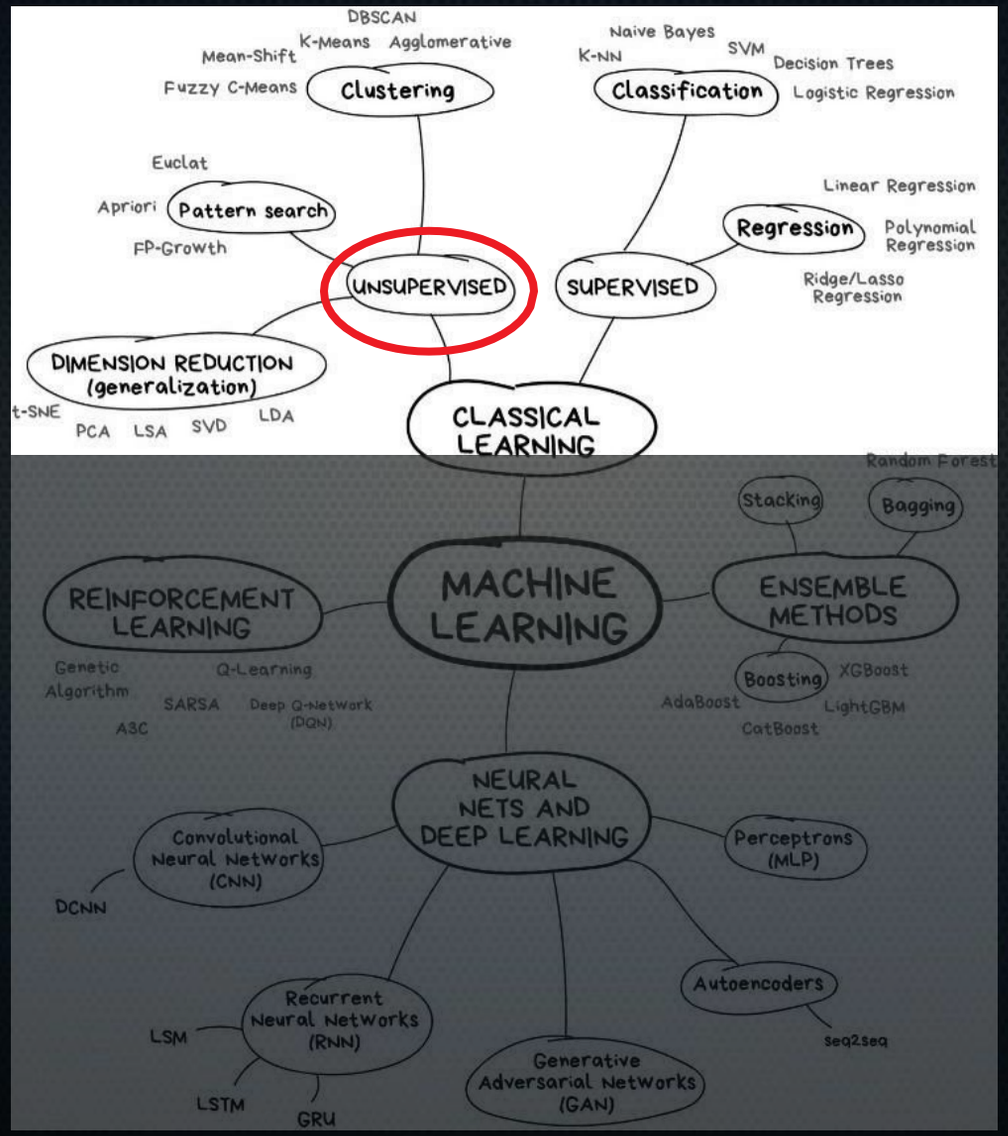
Map of Machine Learning



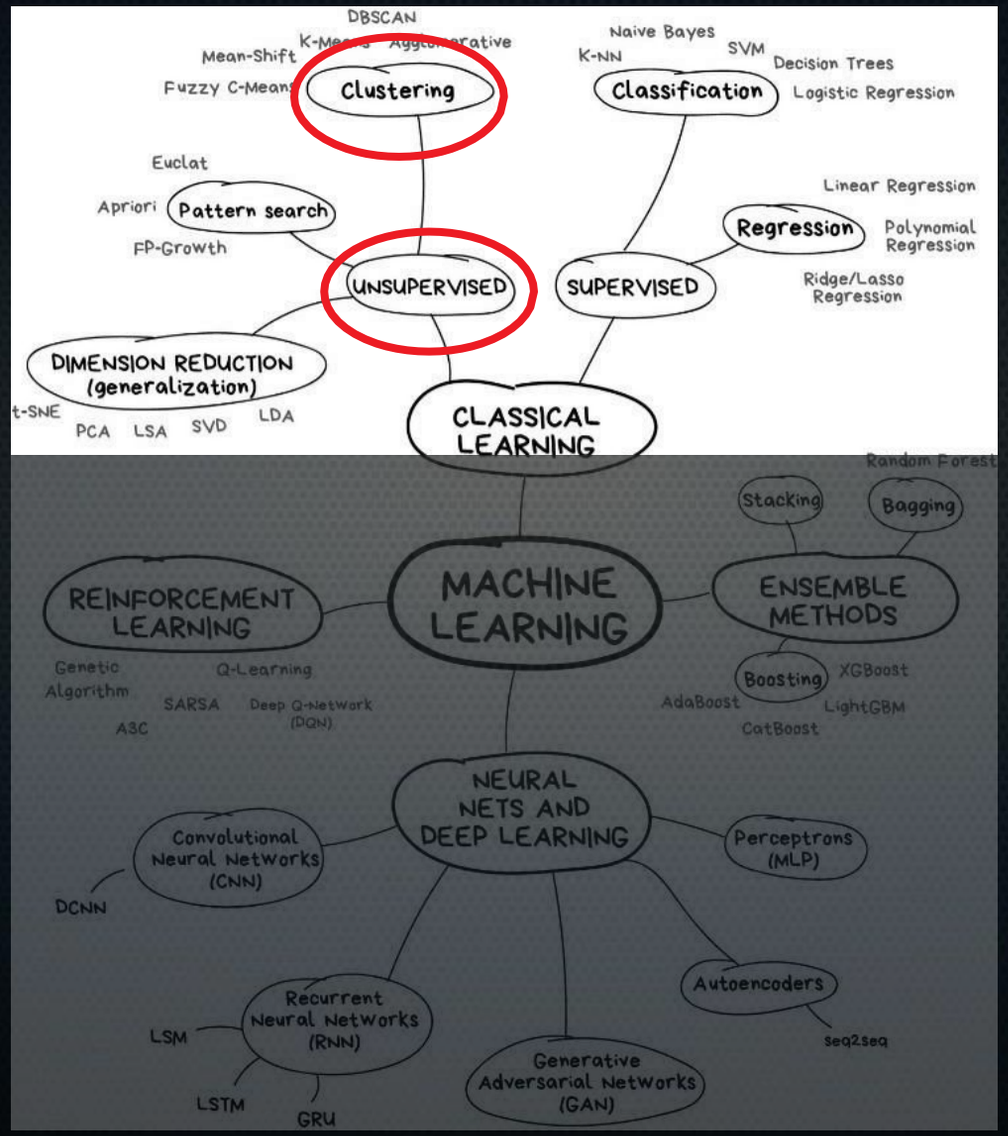
Map of Machine Learning



Map of Machine Learning

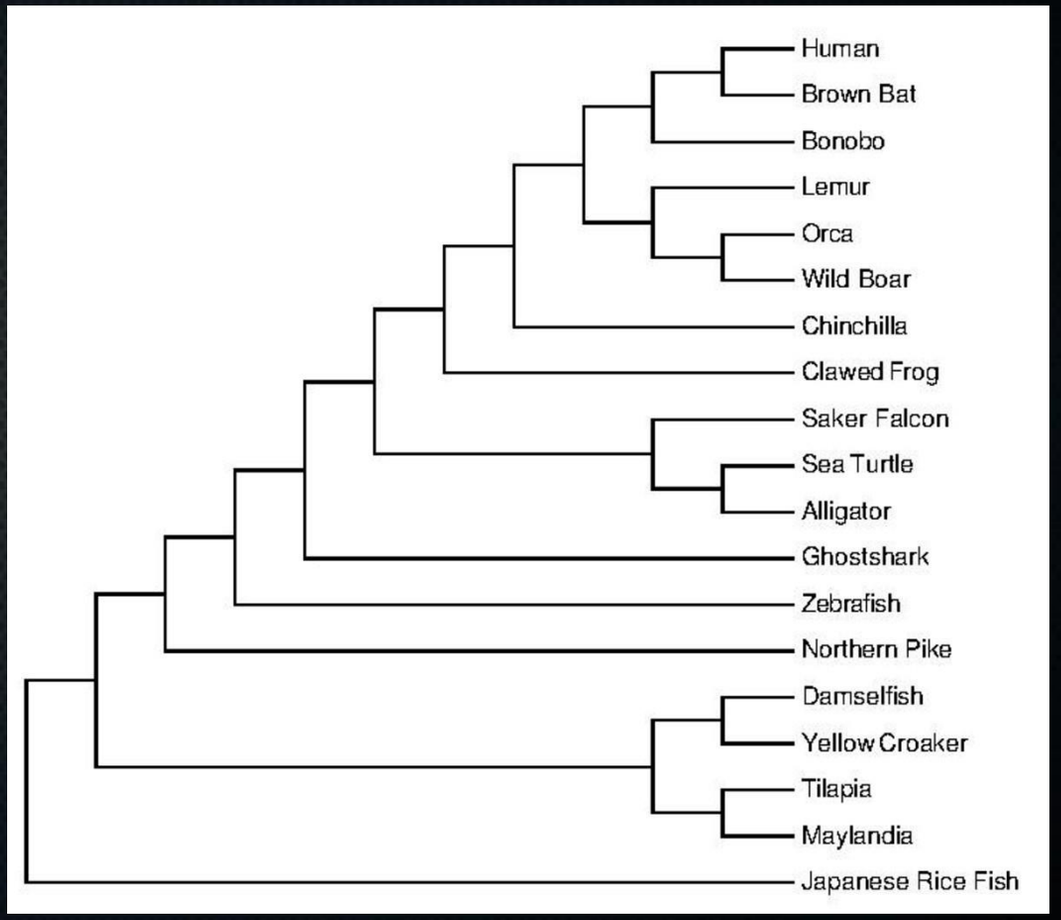


Map of Machine Learning



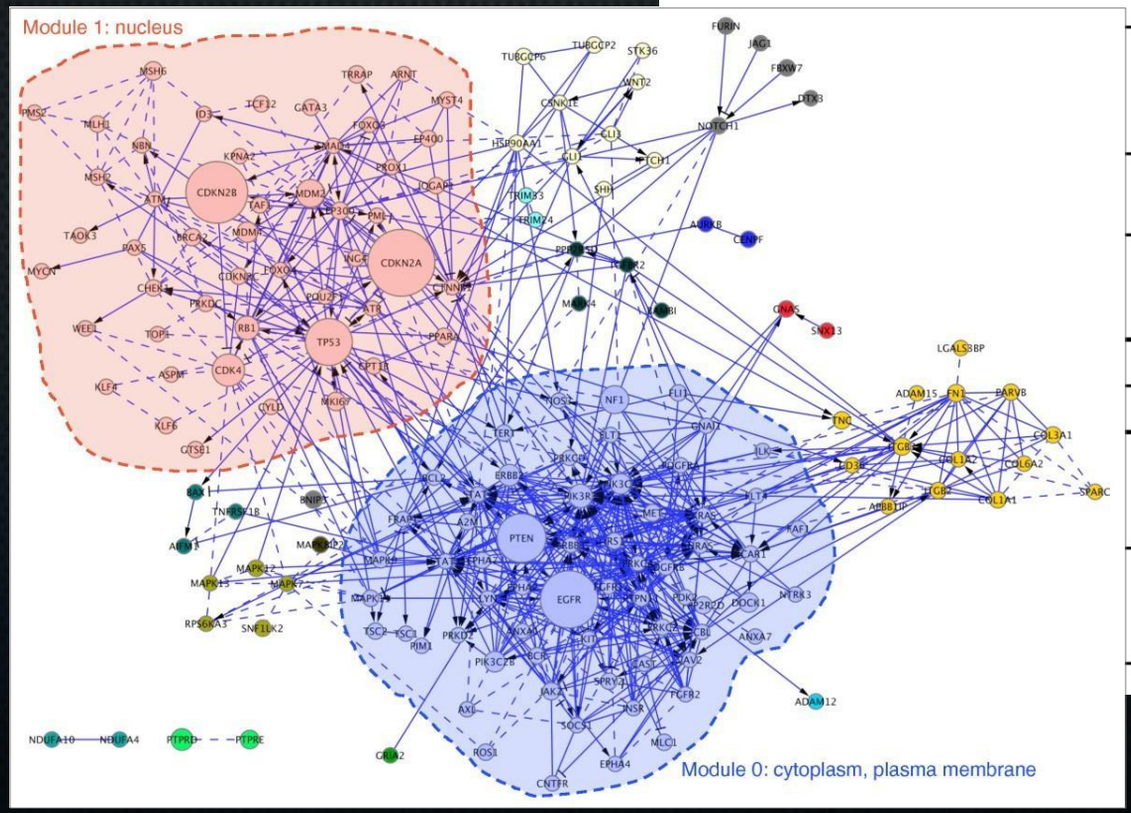
Unsupervised learning: clustering

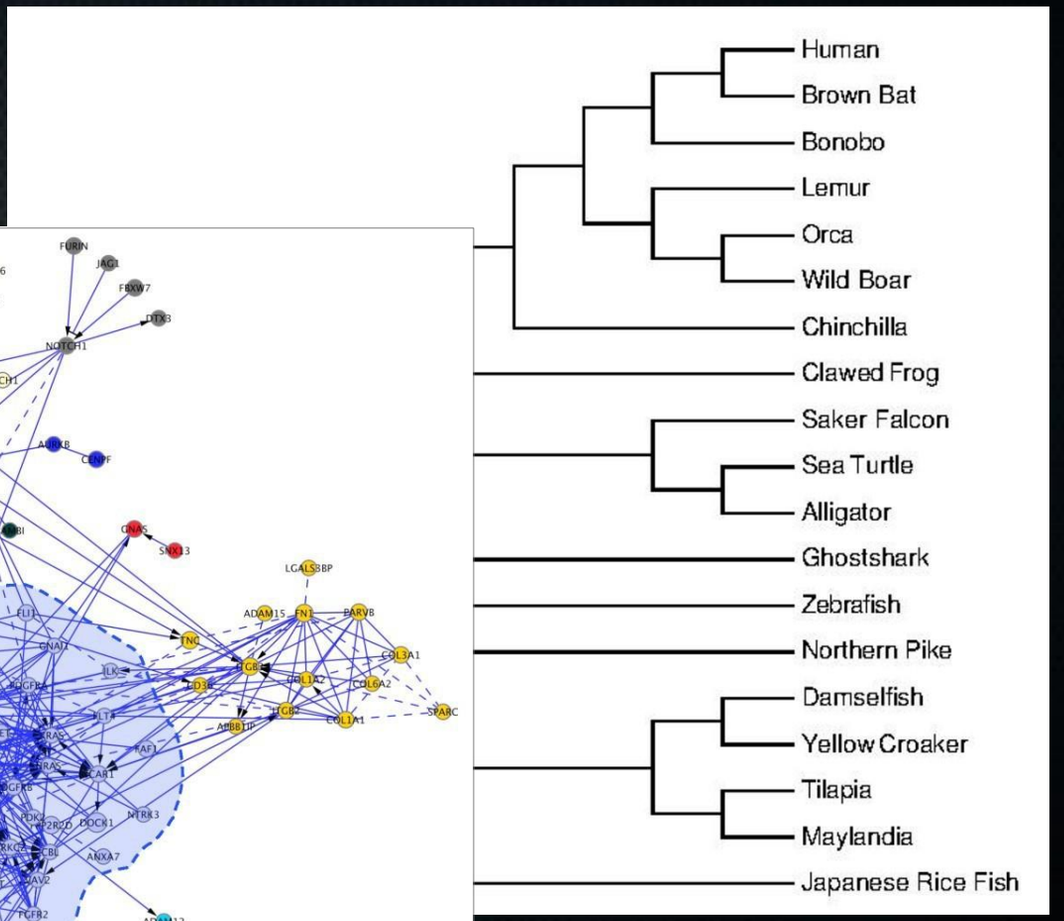
- Automatically find some structure in the data.



Unsupervised learning: clustering

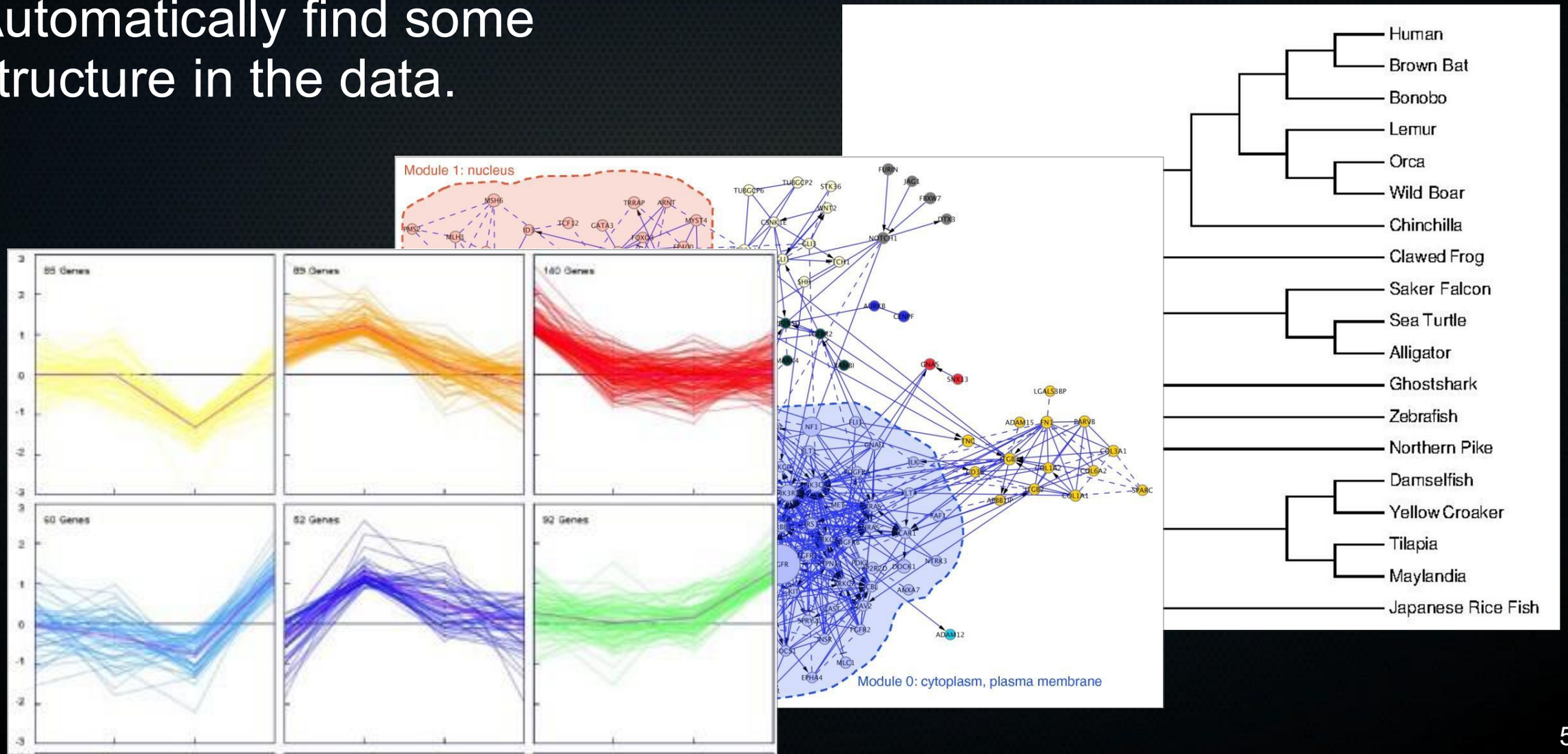
- Automatically find some structure in the data.





Unsupervised learning: clustering

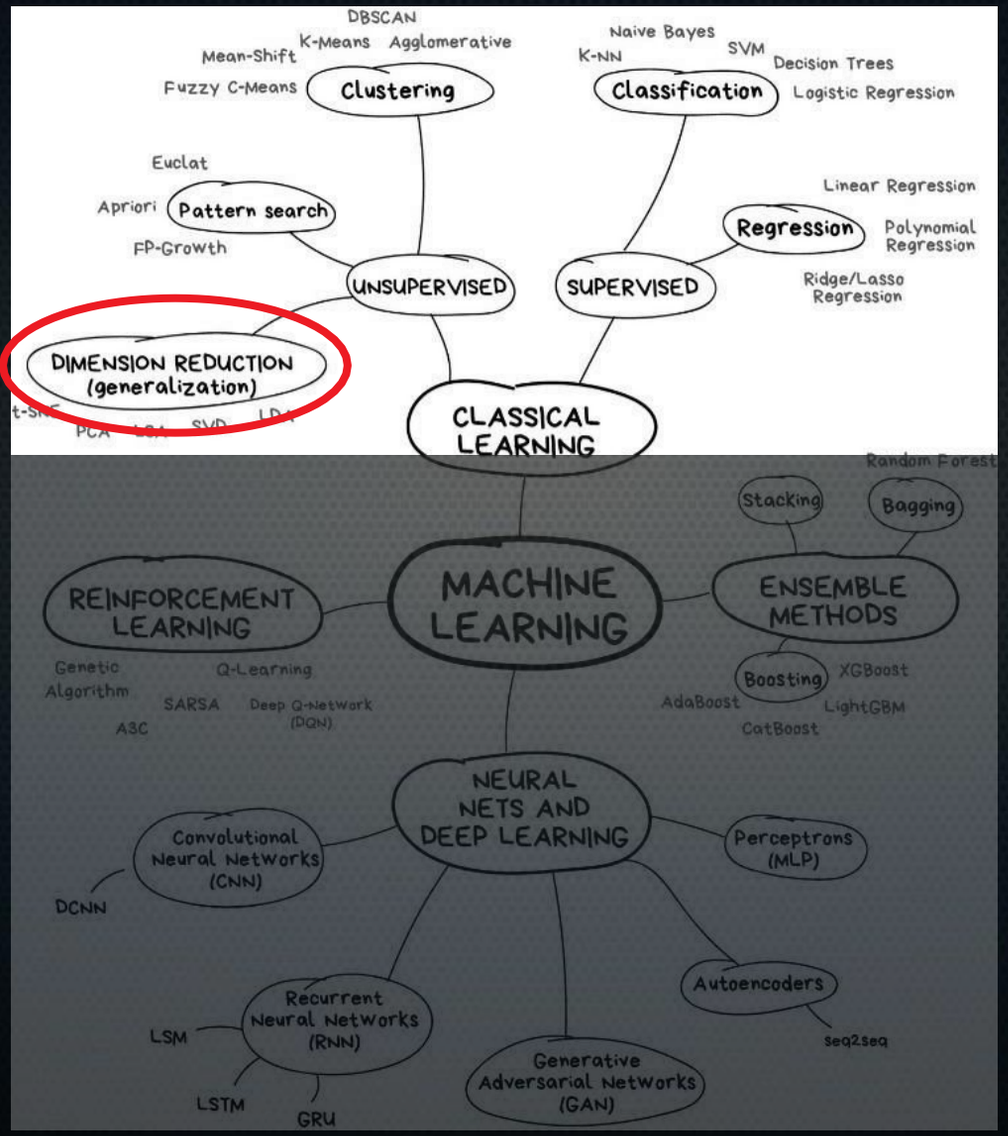
- Automatically find some structure in the data.



Unsupervised learning: clustering

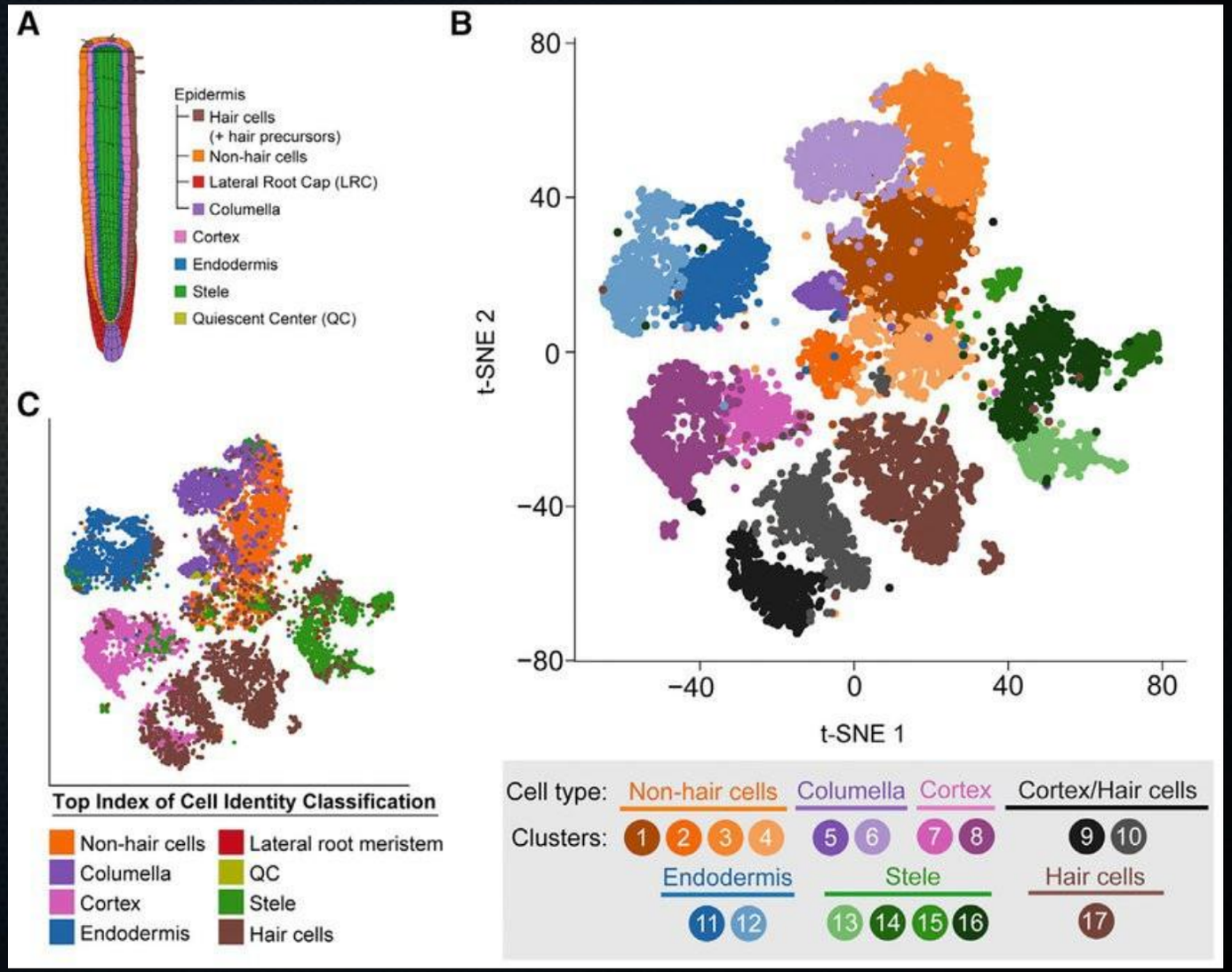
- No right or wrong:
  - Back-and-forth between different clustering algorithms, your knowledge, and the data.
  - You don‘t *know* correct clustering.

Map of machine learning



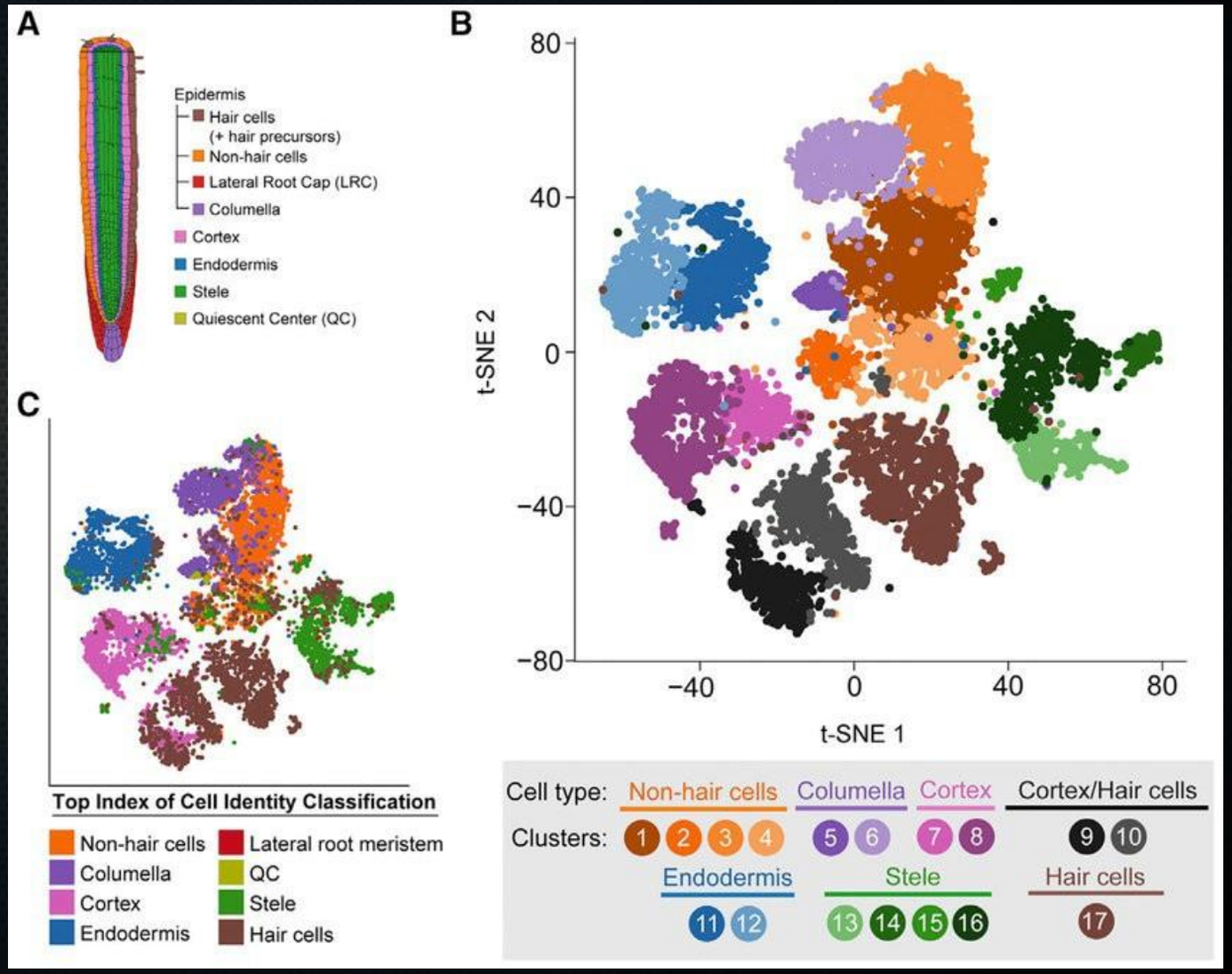
Unsupervised learning: dimensionality reduction

- Single-cell RNAseq of 12,198 Arabidopsis root cells.
- How do they differ?



Unsupervised learning: dimensionality reduction

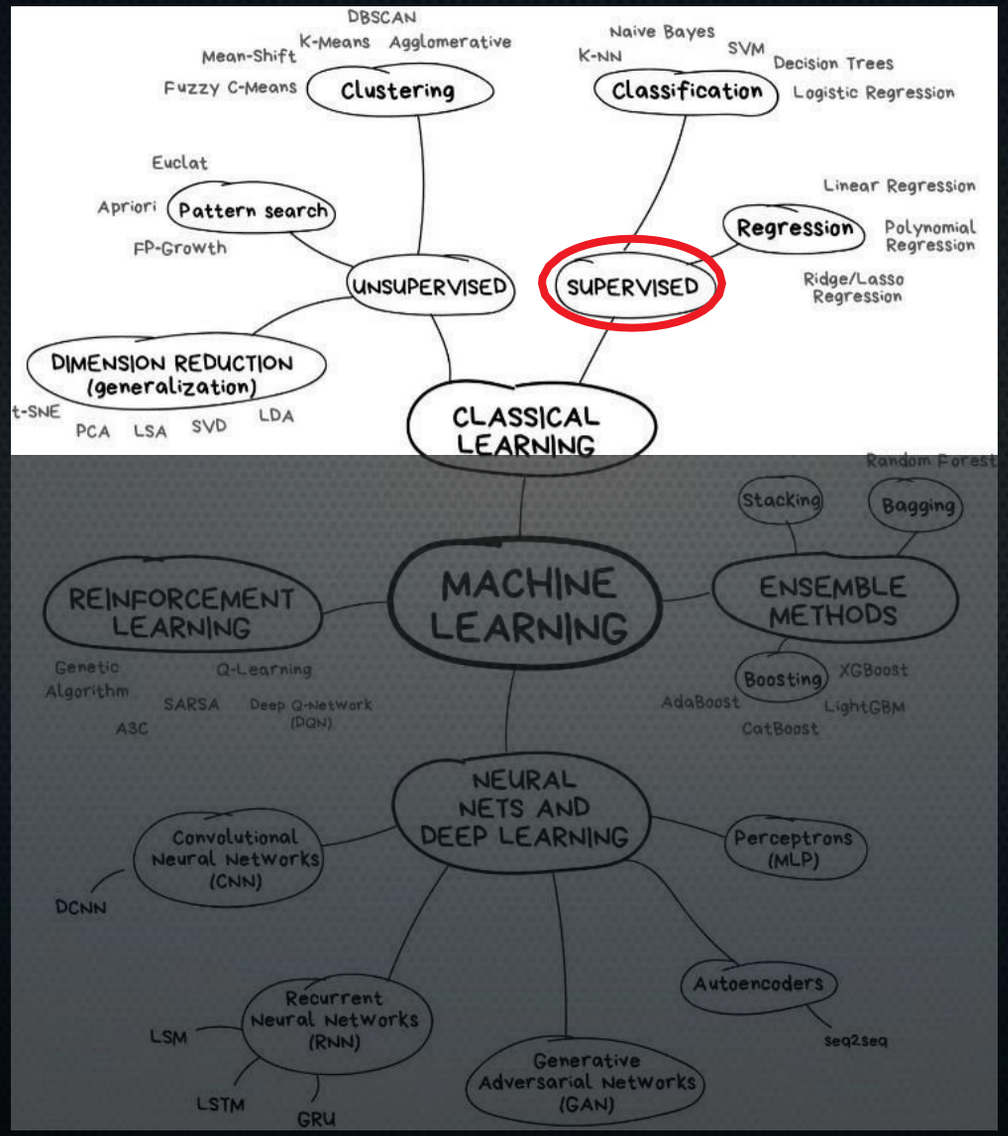
- Single-cell RNAseq of 12,198 Arabidopsis root cells.
- How do they differ?
- Visualise differences in all RNAs between all these cells in 2 dimensions.
- Used in conjunction with clustering (colours)



Unsupervised learning: dimensionality reduction

- Used for:
  - Visualisation (our visual systems cannot deal with > 3D)
  - Compressing data (capture 90% of the variation with much less data, say)
  - Preventing overfitting and other ill effects of high dimensionality

Map of Machine Learning

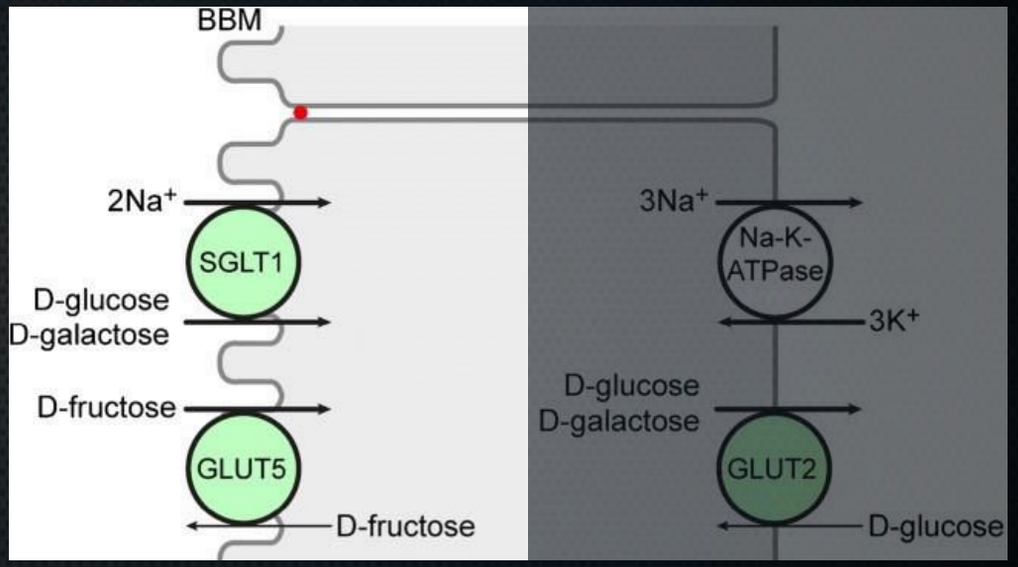


Supervised learning

- Given known examples, automatically find a function to map new examples.

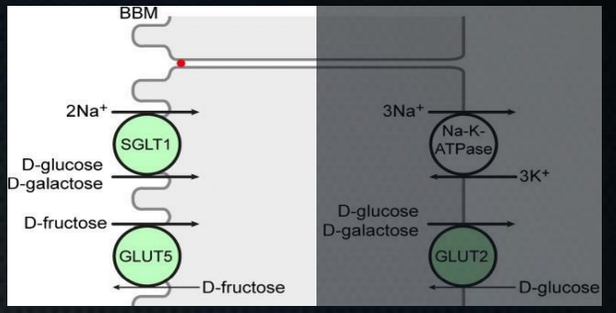
Supervised learning: regression

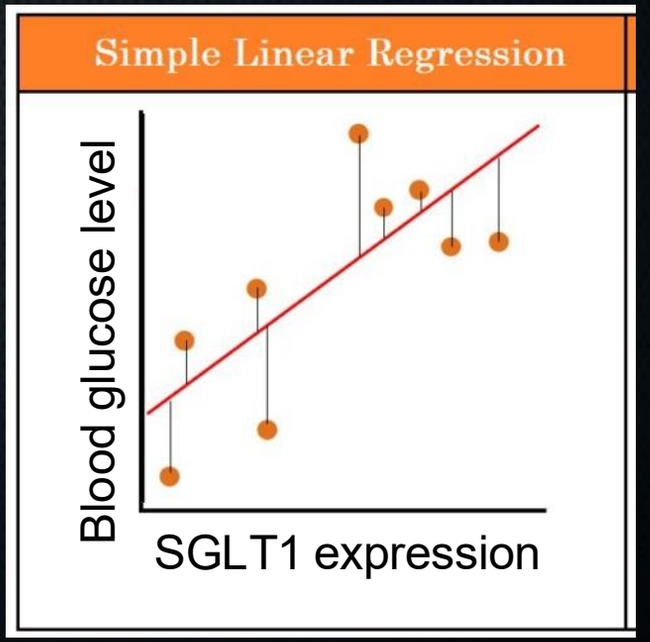
- Given known examples, automatically find a function to map new examples.
- Real-valued outputs.



Supervised learning: regression

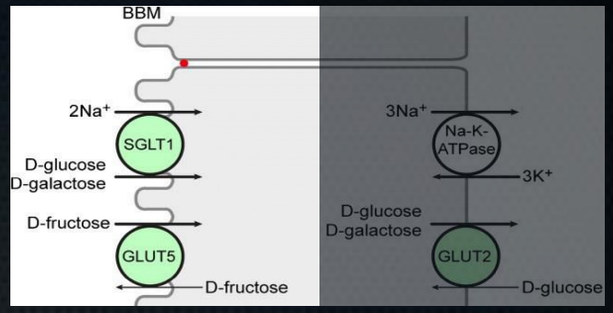
- Given known examples, automatically find a function to map new examples.
- Real-valued outputs.

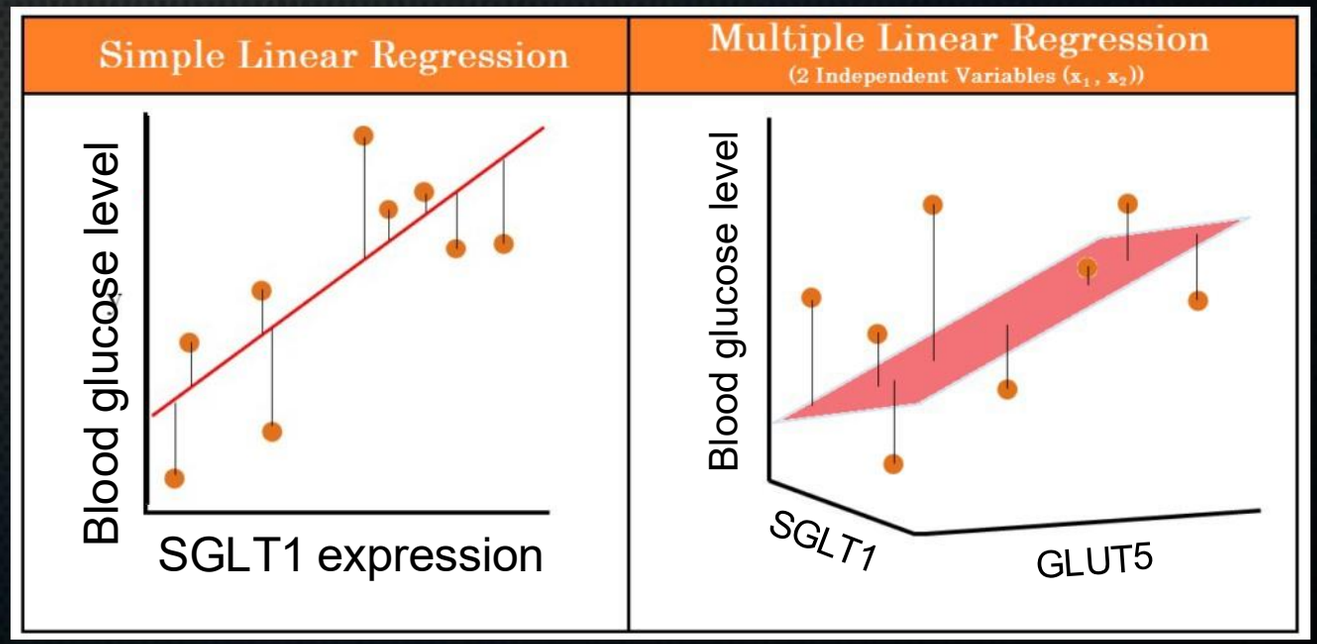




Supervised learning: regression

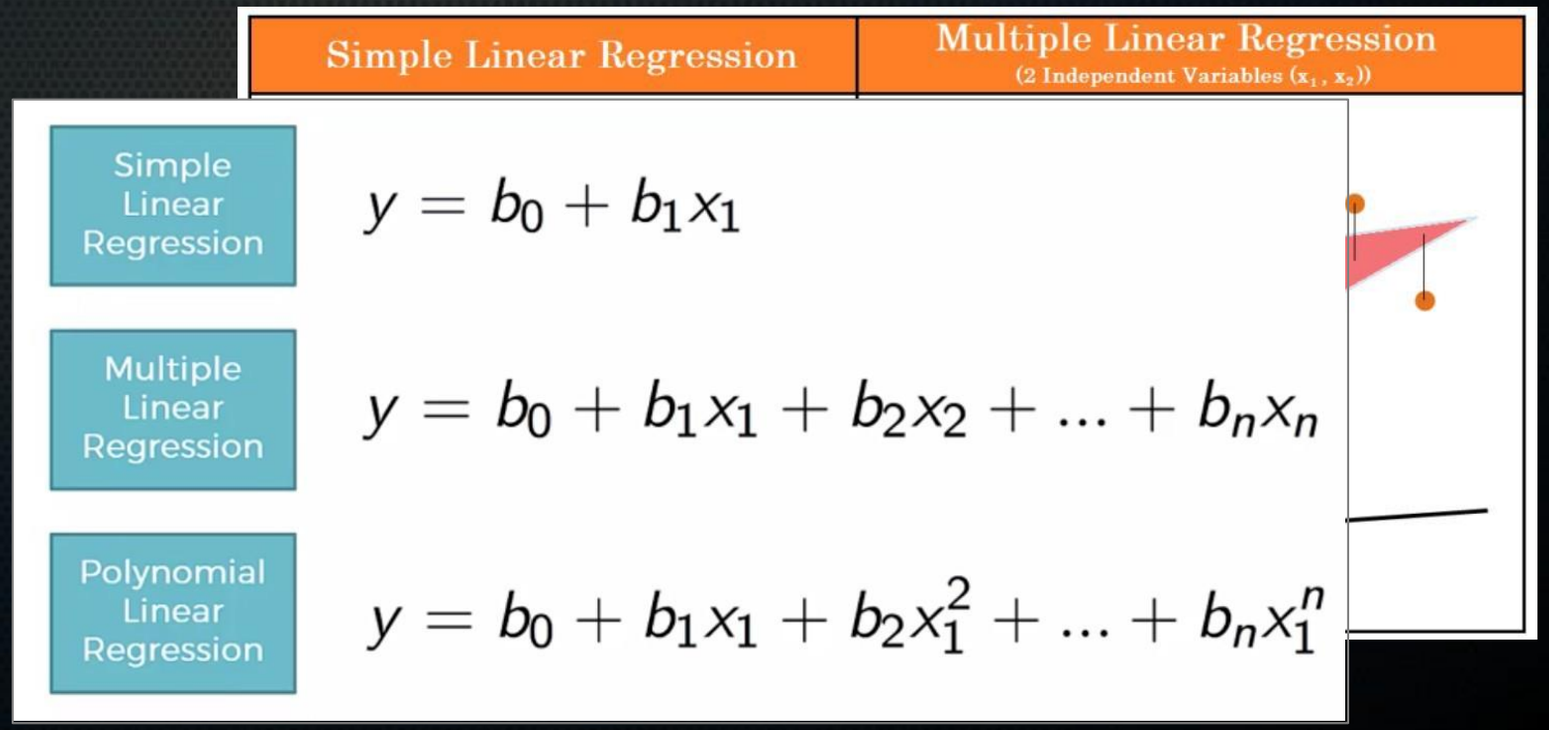
- Given known examples, automatically find a function to map new examples.
- Real-valued outputs.





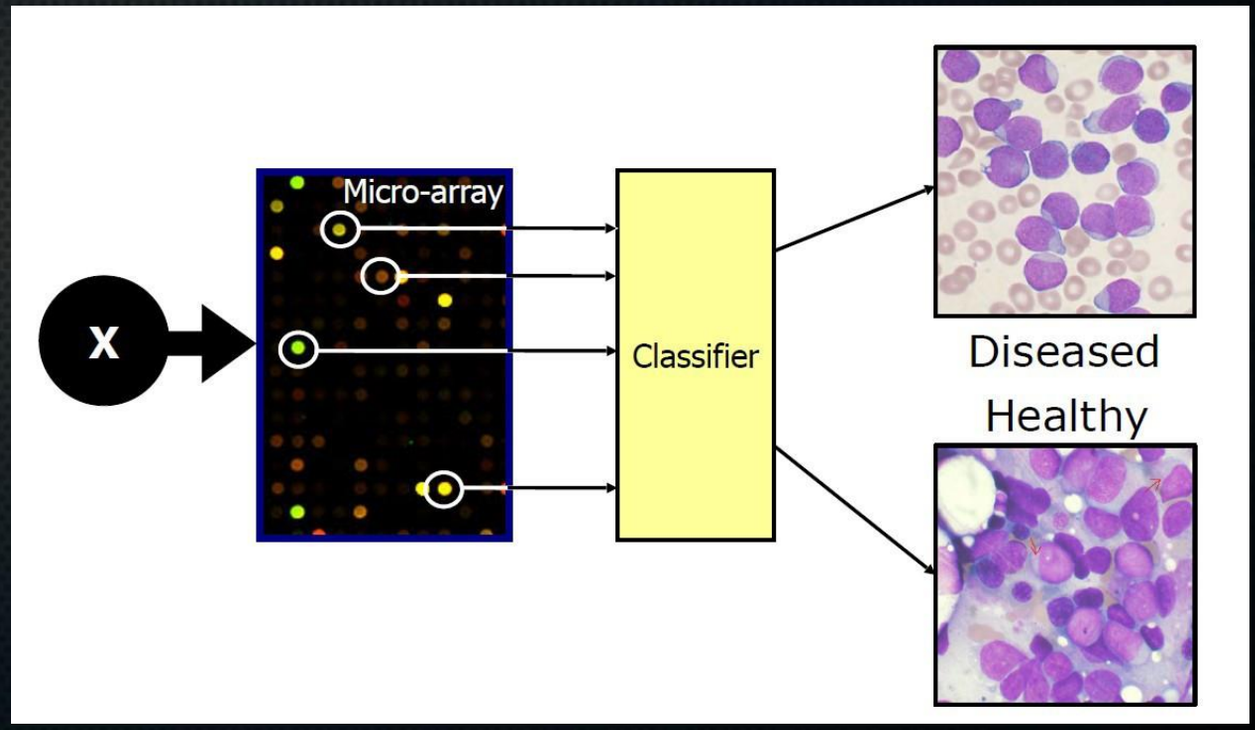
Supervised learning: regression

- Given known examples, automatically find a function to map new examples.
- Real-valued outputs: regression.



Supervised learning: classification

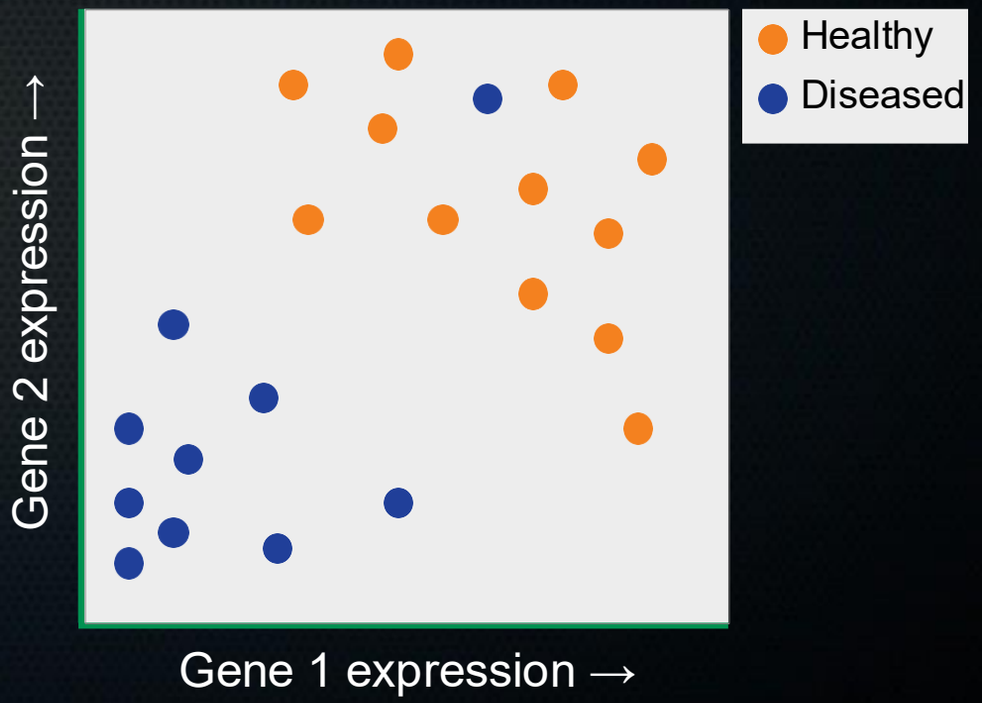
- Given known examples, automatically find a function to map new examples.
- Discrete outputs.



Source: Jeroen de Ridder

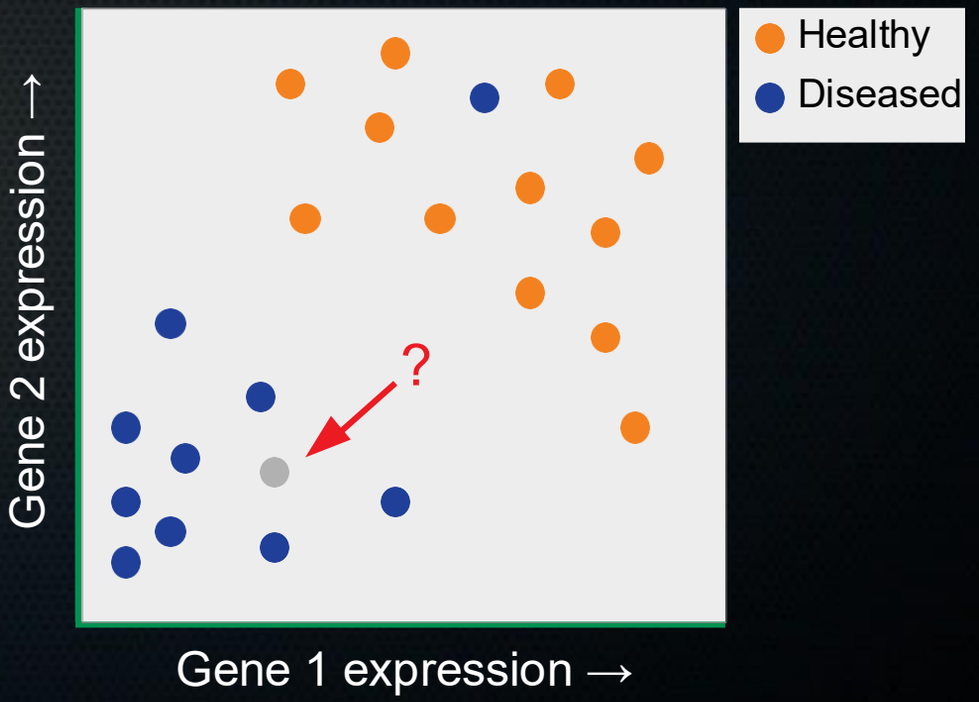
Supervised learning: classification

- Given known examples, automatically find a function to map new examples.
- Discrete outputs.



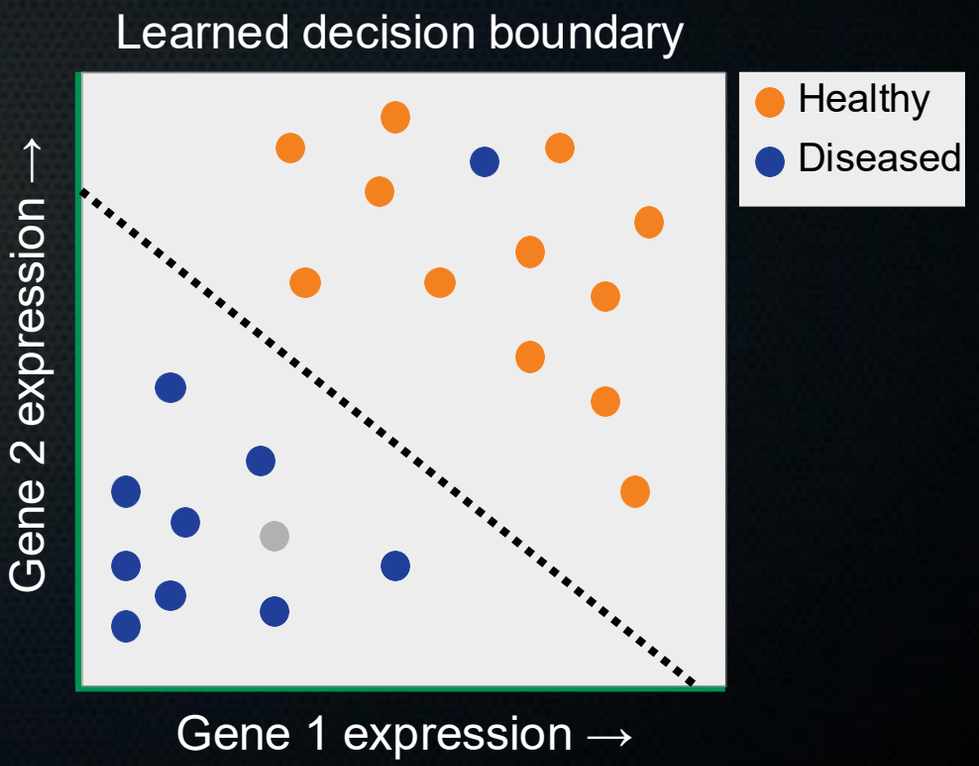
Supervised learning: classification

- Given known examples, automatically find a function to map new examples.
- Discrete outputs.



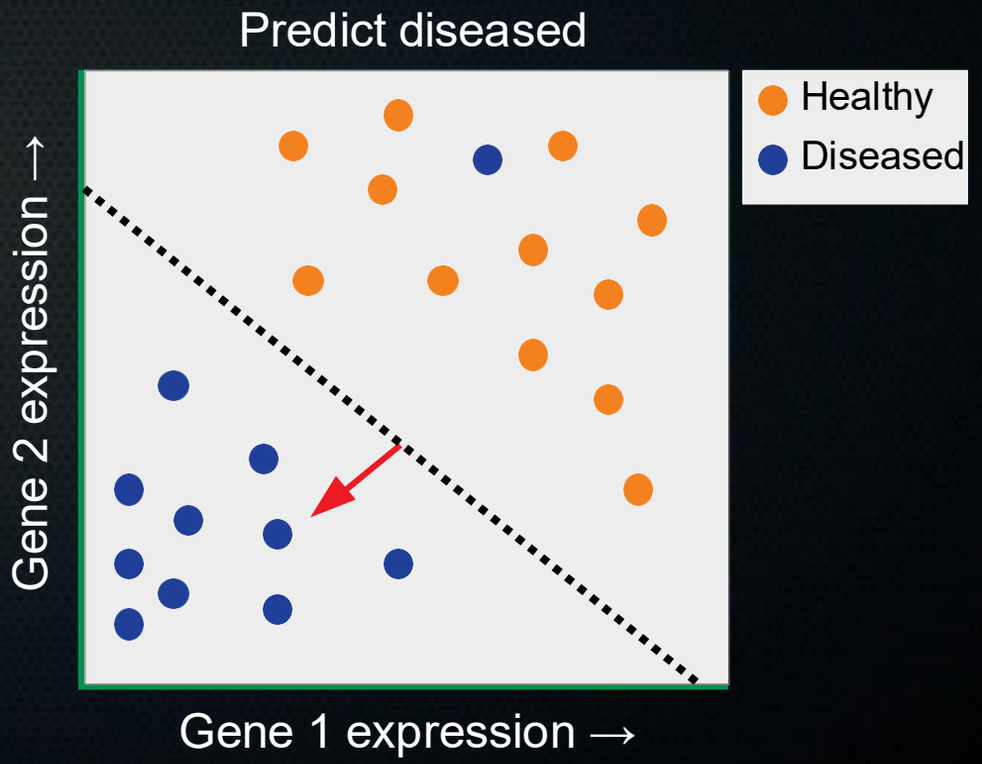
Supervised learning: classification

- Given known examples, automatically find a function to map new examples.
- Discrete outputs.

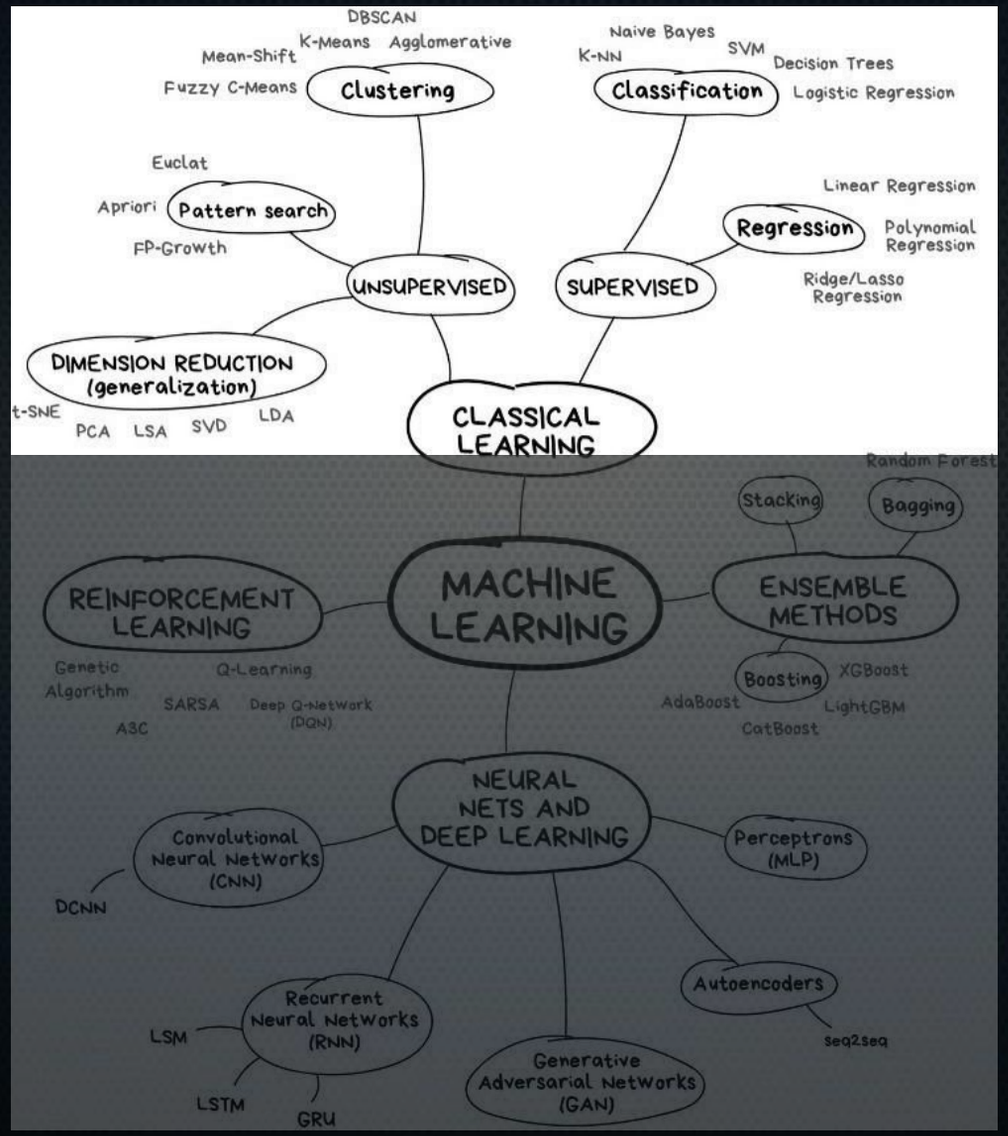


Supervised learning: classification

- Given known examples, automatically find a function to map new examples.
- Discrete outputs.



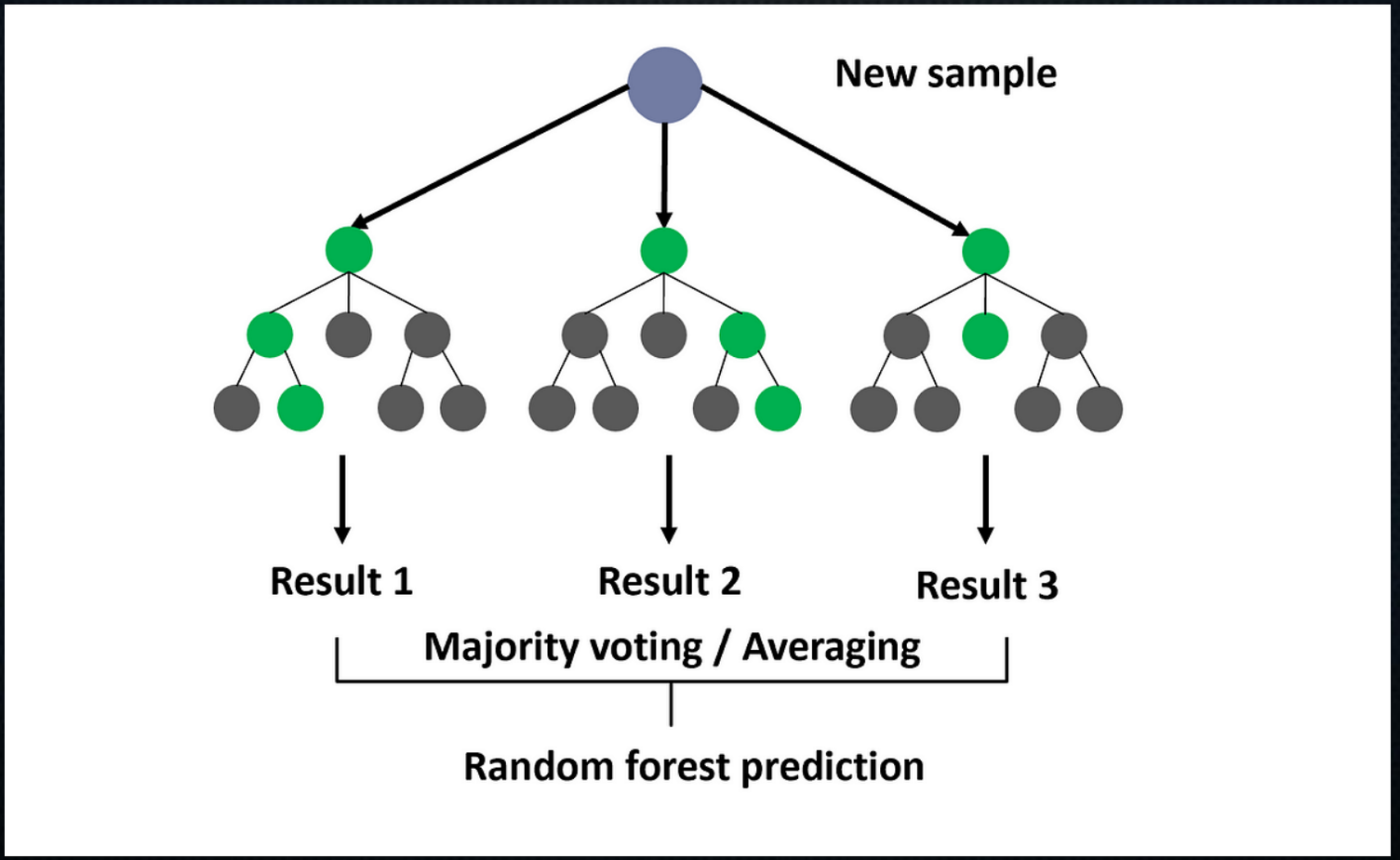
Map of Machine Learning



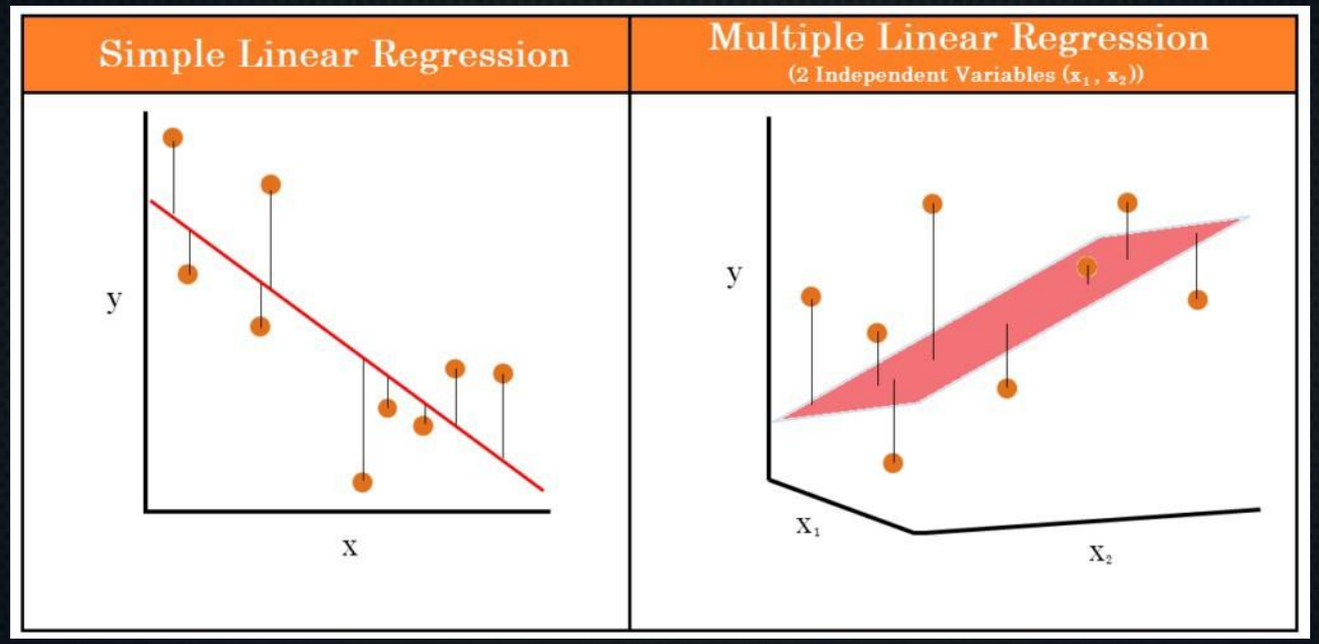
Summary

- Unsupervised:
  - Find some structure in your data (clusters, projection into lower-dimensional space that captures much of variance)
  - Don‘t know the „correct“ structure (no *labels*)
- Supervised:
  - Automatically learn a function that maps *features* (e.g. gene expression) to a real-valued output (regression, e.g. blood glucose level) or discrete *classes/labels* (classification, e.g. healthy/diseased) using training data for which you know these outcomes.

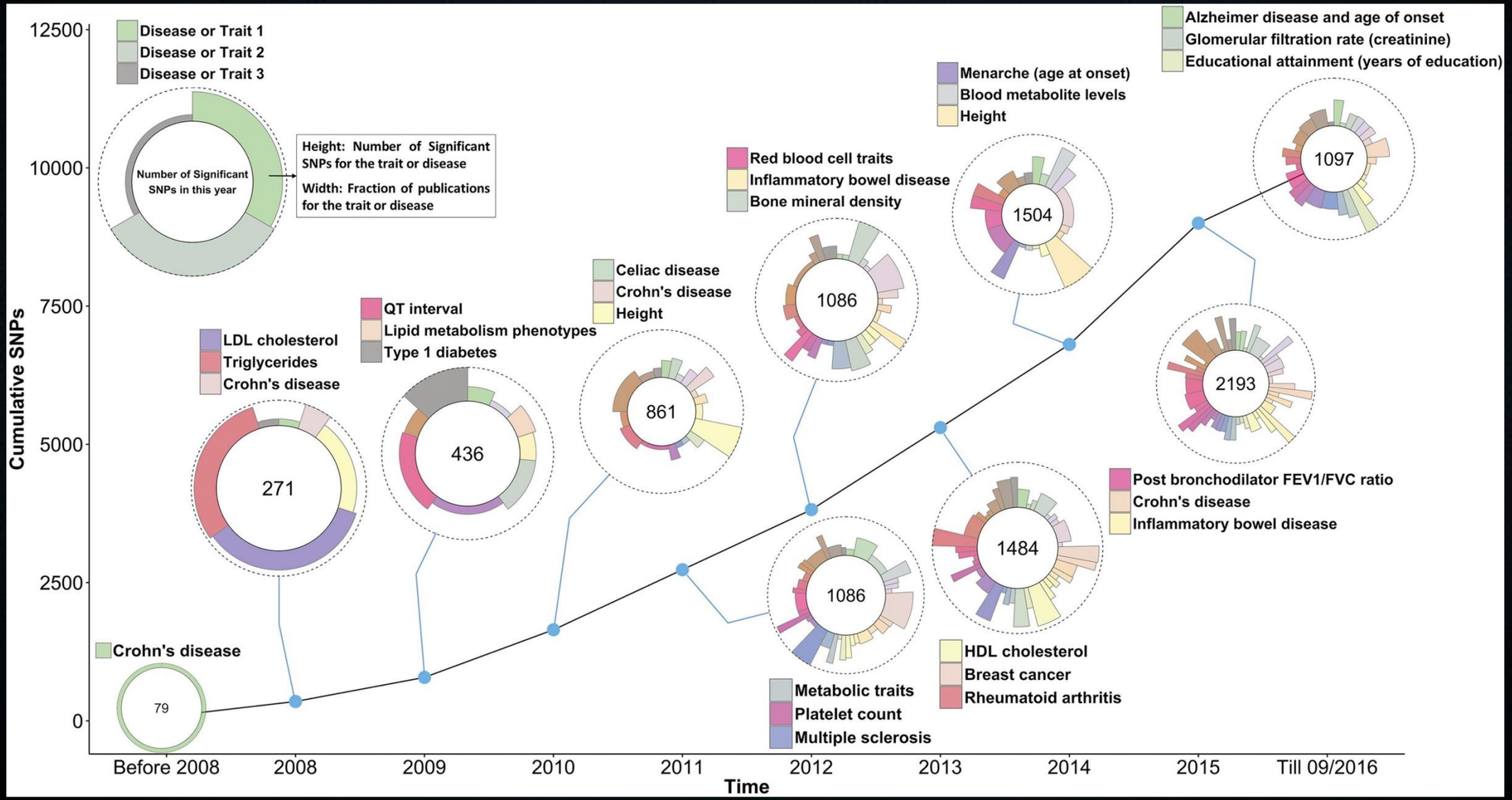
Regression or classification?



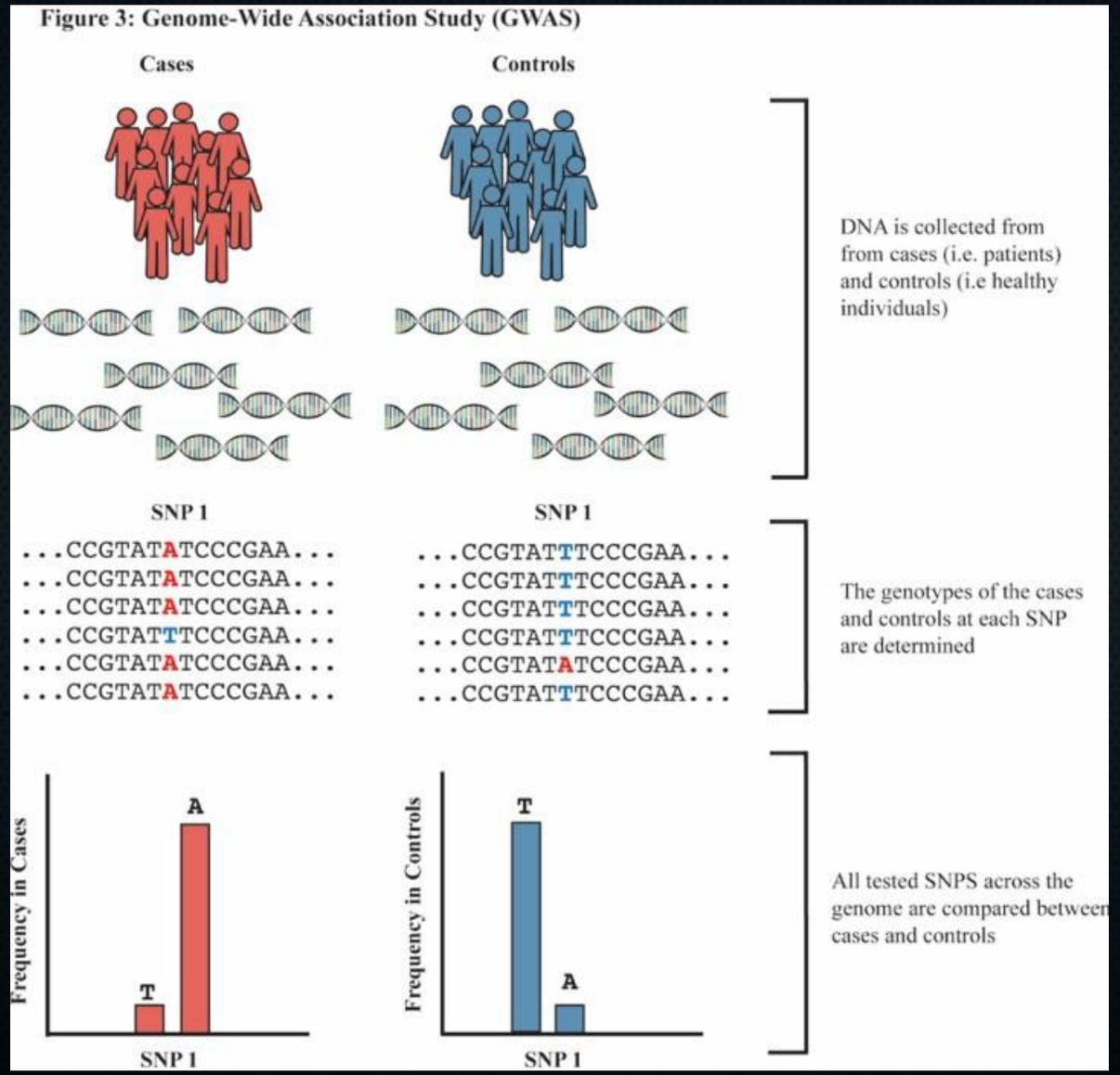
Linear regression



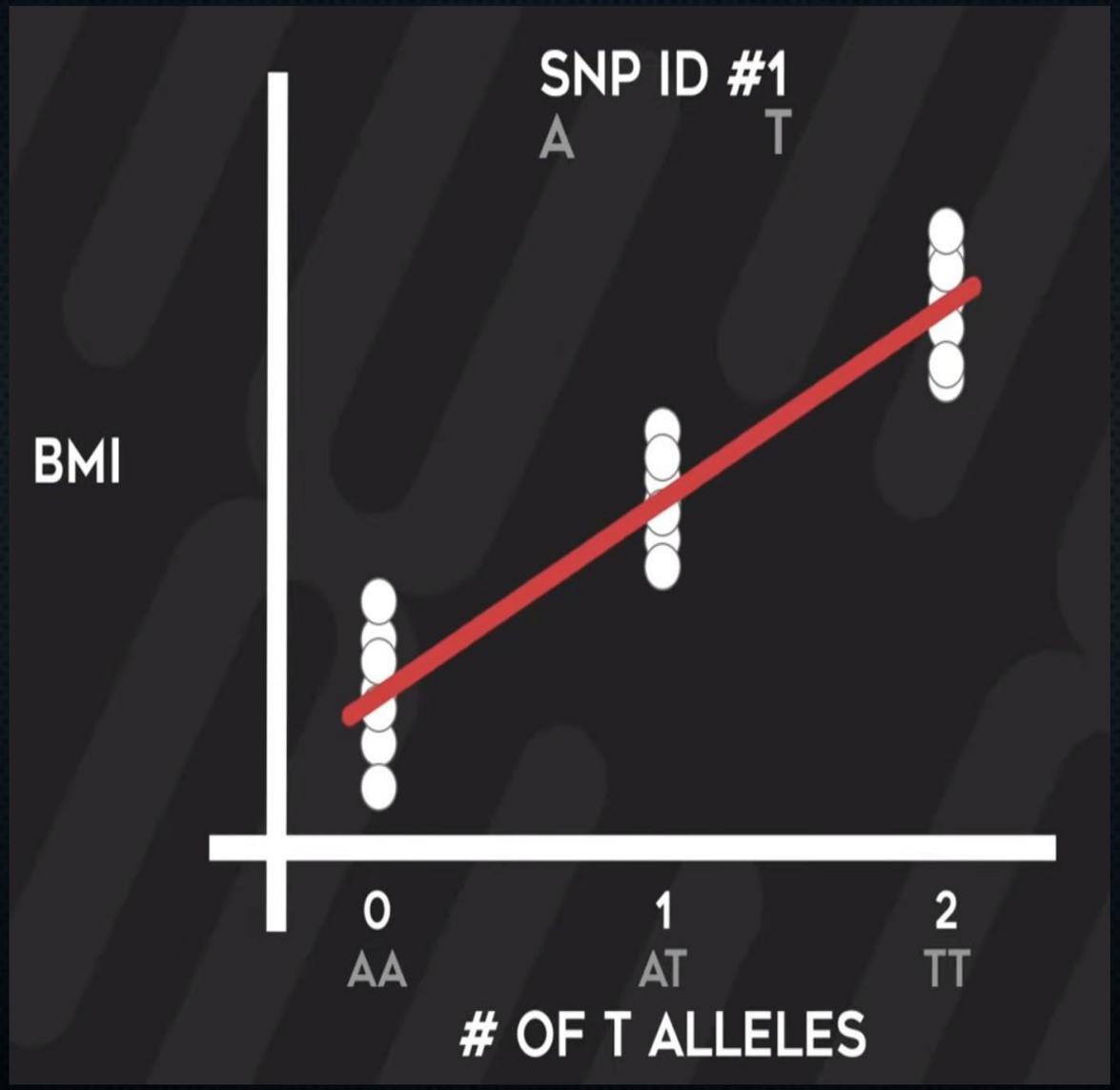
Linear regression: GWAS



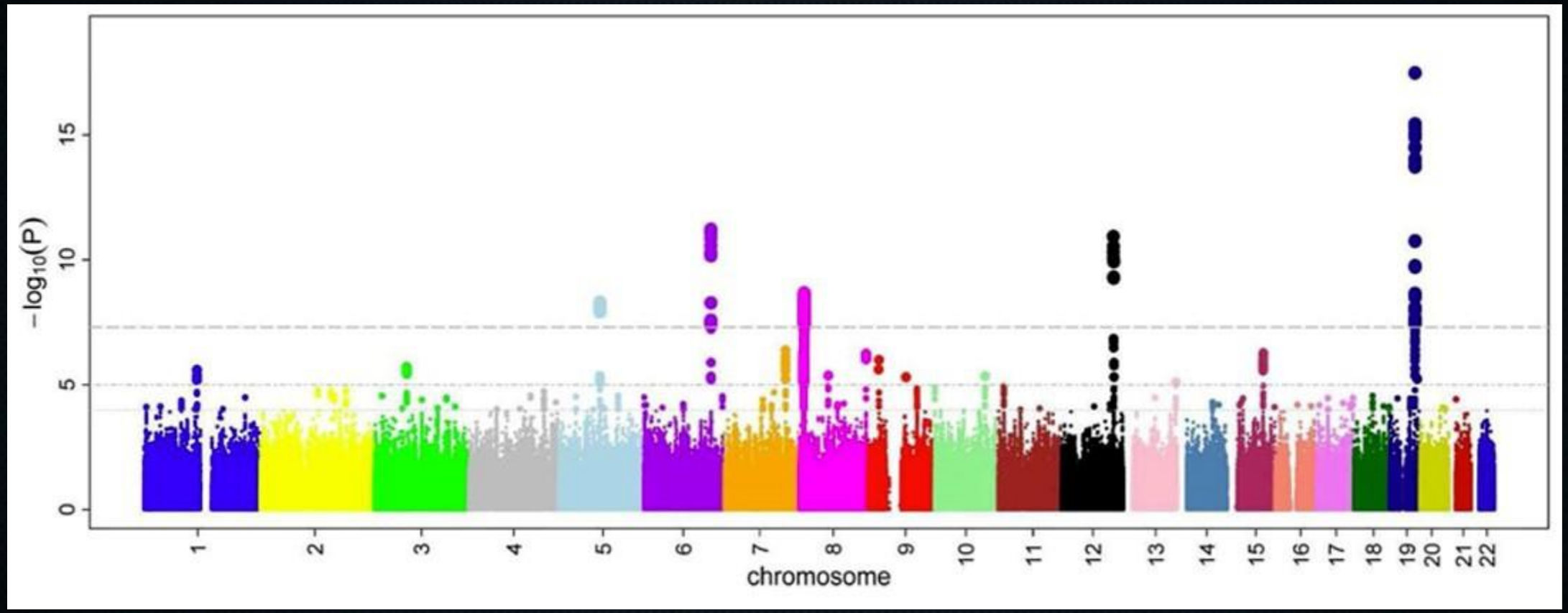
Linear regression: GWAS



Linear regression: GWAS

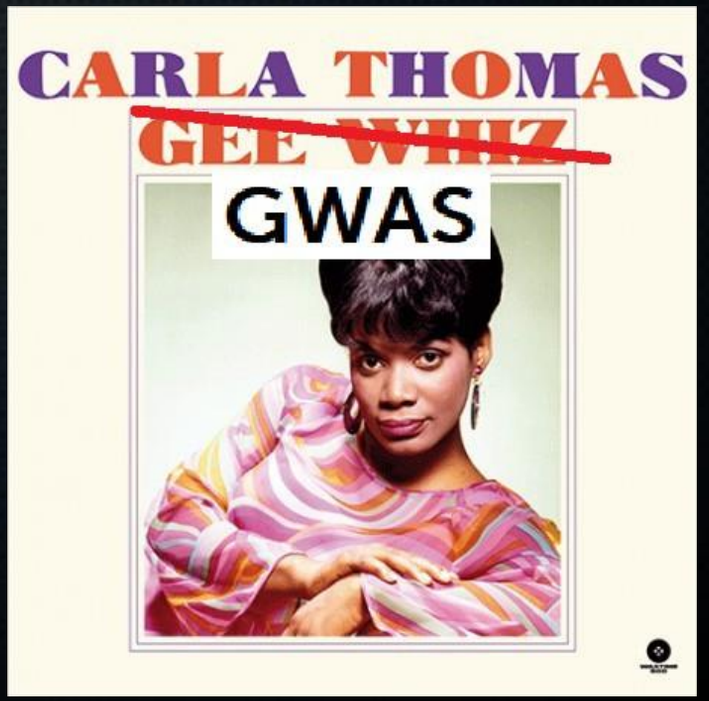


Linear regression: GWAS



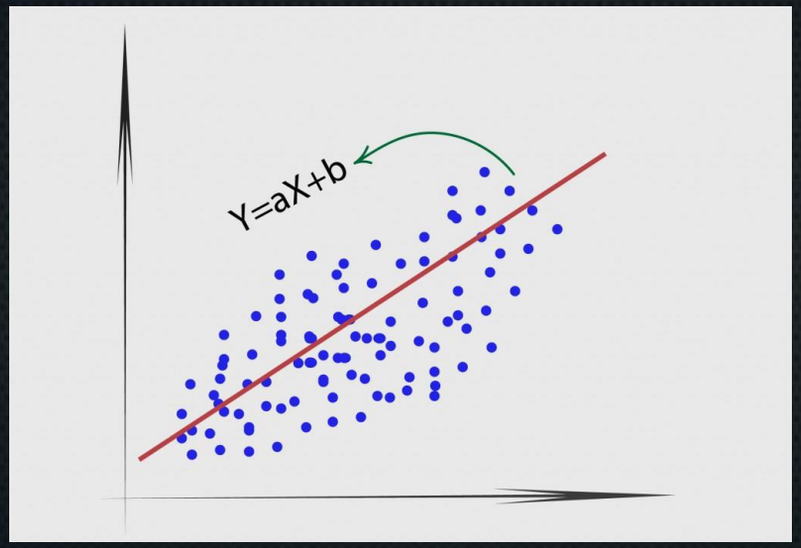
Linear regression: GWAS

- Connect SNPs to nearby genes (non-trivial!)
- Yielded huge advances in our knowledge on many complex diseases over the past ~15 years.



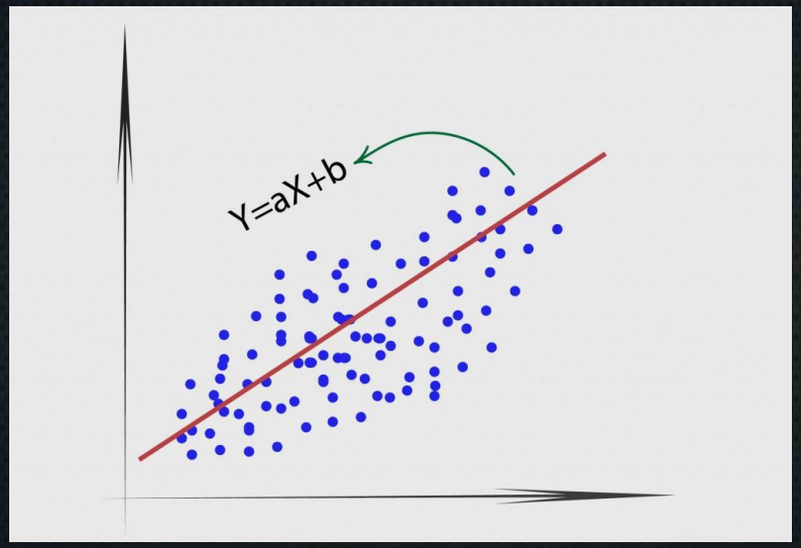
Univariate linear regression

$y = ax + b$



Univariate linear regression

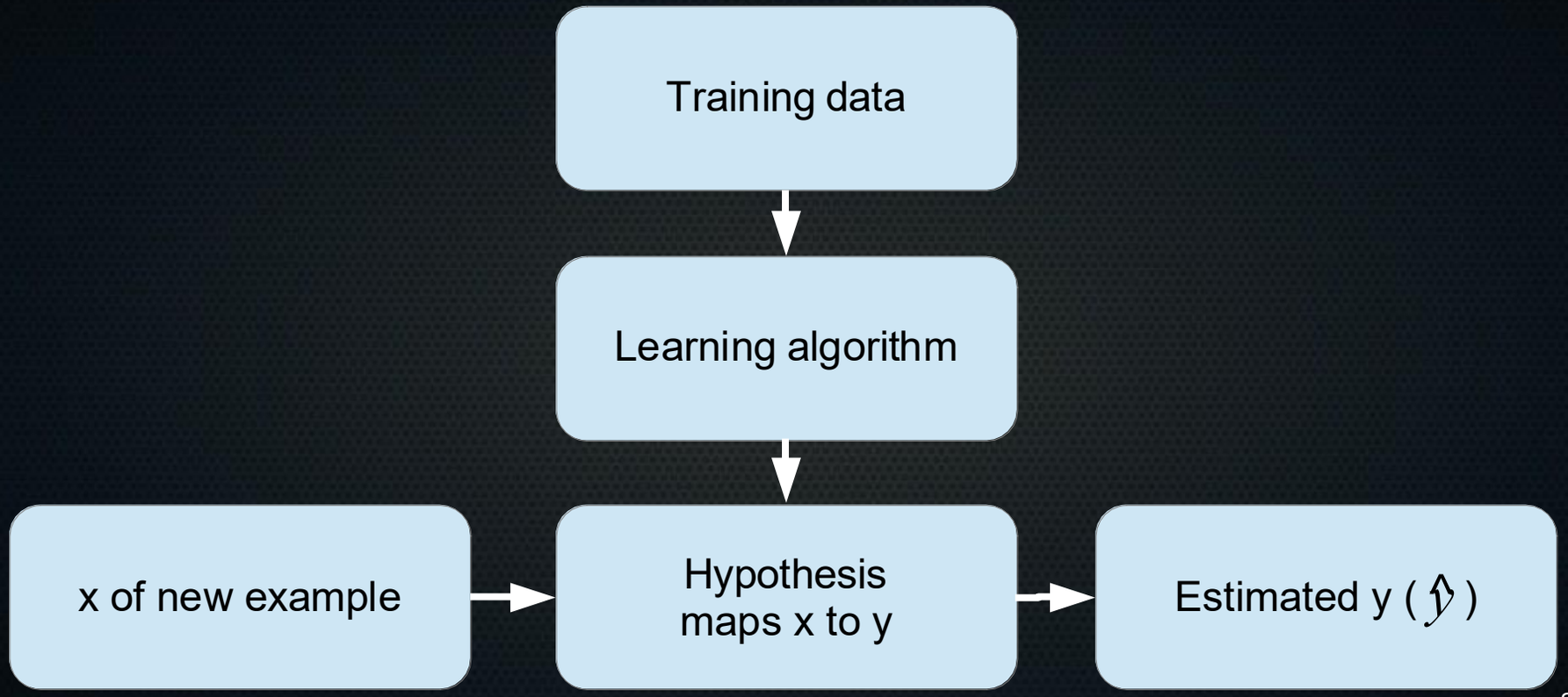
$y=ax+b$



$y=\theta_0+\theta_1 x$

Process

$y=ax+b$



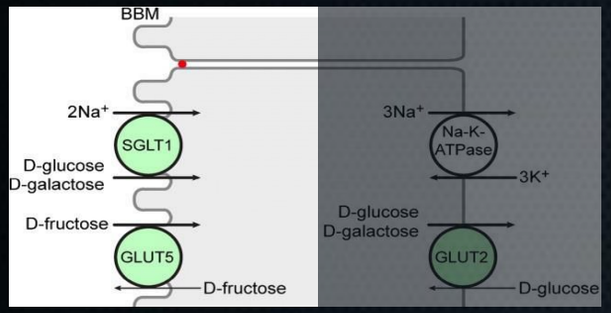
Source: Andrew Ng, Coursera

$$h_\theta(x)=\theta_0+\theta_1 x$$

Terminology alert! #StatQuest BAM

$y = ax + b$

Training data

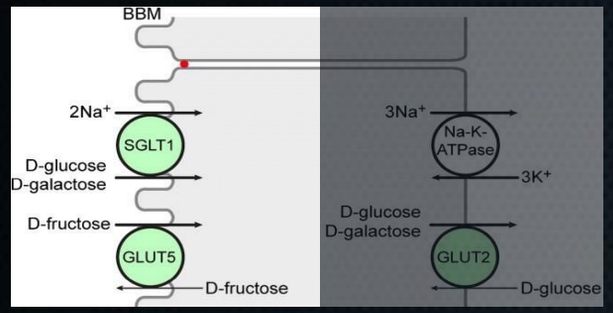


Sample #	SGLT1 expression level (arbitrary units relative to housekeeping gene)	Blood glucose level (mg/dL)
1	3	80
2	8	130
3	12	170
4	2	89

Terminology

$y = ax + b$

Training data



m = # of training examples

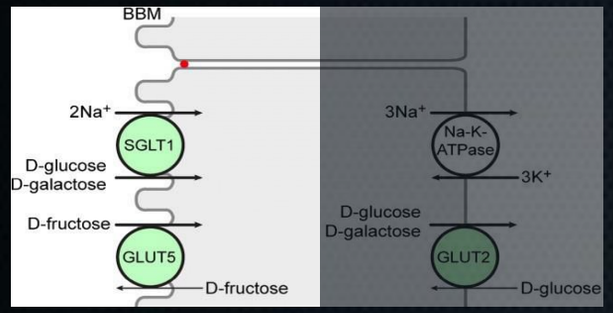
m

Sample #	SGLT1 expression level (arbitrary units relative to housekeeping gene)	Blood glucose level (mg/dL)
1	3	80
2	8	130
3	12	170
4	2	89

Terminology

$y = ax + b$

Training data



m = # of training examples
n = # of features/variables

Sample #	SGLT1 expression level (arbitrary units relative to housekeeping gene)	Blood glucose level (mg/dL)
1	3	80
2	8	130
3	12	170
4	2	89

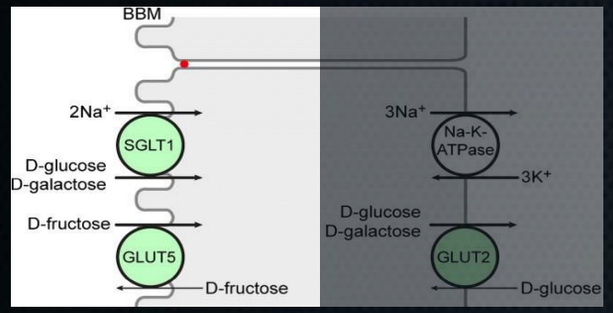
m

n

Terminology

$y = ax + b$

Training data



m = # of training examples
n = # of features/variables

Sample #	x1	x2	x3	Blood glucose level (mg/dL)
1	3	2	11	80
2	8	3	2	130
3	12	4	666	170
4	2	6	5	89

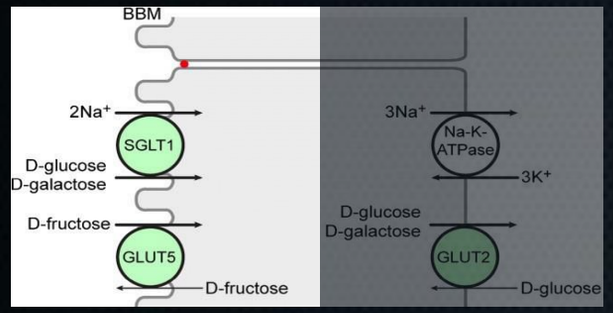
m

n

Terminology

$y = ax + b$

Training data



m = # of training examples
n = # of features/variables
y = output variable/target variable or label (classification)

m

Sample #	SGLT1 expression level (arbitrary units relative to housekeeping gene)	Blood glucose level (mg/dL)
1	3	80
2	8	130
3	12	170
4	2	89

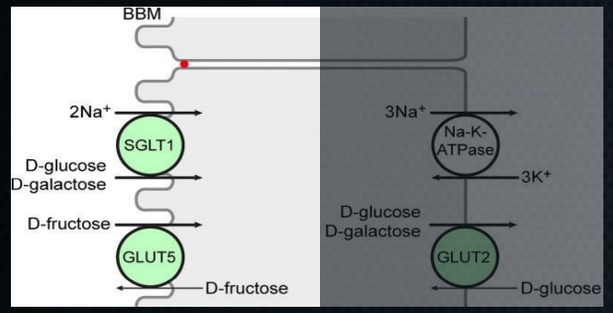
n

y

Terminology

$y = ax + b$

Training data



m = # of training examples
n = # of features/variables
y = output variable/target variable or label (classification)
$(x^{(i)}, y^{(i)})$ = i-th training example

Sample #	SGLT1 expression level (arbitrary units relative to housekeeping gene)	Blood glucose level (mg/dL)
1	3	80
2	8	130
3	12	170
4	2	89

m

n

y

$(x^{(3)}, y^{(3)})$

Cost function and gradient descent

- How to learn theta‘s from data?
- Two parts
  - How wrong are we for given parameters?
  - How do we update our parameters, given how wrong we are?

$$h_\theta(x)=\theta_0+\theta_1 x$$

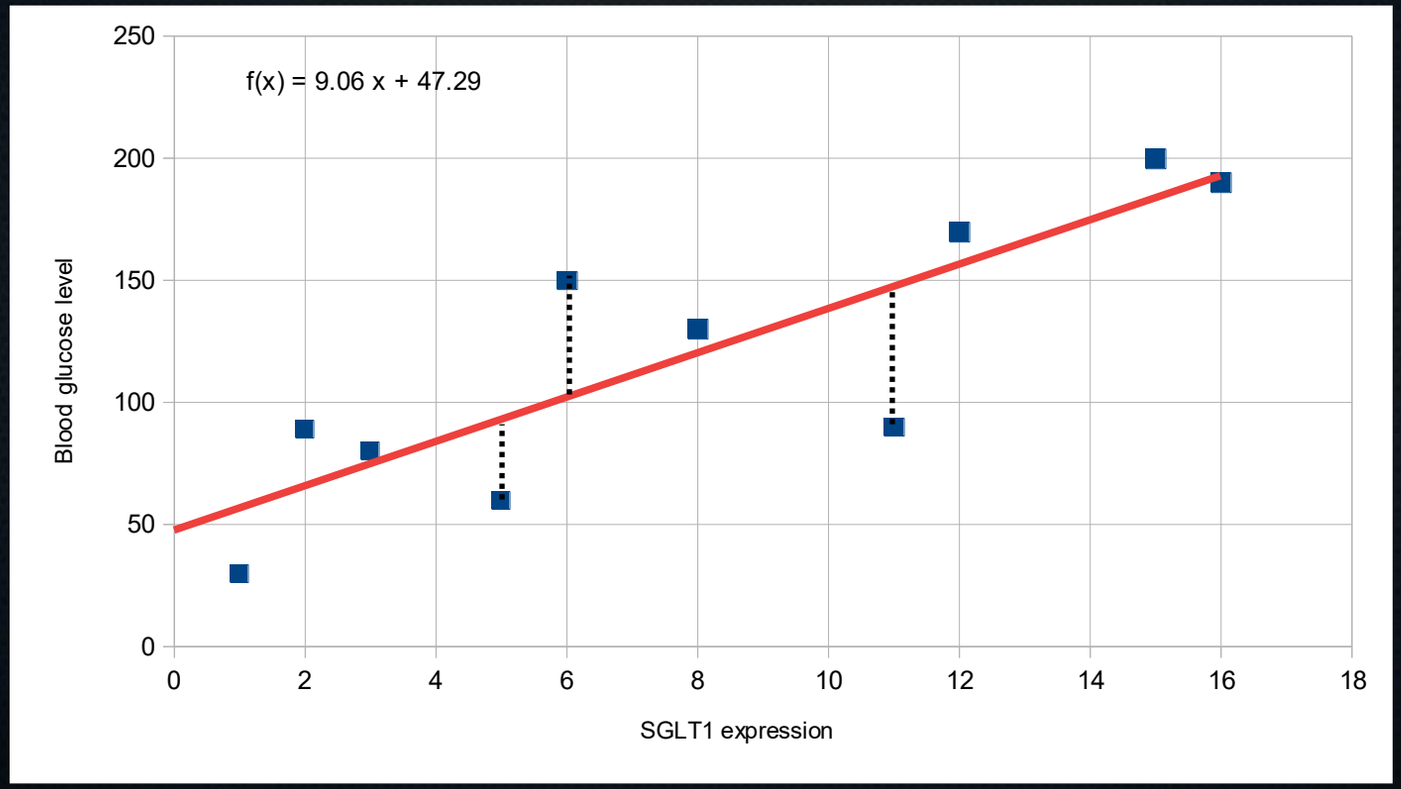
Cost function and gradient descent

- How to learn theta‘s from data?
- Two parts
  - ***How wrong are we for given parameters?***
  - How do we update our parameters, given how wrong we are?

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Cost function

- How wrong are we for given parameters? $h_\theta(x) = \theta_0 + \theta_1 x$

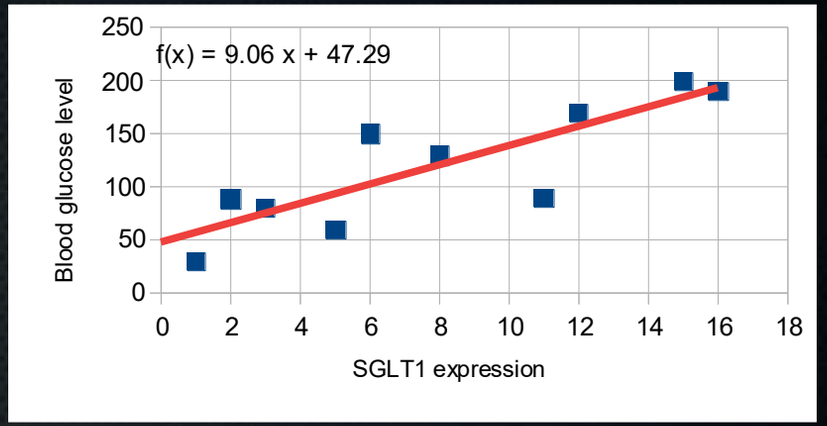


Cost function

- How wrong are we for given parameters?
- Cost function:

$$J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i}^{m}(h_\theta(x^{(i)})-y^{(i)})^2$$

$$h_\theta(x)=\theta_0+\theta_1 x$$



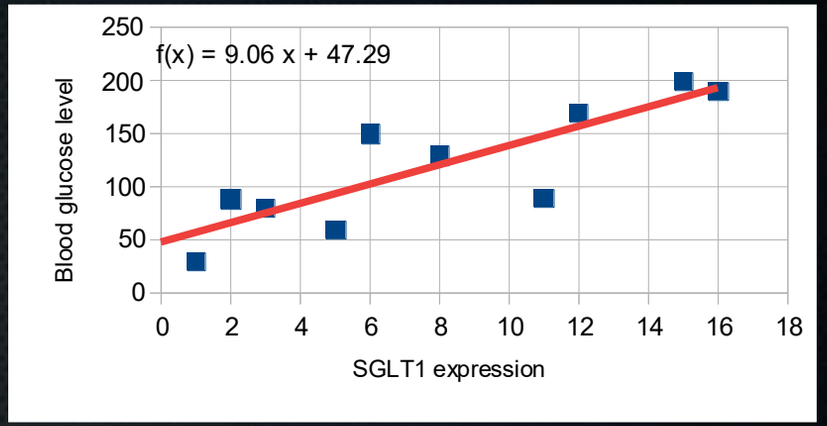
Cost function

- How wrong are we for given parameters?
- Cost function:

$$J(\theta_0,\theta_1)=\frac{1}{m}\sum_i^m (h_\theta(x^{(i)})-y^{(i)})^2$$

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_i^m (h_\theta(x^{(i)})-y^{(i)})^2$$

$$h_\theta(x)=\theta_0+\theta_1 x$$



Cost function

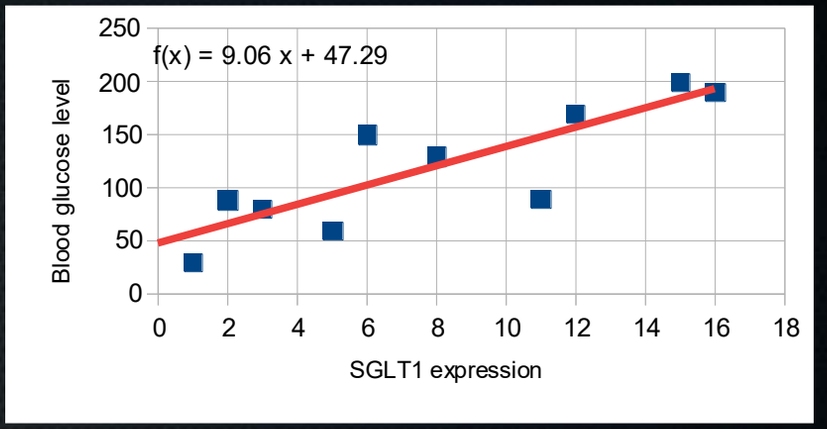
- How wrong are we for given parameters?
- Cost function:

$$J(\theta_0,\theta_1)=\frac{1}{m}\sum_i^m (h_\theta(x^{(i)})-y^{(i)})^2$$

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_i^m (h_\theta(x^{(i)})-y^{(i)})^2$$

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_i^m (\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$h_\theta(x)=\theta_0+\theta_1 x$$



Cost function

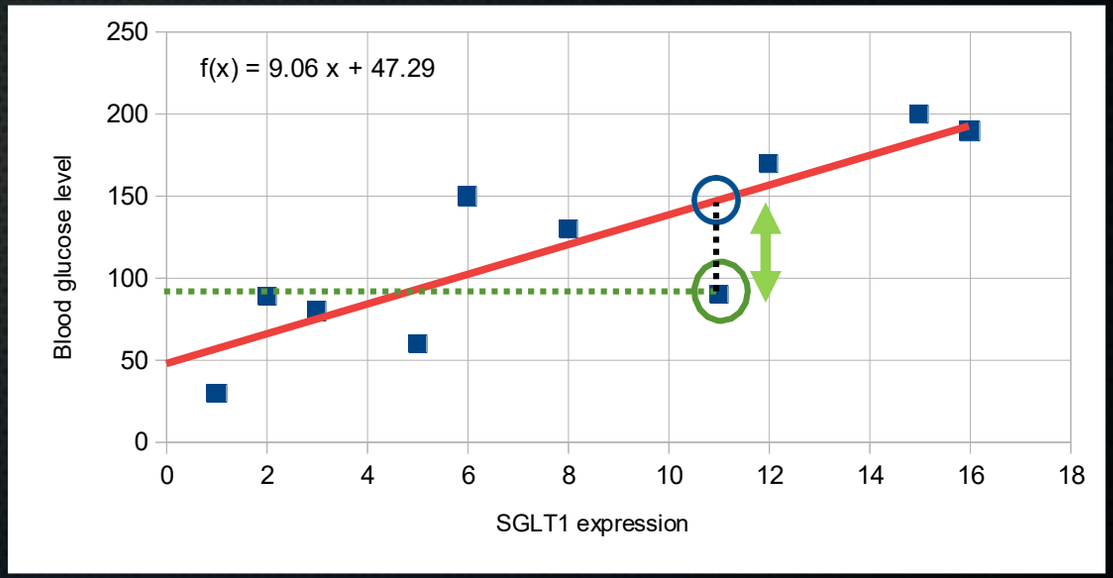
- How wrong are we for given parameters?
- Cost function:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 \cdot x^{(i)} - y^{(i)})^2$$



Cost function conclusion

- If we want to be as correct as possible with our prediction, want to minimise:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 \cdot x^{(i)} - y^{(i)})^2$$

Cost function conclusion

- If we want to be as correct as possible with our prediction, want to minimise:

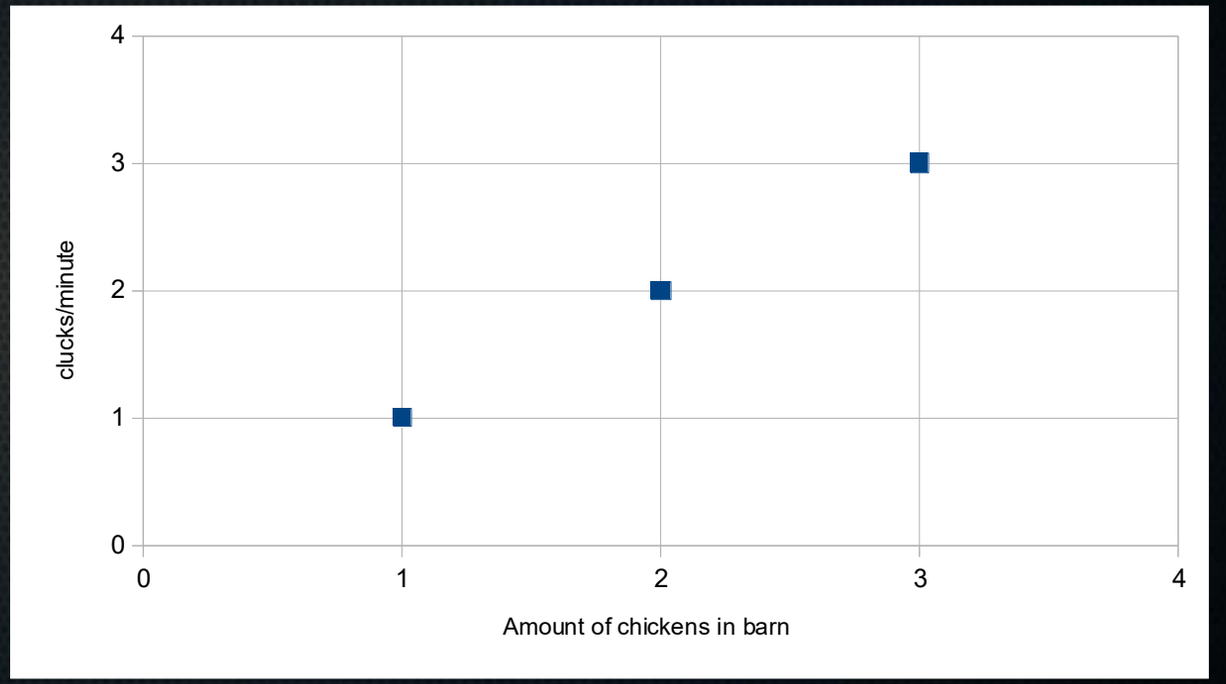
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 \cdot x^{(i)} - y^{(i)})^2$$

- Simplified example: Theta_0 = 0:

$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\theta_1 \cdot x^{(i)} - y^{(i)})^2$$

Cost function conclusion illustration

- Let‘s say theta0 = 0 (so the intercept is 0). What is J(theta1) for different values of theta1?

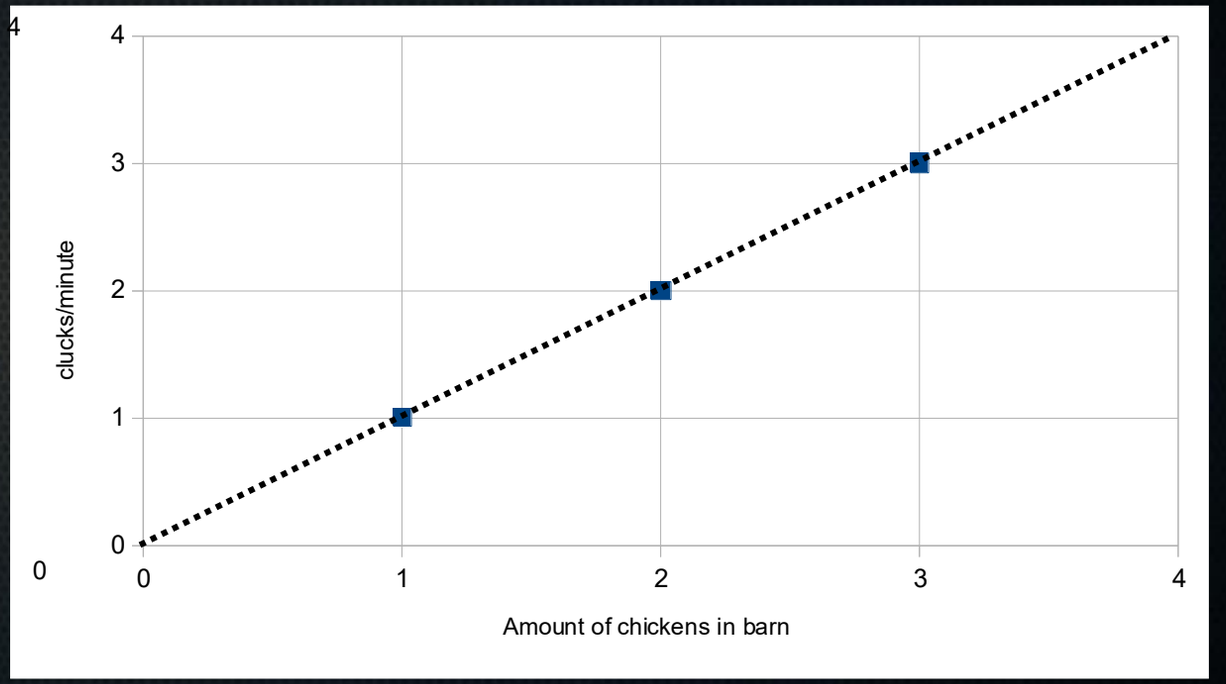


Cost function illustration

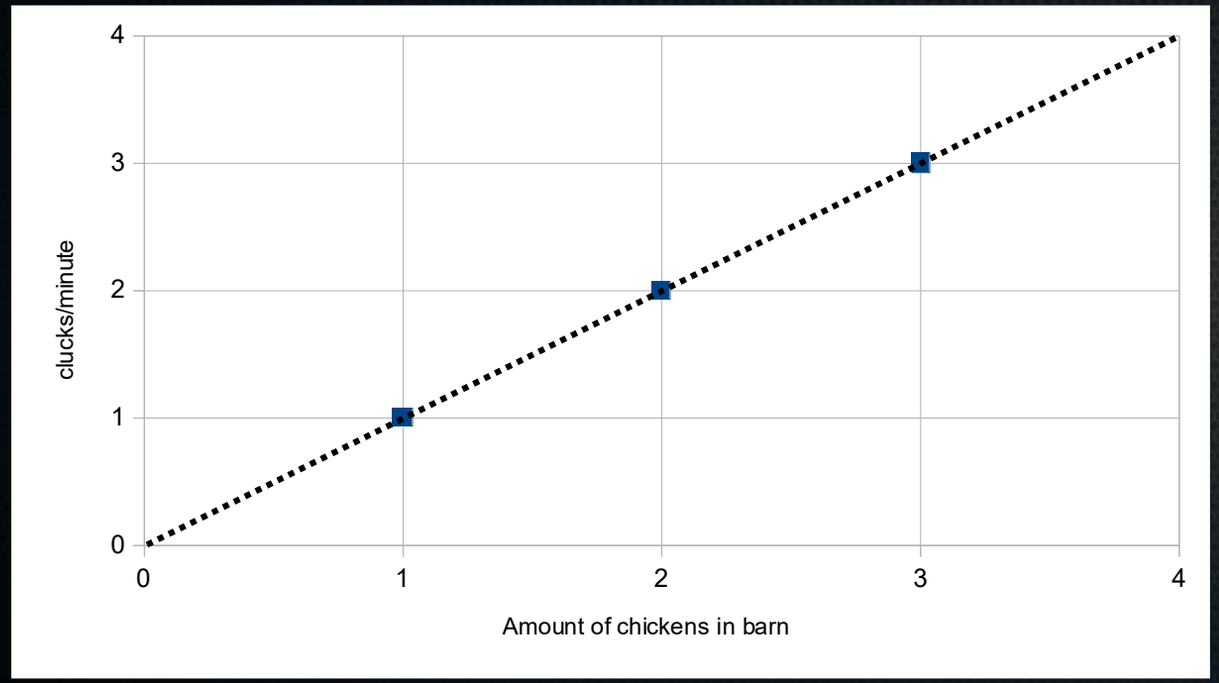
- Let‘s say theta0 = 0 (so the intercept is 0). What is J(theta1) for different values of theta1?
- Theta1 = 1

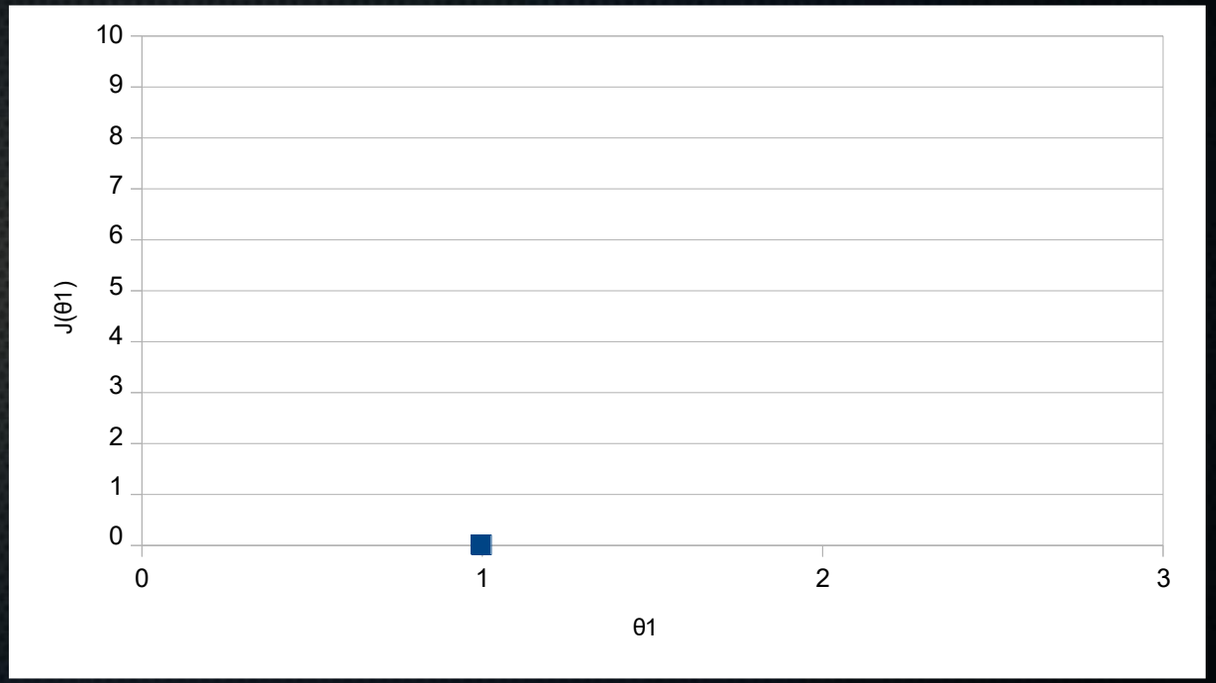
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{3} (\theta_1 \cdot x^{(i)} - y^{(i)})^2$$

$$J(\theta_1) = 0$$



Cost function illustration

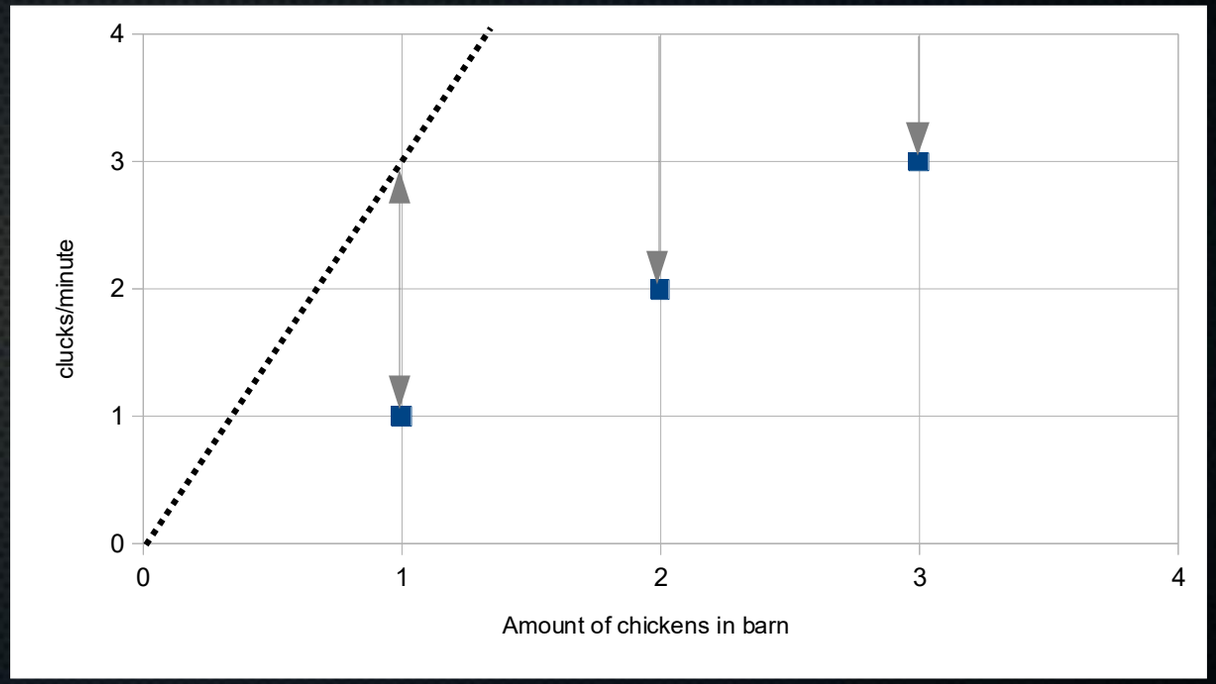




Cost function illustration

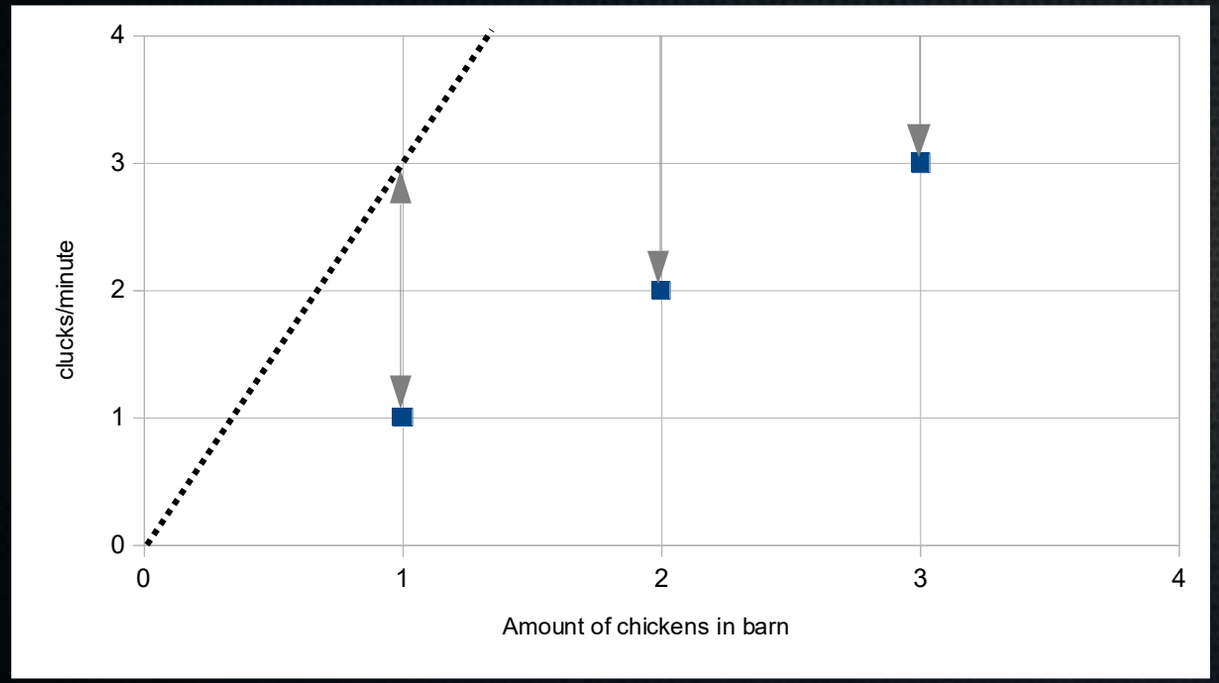
- Let‘s say theta0 = 0 (so the intercept is 0). What is J(theta1) for different values of theta1?
- Theta1 = 3

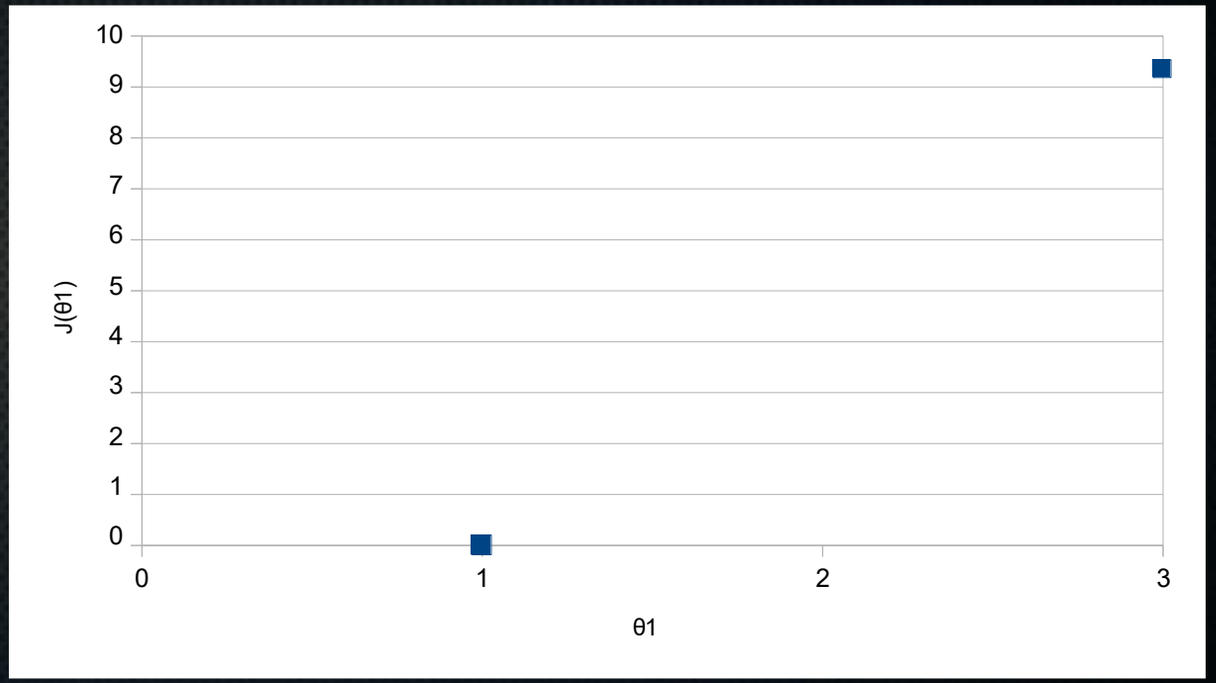
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{3} (\theta_1 \cdot x^{(i)} - y^{(i)})^2$$



$$J(\theta_1) = \frac{1}{2 \cdot 3} \cdot ((3-1)^2 + (6-2)^2 + (9-3)^2) = 56/6 \approx 9.3$$

Cost function illustration

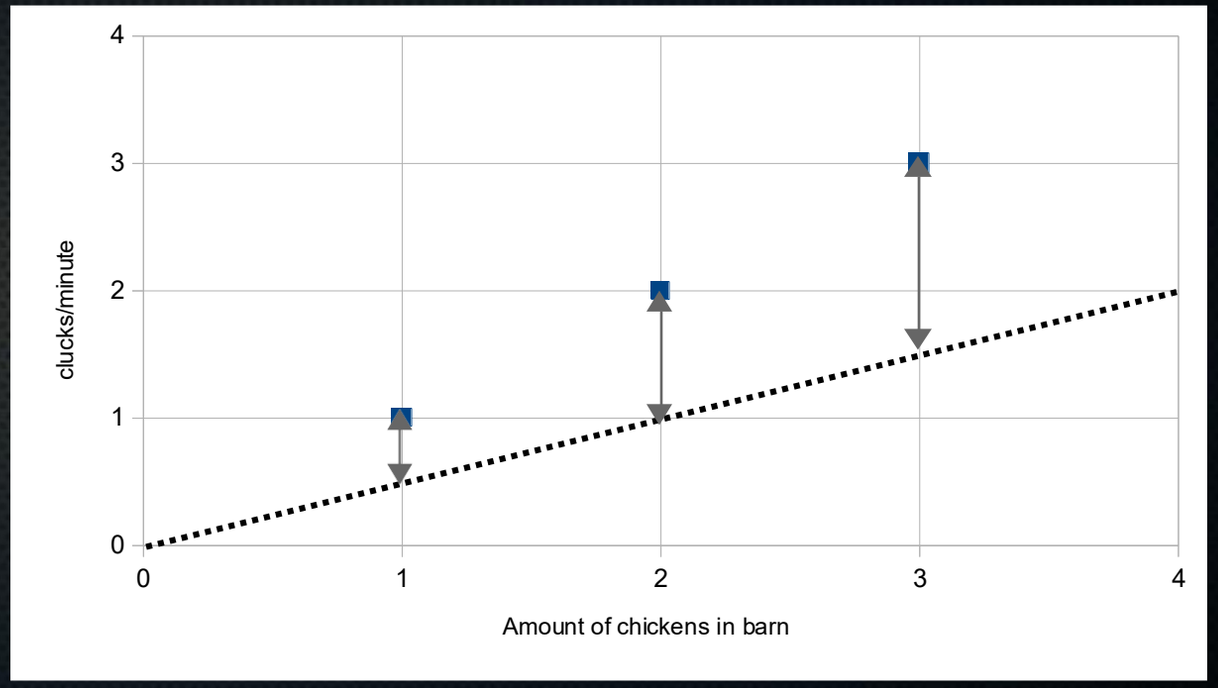




Cost function illustration

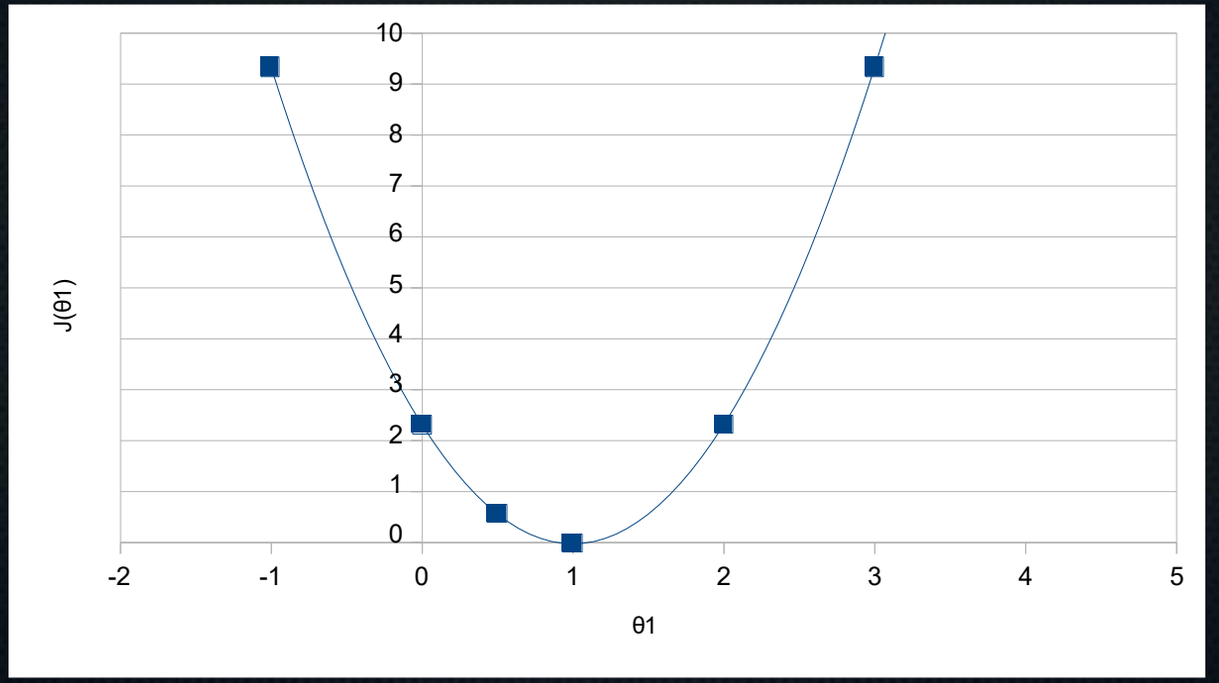
- Let‘s say theta0 = 0 (so the intercept is 0). What is J(theta1) for different values of theta1?
- Theta1 = 0.5

$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{3} (\theta_1 \cdot x^{(i)} - y^{(i)})^2$$



$$J(\theta_1) = \frac{1}{2 \cdot 3} \cdot ((0.5-1)^2 + (1-2)^2 + (1.5-3)^2) = 3.5 / 6 \approx 0.6$$

Cost function illustration

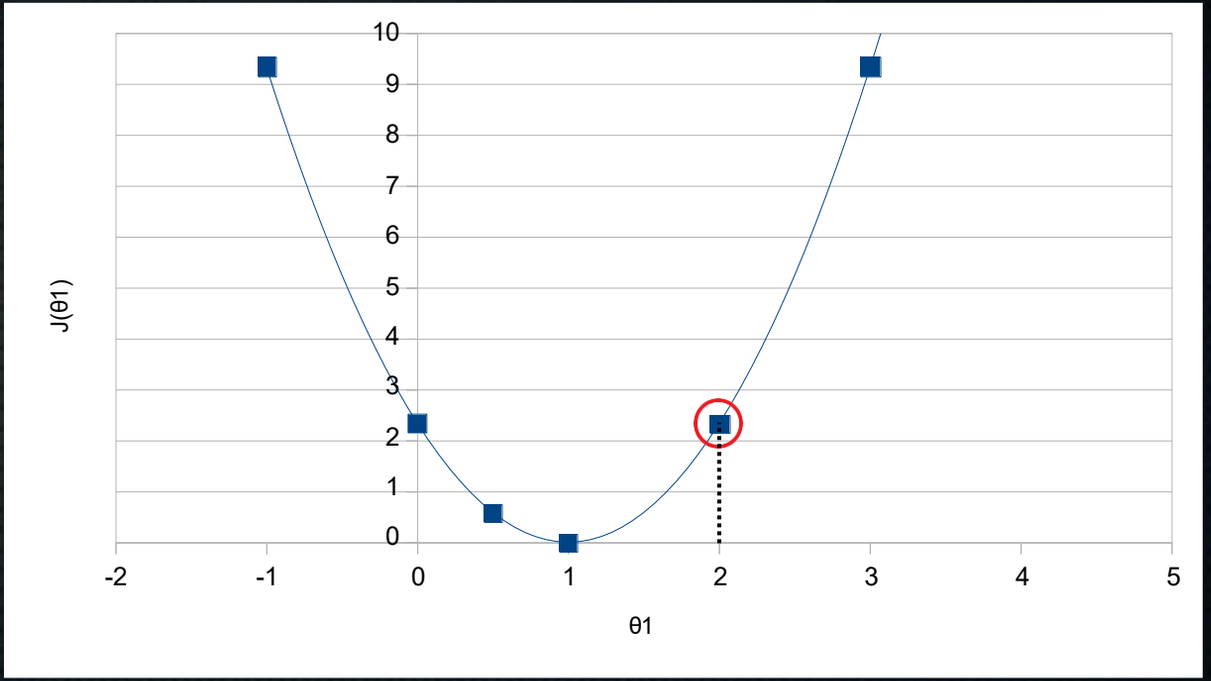


Cost function and gradient descent

- How to learn theta‘s from data?
- Two parts
  - How wrong are we for given parameters? 
  - ***How do we update our parameters, given how wrong we are?***

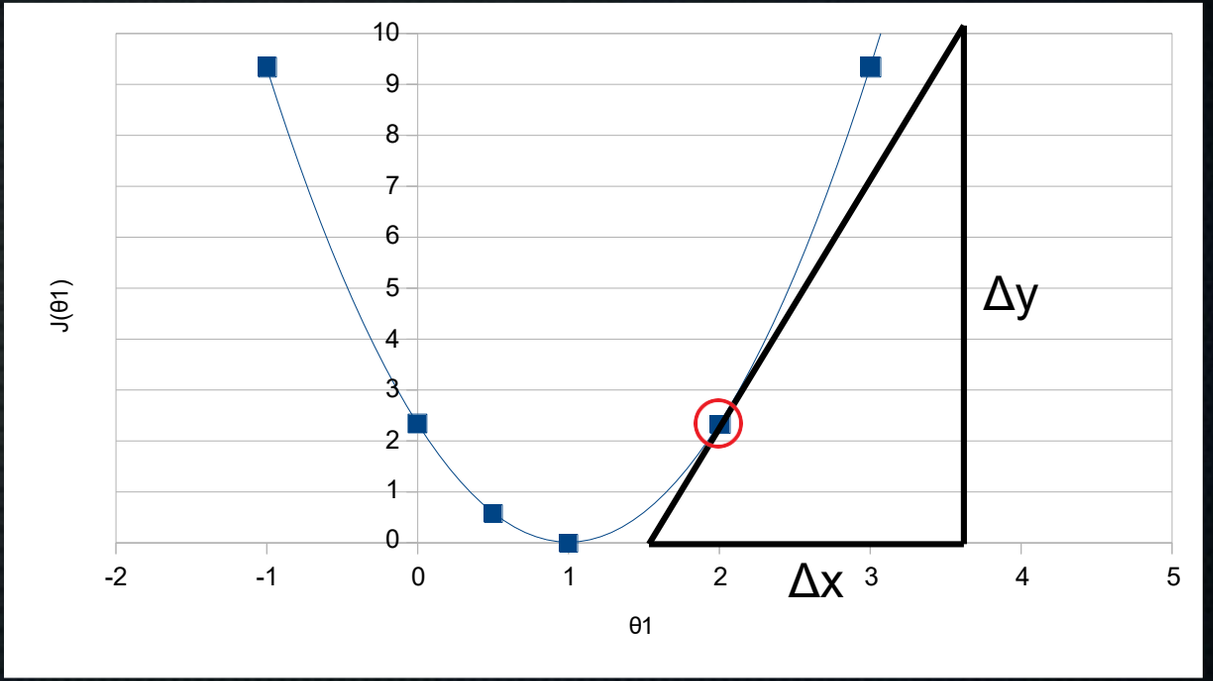
Gradient descent

- Want to minimise
- Where to go?



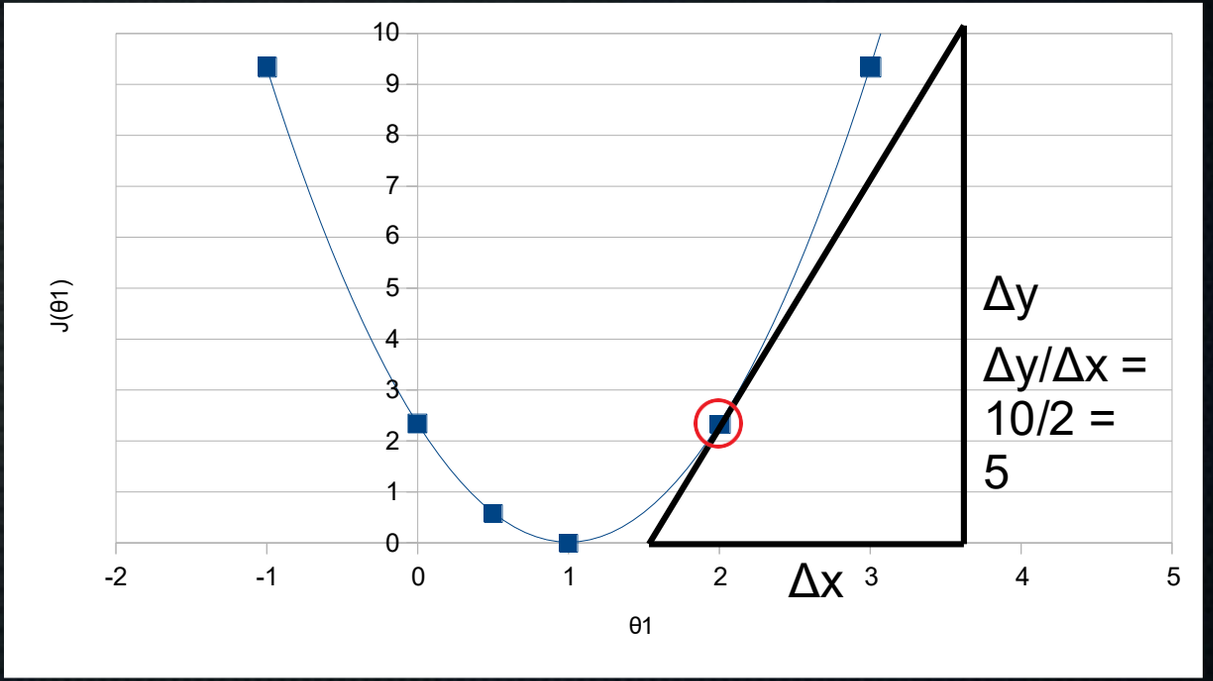
Gradient descent

- Want to minimise
- Where to go?



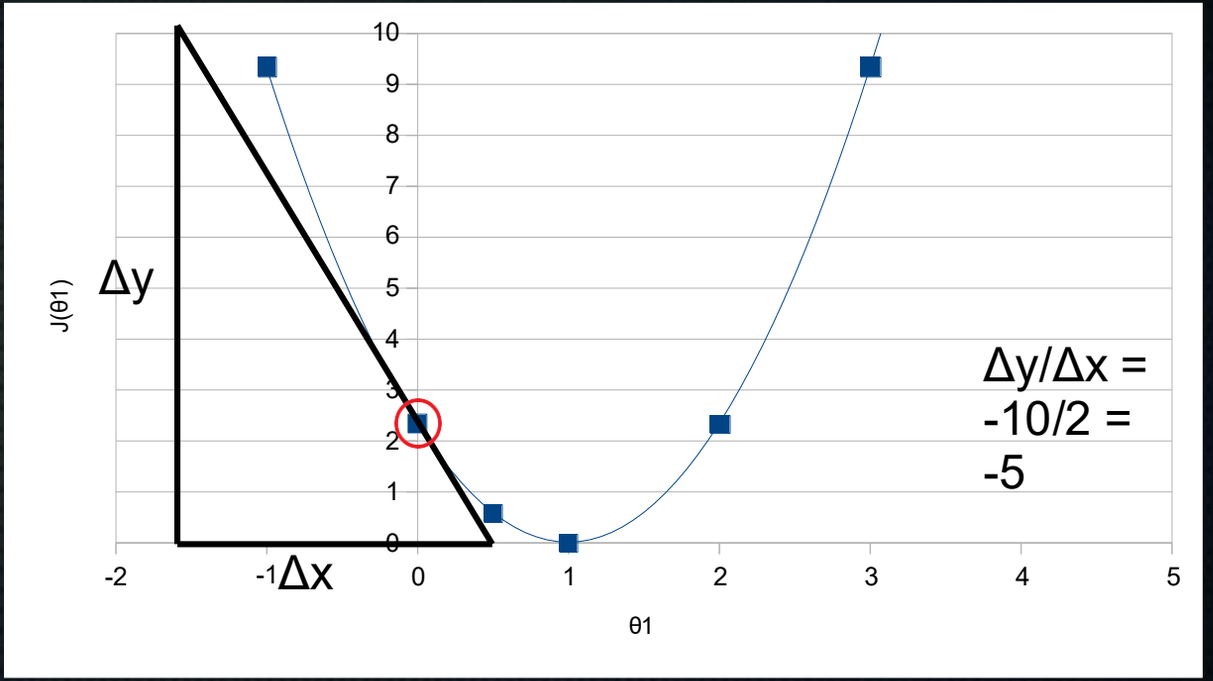
Gradient descent

- Want to minimise
- Where to go?



Gradient descent

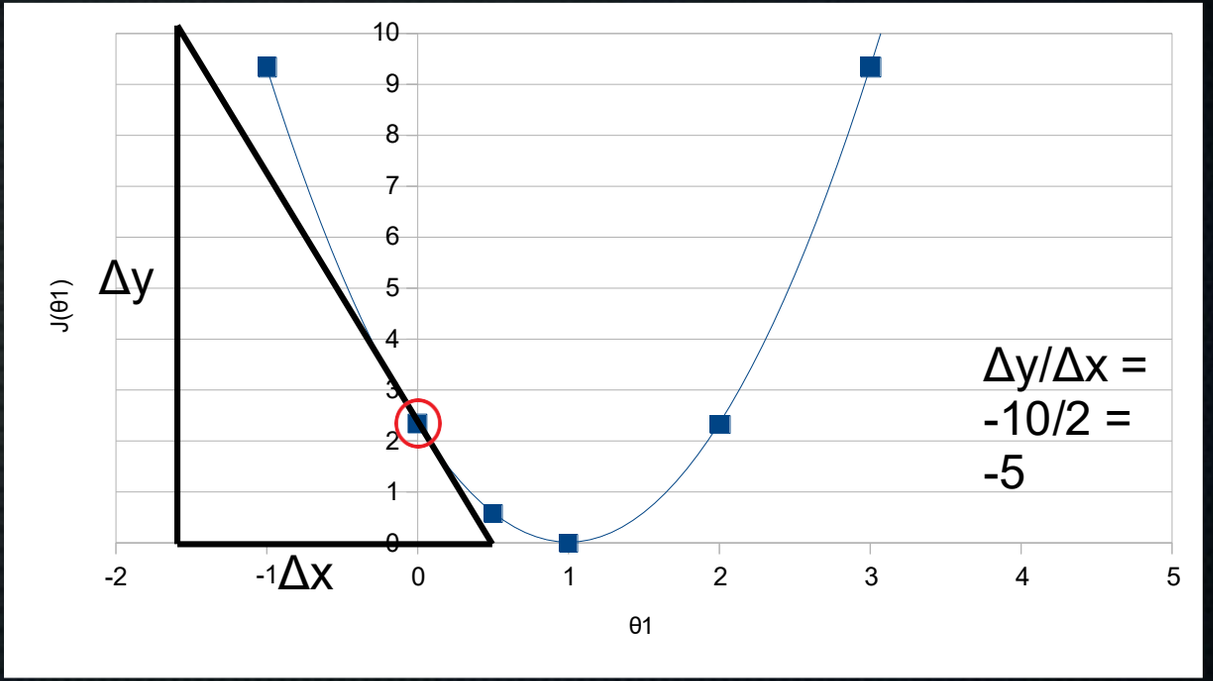
- Want to minimise
- Where to go?



Gradient descent

- Want to minimise
- Where to go?
- Change current theta1 as follows:

$$\theta_1 = \theta_1 - \alpha \cdot \frac{d}{d\theta_1} J(\theta_1)$$



Gradient descent

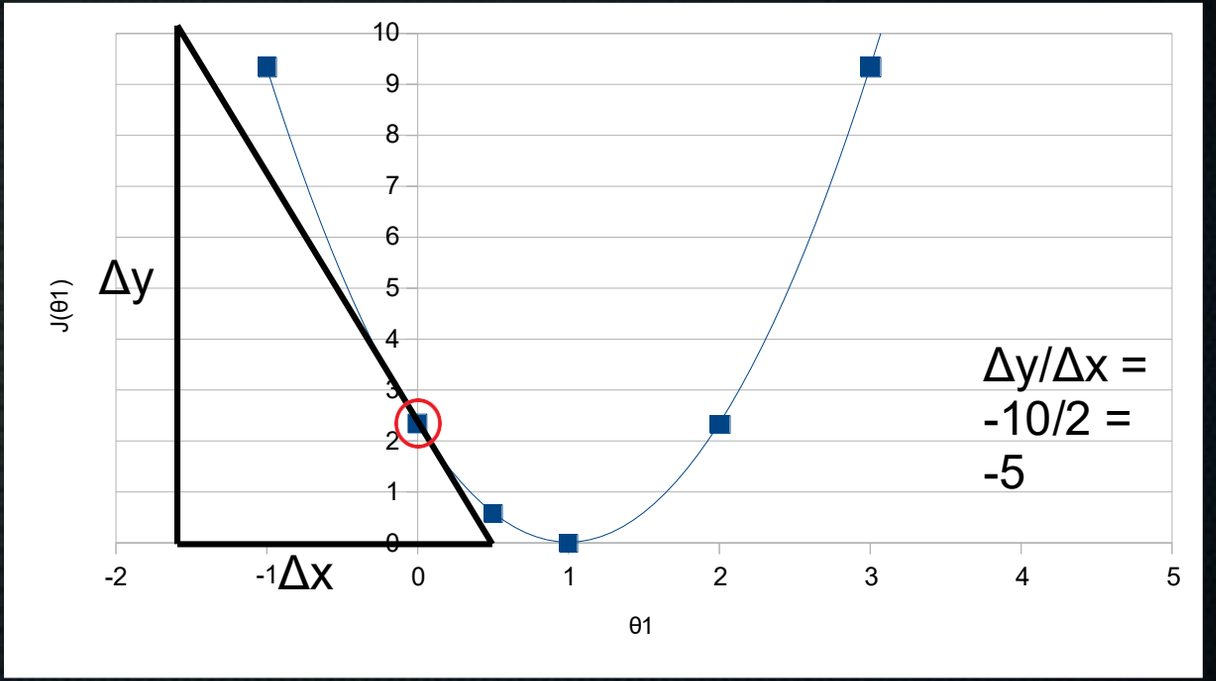
- Want to minimise
- Where to go?
- Change current theta1 as follows:

$$\theta_1 = \theta_1 - \alpha \cdot \frac{d}{d\theta_1} J(\theta_1)$$

$$\alpha = 0.1$$

$$\theta_1 = 0 - 0.1 \cdot -5$$

$$\theta_1 = 0.5$$



Gradient descent

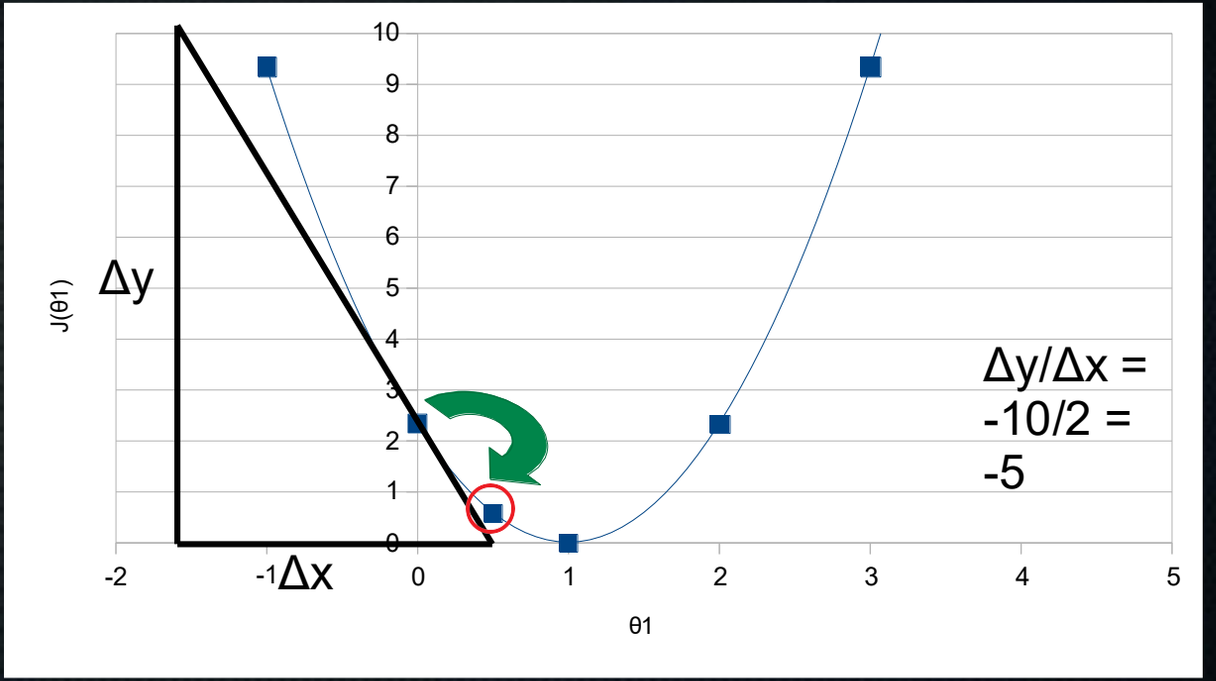
- Want to minimise
- Where to go?
- Change current theta1 as follows:

$$\theta_1 = \theta_1 - \alpha \cdot \frac{d}{d\theta_1} J(\theta_1)$$

$$\alpha = 0.1$$

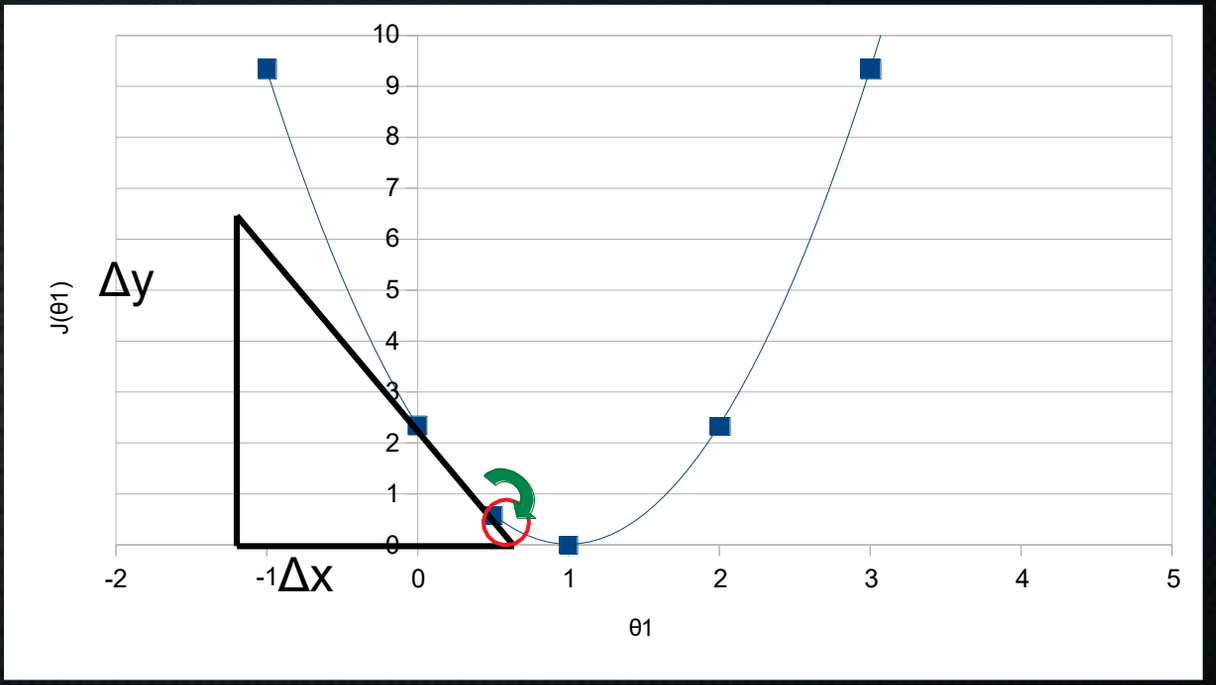
$$\theta_1 = 0 - 0.1 \cdot -5$$

$$\theta_1 = 0.5$$



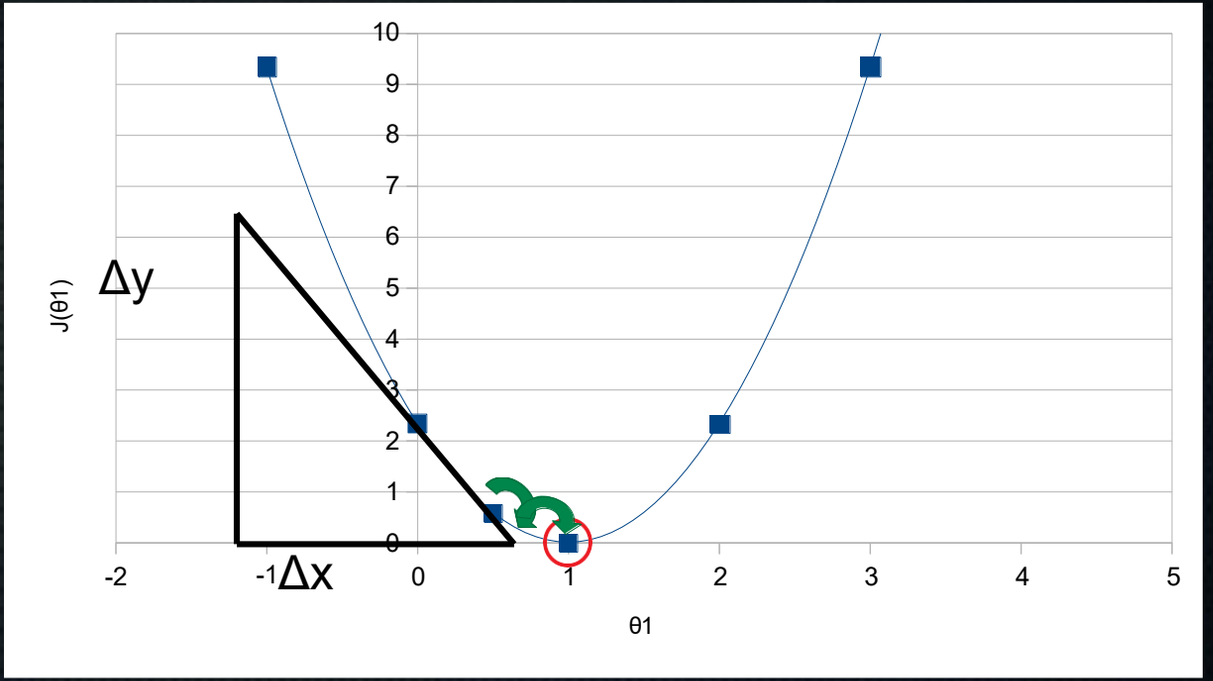
Gradient descent

- Want to minimise
- Where to go?
- Change current theta1
- Note: gradient becomes smaller closer to optimum, so can use fixed value for α



Gradient descent

- Want to minimise
- Where to go?
- Change current theta1
- Note: gradient becomes smaller closer to optimum, so can use fixed value for α



Gradient descent

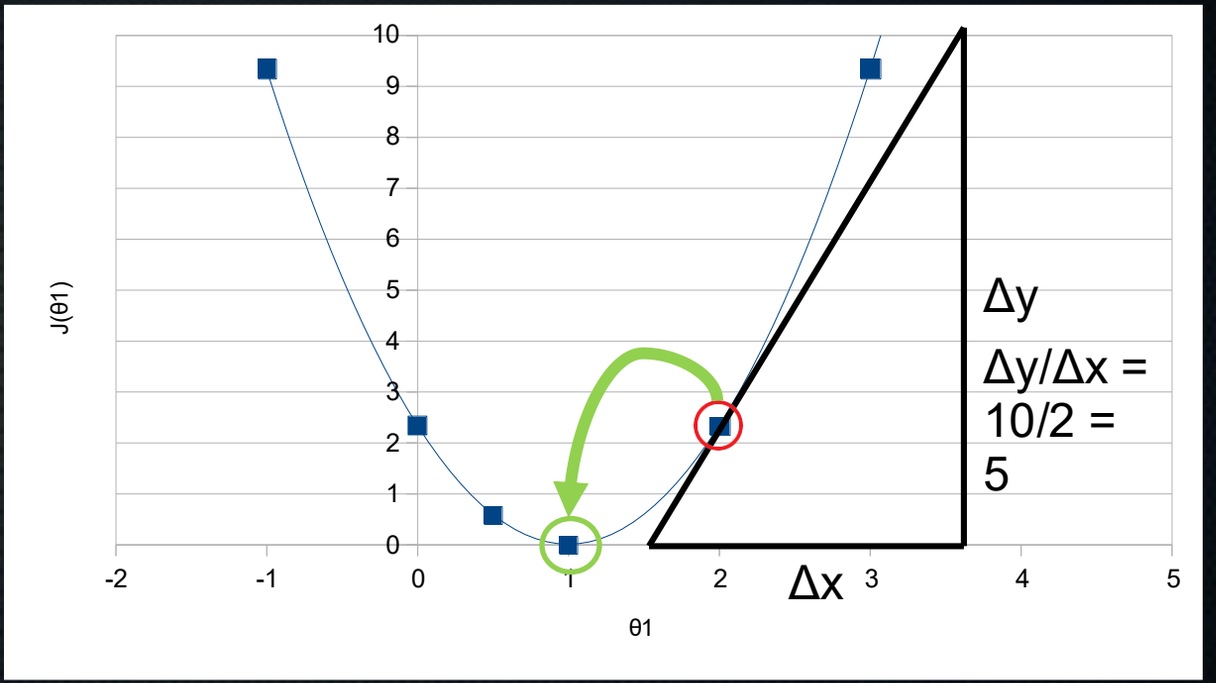
- Works also from other direction.

$$\theta_1 = \theta_1 - a \cdot \frac{d}{d\theta_1} J(\theta_1)$$

$$a = 0.2$$

$$\theta_1 = 2 - 0.2 \cdot 5$$

$$\theta_1 = 1$$

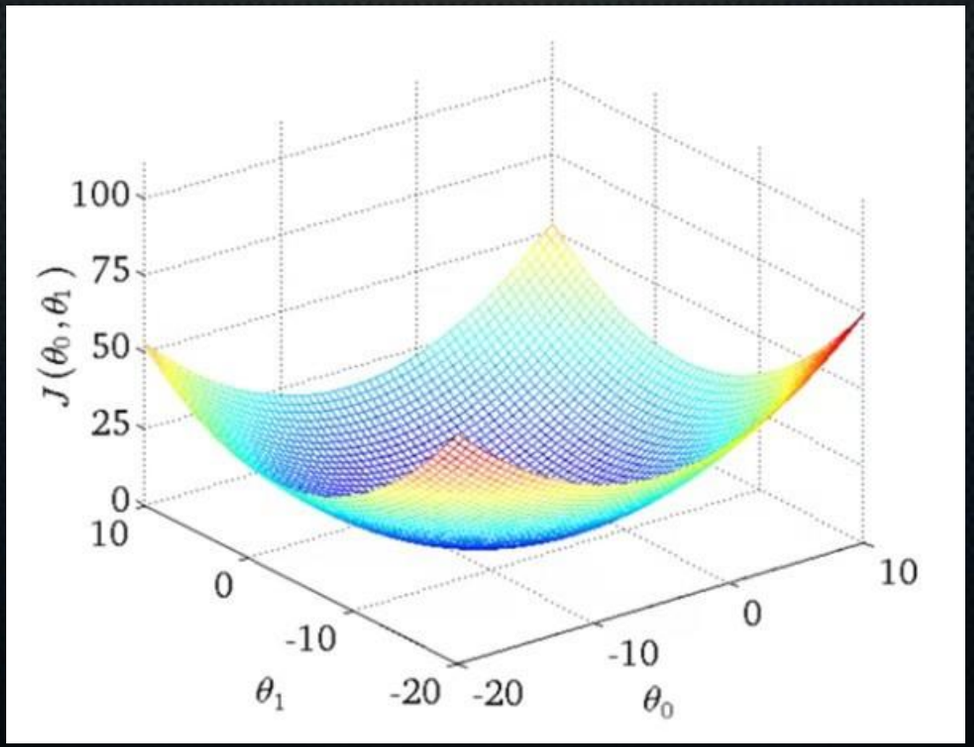


Gradient descent

- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!

Gradient descent

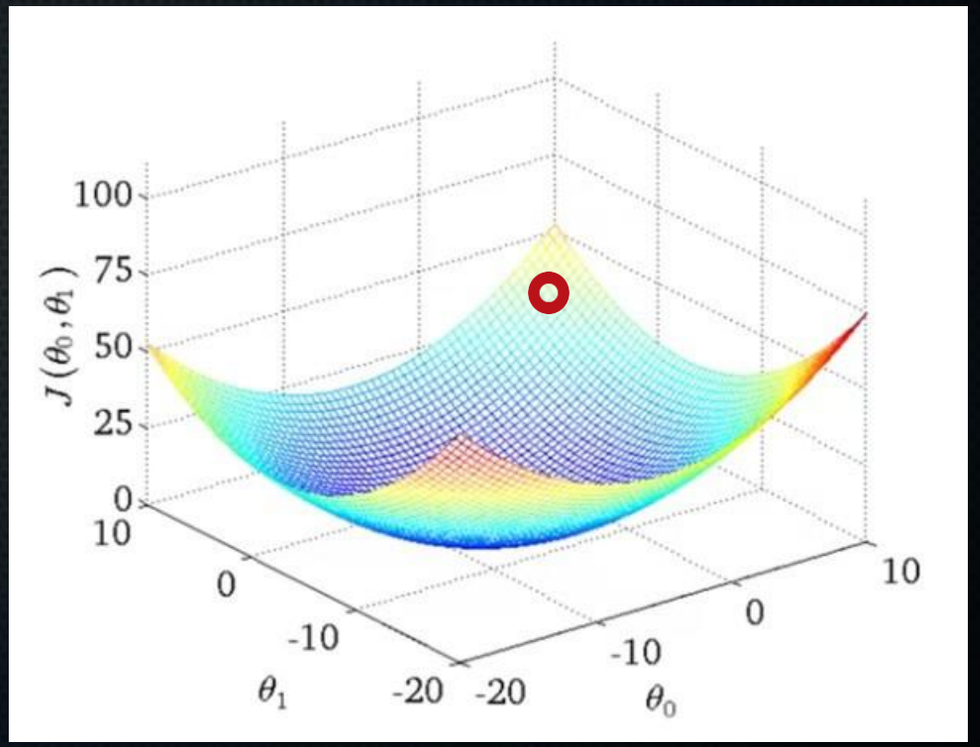
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

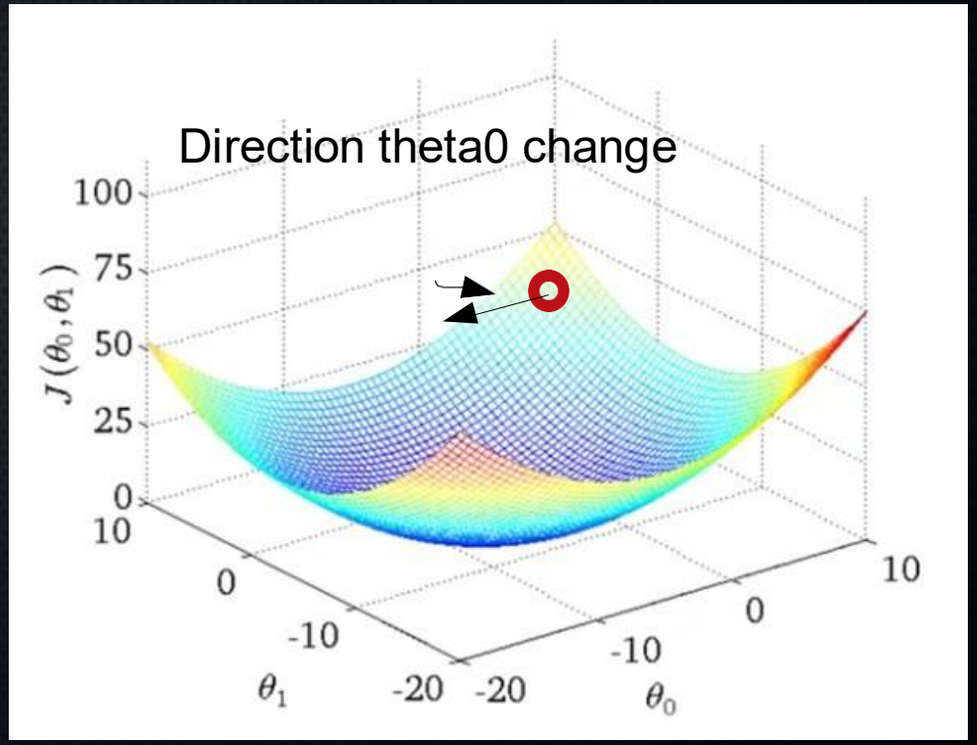
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional. Want to fit intercept *and* slope.
- Use partial derivatives instead:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

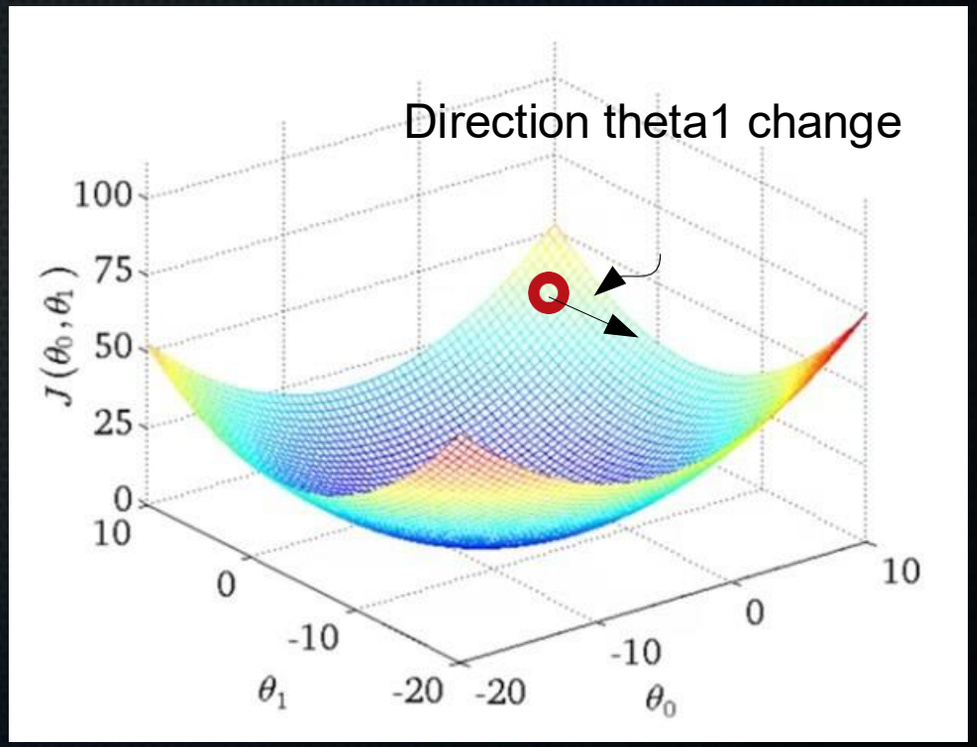
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional. Want to fit intercept *and* slope.
- Use partial derivatives instead:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

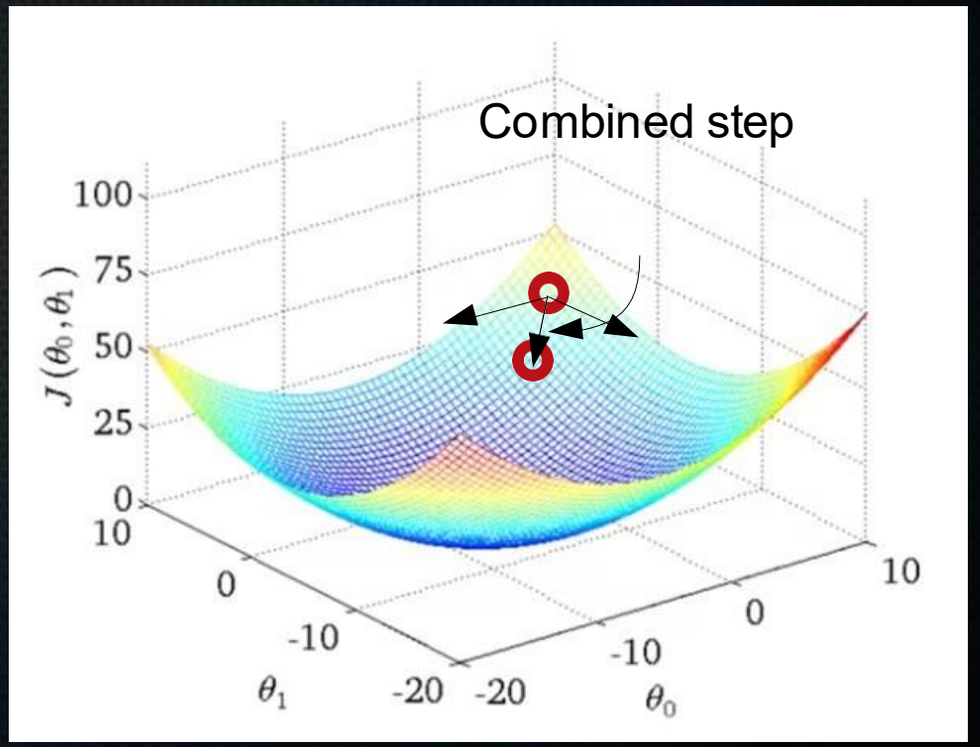
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional. Want to fit intercept *and* slope.
- Use partial derivatives instead:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

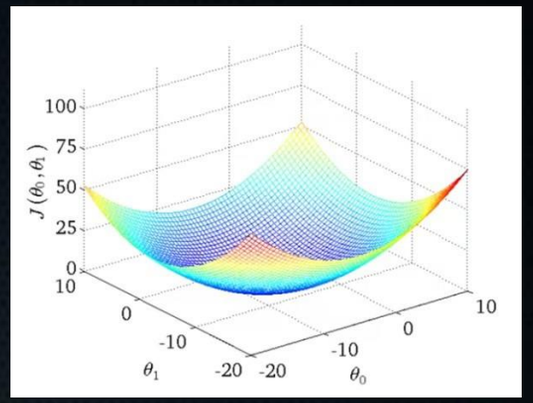
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional. Want to fit intercept *and* slope.
- Use partial derivatives instead:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

- Main idea:
- Act like your function depends on only 1 parameter and treat the rest as constants (just numbers), then take the derivative. Do that for every parameter!

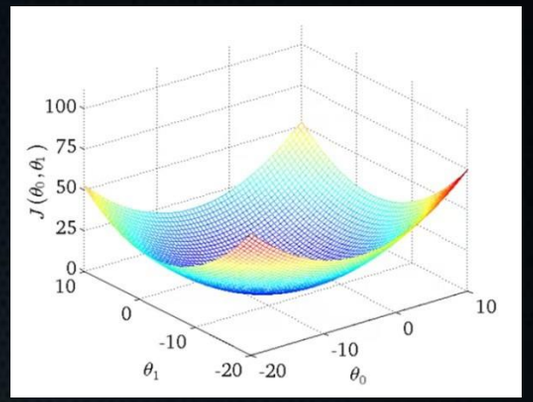


Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$



Source: Andrew Ng, Coursera

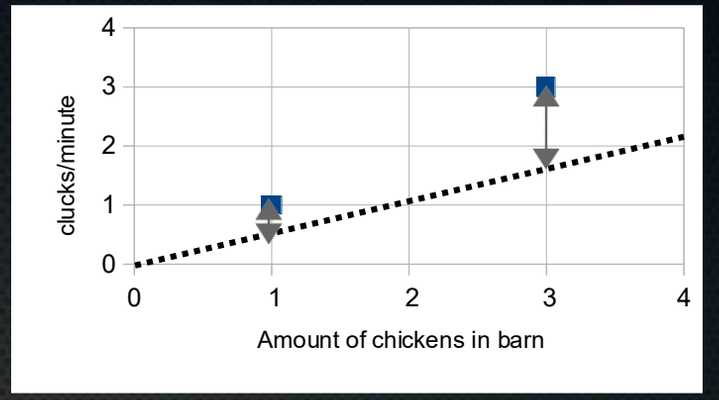
Gradient descent: partial derivatives

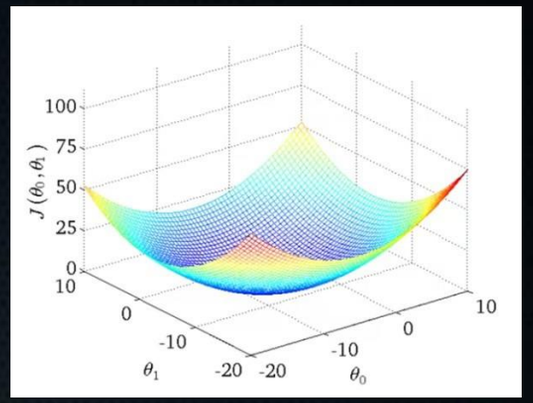
- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})^2+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})^2)$$





Source: Andrew Ng, Coursera

$$f(g(x)) \qquad \frac{dy}{dx}=f'(g(x))\times g'(x)$$

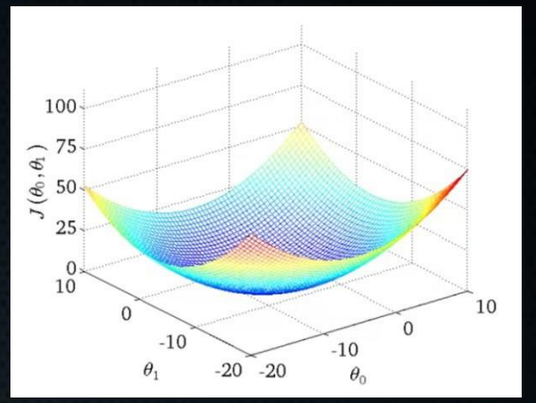
Gradient descent: partial derivatives

- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\underbrace{\theta_0+\theta_1\cdot x^{(1)}-y^{(1)}}_{x^2})^2+(\underbrace{\theta_0+\theta_1\cdot x^{(2)}-y^{(2)}}_{z^2})^2)$$



Source: Andrew Ng, Coursera

$$f(g(x)) \qquad \frac{dy}{dx}=f'(g(x))\times g'(x)$$

Gradient descent: partial derivatives

- Partial derivatives:

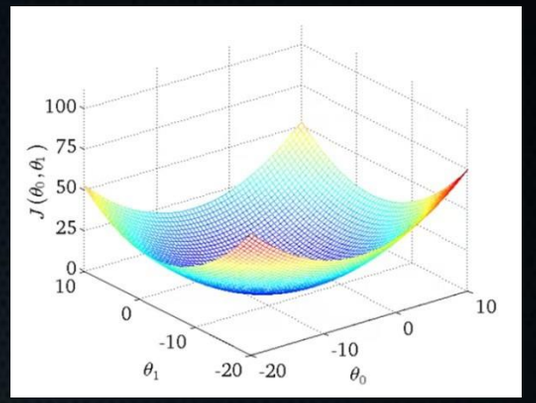
$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\underbrace{\theta_0+\theta_1\cdot x^{(1)}-y^{(1)}}_{x})^2+(\underbrace{\theta_0+\theta_1\cdot x^{(2)}-y^{(2)}}_{z})^2)$$

x^2 $\quad$ z^2

$2x$ $\quad$ $2z$



Source: Andrew Ng, Coursera

$$f(g(x)) \qquad \frac{dy}{dx}=f'(g(x))\times g'(x)$$

Gradient descent: partial derivatives

- Partial derivatives:

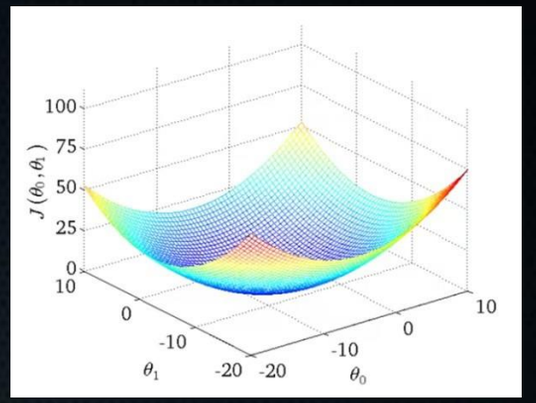
$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\underline{\theta_0+\theta_1\cdot x^{(1)}-y^{(1)}})^2+(\underline{\theta_0+\theta_1\cdot x^{(2)}-y^{(2)}})^2)$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}(\underline{2(\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})}*1$$

$$\underline{+2(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})}*1)$$



Source: Andrew Ng, Coursera

$$f(\underline{g(x)})$$

$$\frac{dy}{dx}=\underline{f'(g(x))}\times g'(x)$$

Gradient descent: partial derivatives

- Partial derivatives:

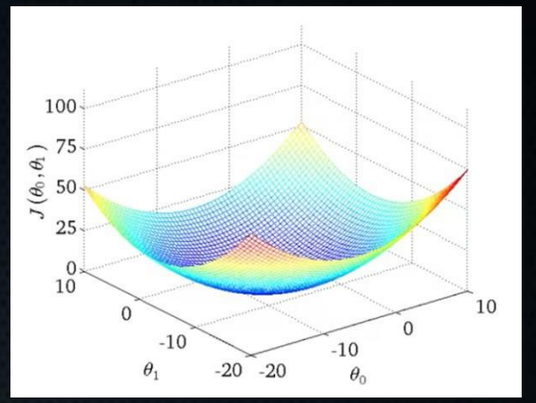
$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\underline{\theta_0+\theta_1\cdot x^{(1)}-y^{(1)}})^2+(\underline{\theta_0+\theta_1\cdot x^{(2)}-y^{(2)}})^2)$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}(\underline{2(\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})}*\underline{1}$$

$$+\underline{2(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})}*\underline{1})$$



Source: Andrew Ng, Coursera

$$f(\underline{g(x)})$$

$$\frac{dy}{dx}=\underline{f'(g(x))}\times\underline{g'(x)}$$

Gradient descent: partial derivatives

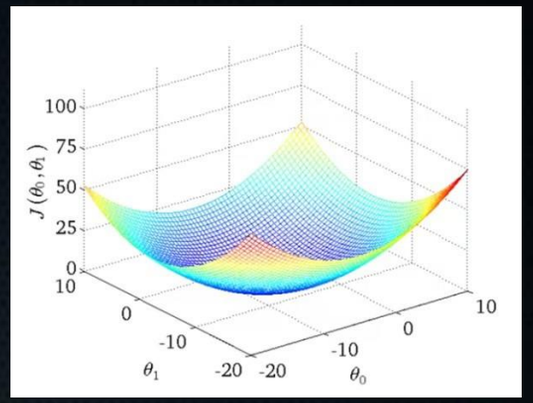
- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})^2+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})^2)$$

$$g'(x)=\frac{\partial}{\partial\theta_0}(1\cdot\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})=1$$



Source: Andrew Ng, Coursera

$$f(g(x)) \qquad \frac{dy}{dx}=f'(g(x))\times g'(x)$$

Gradient descent: partial derivatives

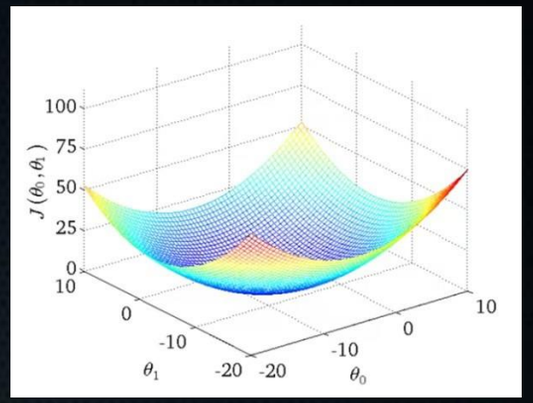
- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})^2+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})^2)$$

$$f'(g(x))=\frac{\partial}{\partial\theta_0}(g(x))^2=2\cdot g(x)$$



Source: Andrew Ng, Coursera

$$f(g(x)) \qquad \frac{dy}{dx}=f'(g(x))\times g'(x)$$

Gradient descent: partial derivatives

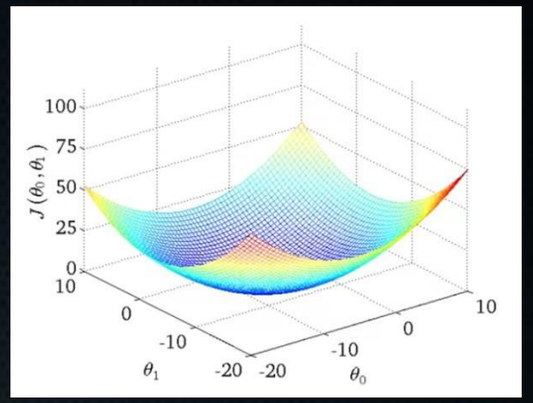
- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}(2(\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})*1$$

$$+2(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})*1)$$



Source: Andrew Ng, Coursera

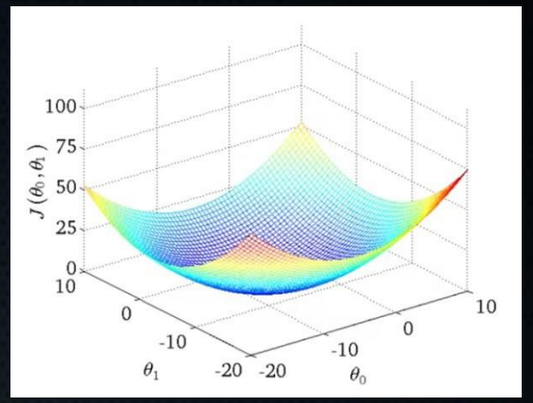
Gradient descent: partial derivatives

- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}(\underbrace{2(\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})*1}+\underbrace{2(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})*1})$$



Source: Andrew Ng, Coursera

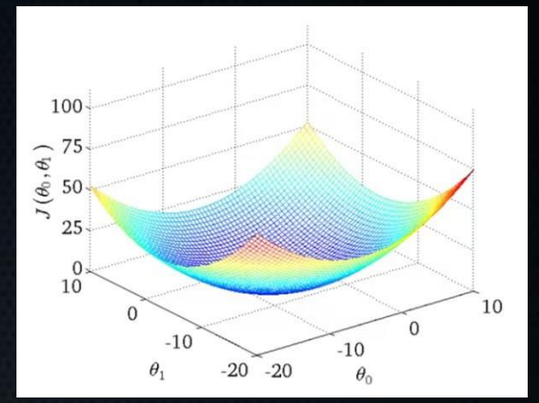
Gradient descent: partial derivatives

- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}(2((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})*1+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})*1))$$



Source: Andrew Ng, Coursera

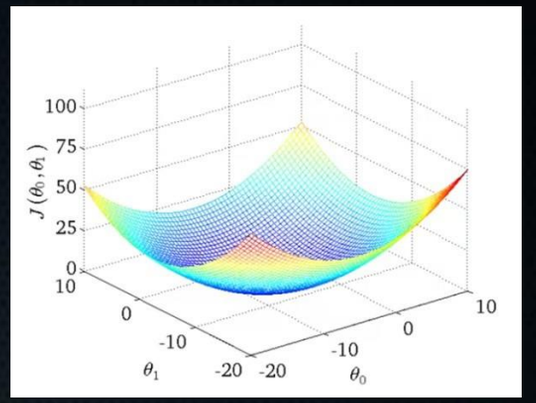
Gradient descent: partial derivatives

- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}(2((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})*1+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})*1))$$



Source: Andrew Ng, Coursera

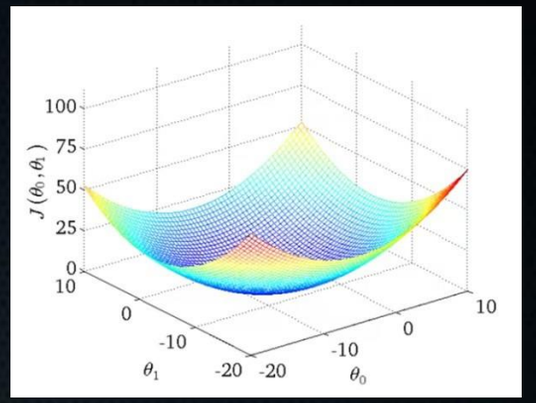
Gradient descent: partial derivatives

- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2}((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)}))$$



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

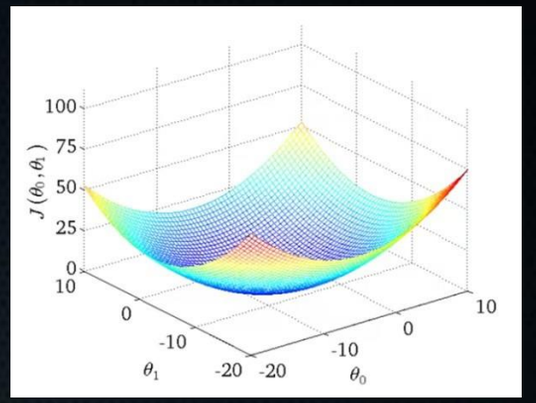
- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2}((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)}))$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})$$



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

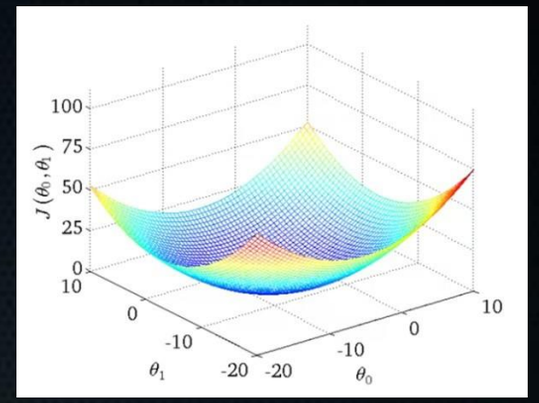
- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

$$m=2$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{2}((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)}))$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})=\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})$$



Source: Andrew Ng, Coursera

$$h_\theta(x)=\theta_0+\theta_1 x$$

Gradient descent: partial derivatives

- Partial derivatives:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1\cdot x^{(i)}-y^{(i)})^2$$

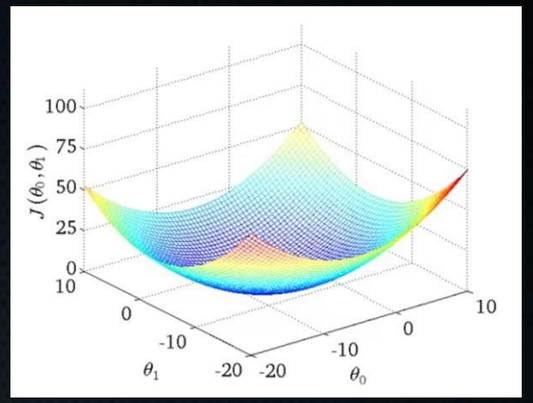
$$m=2$$

$$J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}((\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})^2+(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})^2)$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\frac{1}{2\cdot 2}(2(\theta_0+\theta_1\cdot x^{(1)}-y^{(1)})*x^{(1)}$$

$$+2(\theta_0+\theta_1\cdot x^{(2)}-y^{(2)})*x^{(2)})$$

$$(2(\underbrace{\theta_0}+\theta_1\underbrace{\cdot x^{(1)}}-\underbrace{y^{(1)}})$$



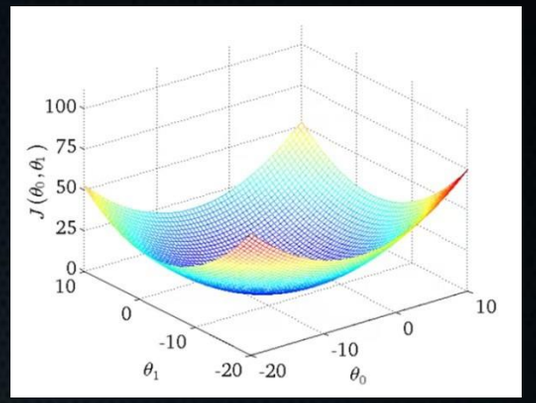
Source: Andrew Ng, Coursera

$$f(g(x)) \qquad \frac{dy}{dx}=f'(g(x))\times g'(x)$$

Gradient descent: partial derivatives

- Partial derivative theta1:

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)})$$



Source: Andrew Ng, Coursera

Cost function and gradient descent

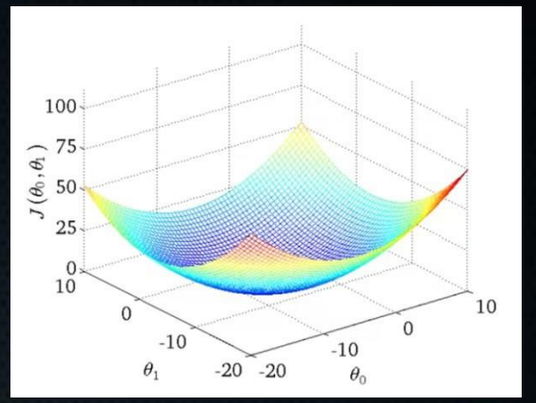
- Cost function:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x)-y^{(i)})^2$$

- Partial derivatives for gradient descent:

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}((h_\theta(x^{(i)})-y^{(i)})\cdot x^{(i)})$$



Source: Andrew Ng, Coursera

$$h_\theta(x)=\theta_0+\theta_1 x$$

Cost function and gradient descent

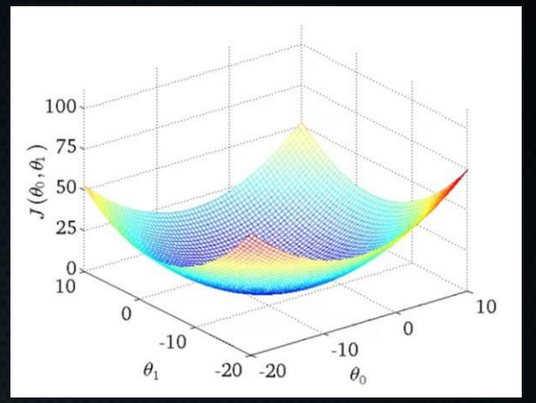
- Cost function:

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x)-y^{(i)})^2$$

- Gradient descent update:

$$\theta_0=\theta_0-a\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\theta_0-\frac{a}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})$$

$$\theta_1=\theta_1-a\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\theta_1-\frac{a}{m}\sum_{i=1}^{m}((h_\theta(x^{(i)})-y^{(i)})\cdot x^{(i)})$$

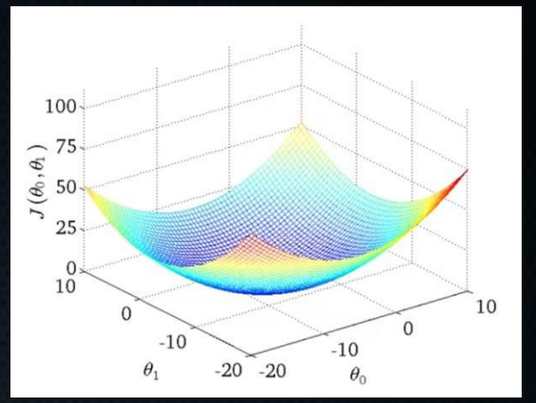


Source: Andrew Ng, Coursera

$$h_\theta(x)=\theta_0+\theta_1 x$$

What about α?

- Learning rate, so-called hyperparameter.

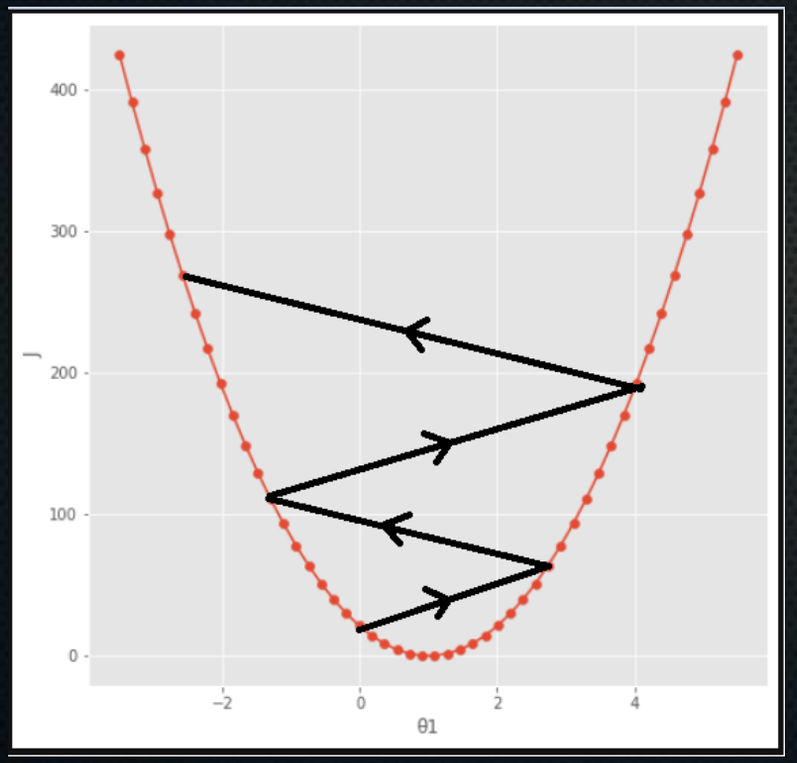


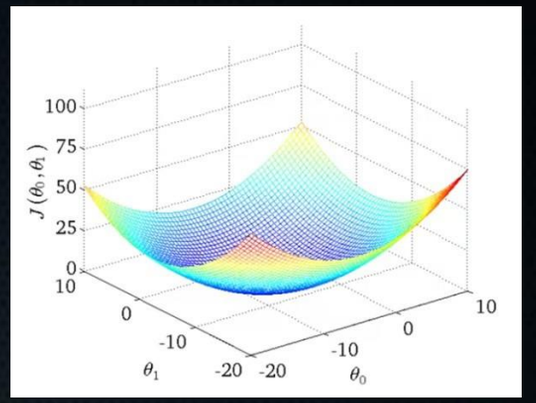
Source: Andrew Ng, Coursera

$$\theta_0 = \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \theta_0 - \frac{\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

What about α?

- Learning rate, so-called hyperparameter.
- Too high:





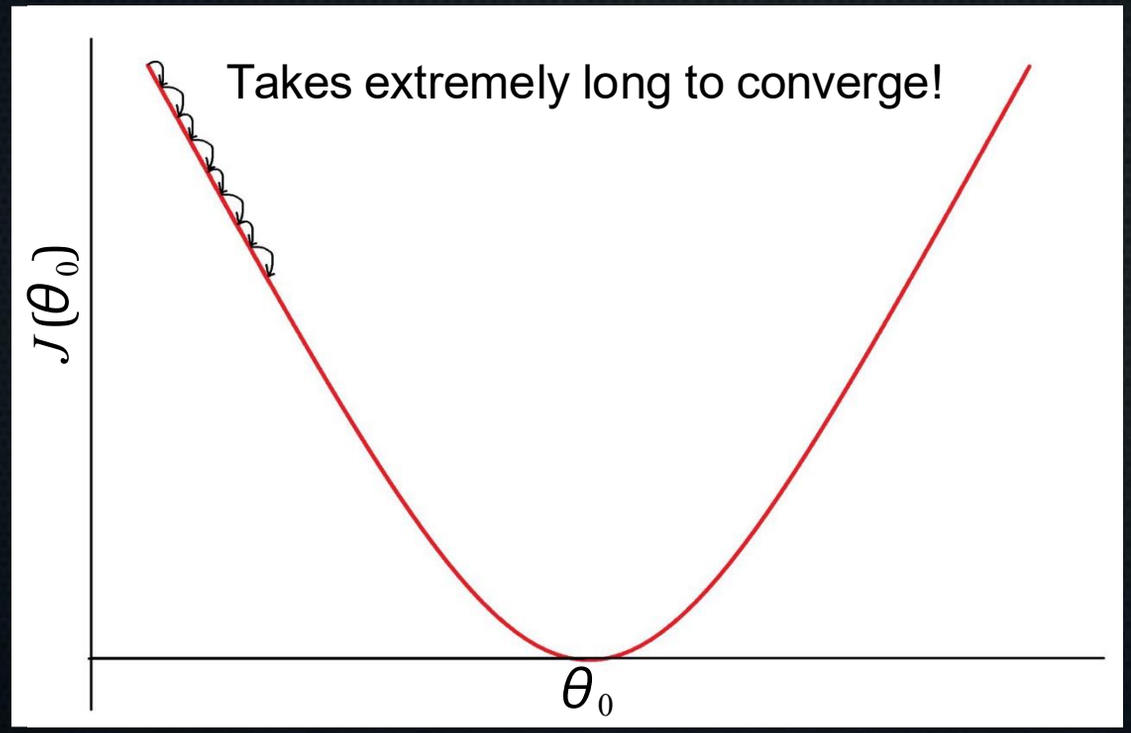
Source: Andrew Ng, Coursera

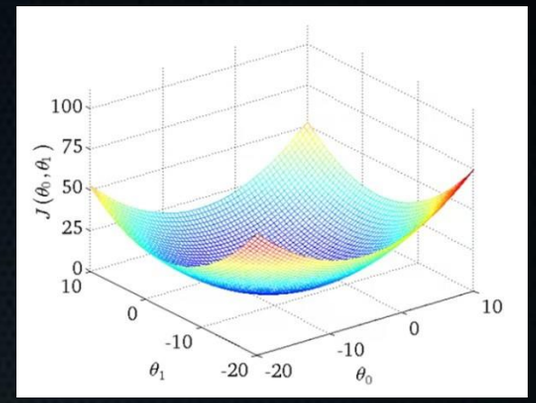
Source: https://towardsdatascience.com/univariate-linear-regression-theory-and-practice-99329845e85d

$$\theta_0=\theta_0-\alpha\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\theta_0-\frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})$$

What about α?

- Learning rate, so-called hyperparameter.
- Too low:



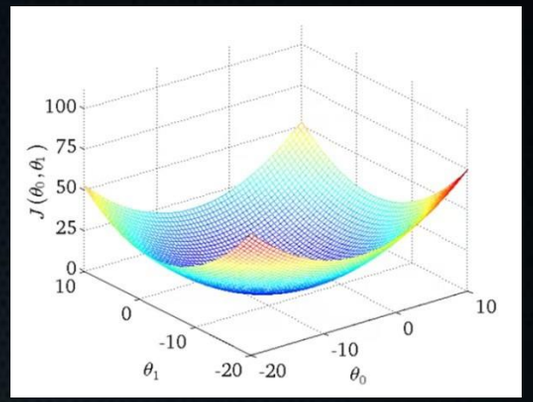


Source: Andrew Ng, Coursera

$$\theta_0 = \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \theta_0 - \frac{\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

What about α?

- Learning rate, so-called hyperparameter.
- Will discuss later how we pick it!



Source: Andrew Ng, Coursera

Summary

- Defined a cost function for linear regression
- Showed how gradient descent can be used to minimise this cost function by changing the parameters
- Calculated partial derivatives for use with gradient descent
- Encountered our first hyperparameter, α, which governs the size of update steps

Computer Labs

- Feel free to add code cells to experiment in, and always experiment before putting something in a function!
- Handy shortcuts:
  - Ctrl + Enter → run current code cell
  - Ctrl + b → add new cell below current cell
  - Ctrl + a → add new cell above current cell
  - Esc + m → turns selected cell into a markdown cell (for writing notes)
  - Shift tab: go one indentation level back on all selected lines
  - Tab: Go one indentation level deeper on all selected lines
  - You can toggle line numbers under View in the top menu bar.

Computer Labs

- You should now open and do *Day1/Practical_1/PracticalMaterialDay1_ShortPractical1.ipynb*
- DuckDuckGo and Google are your friend: numpy takes some getting used to, so search, search, search!

- I will start the next lecture in about an hour. If you're not finished: don't worry! Just continue where you left off after the lecture.