

Daily Inspiration

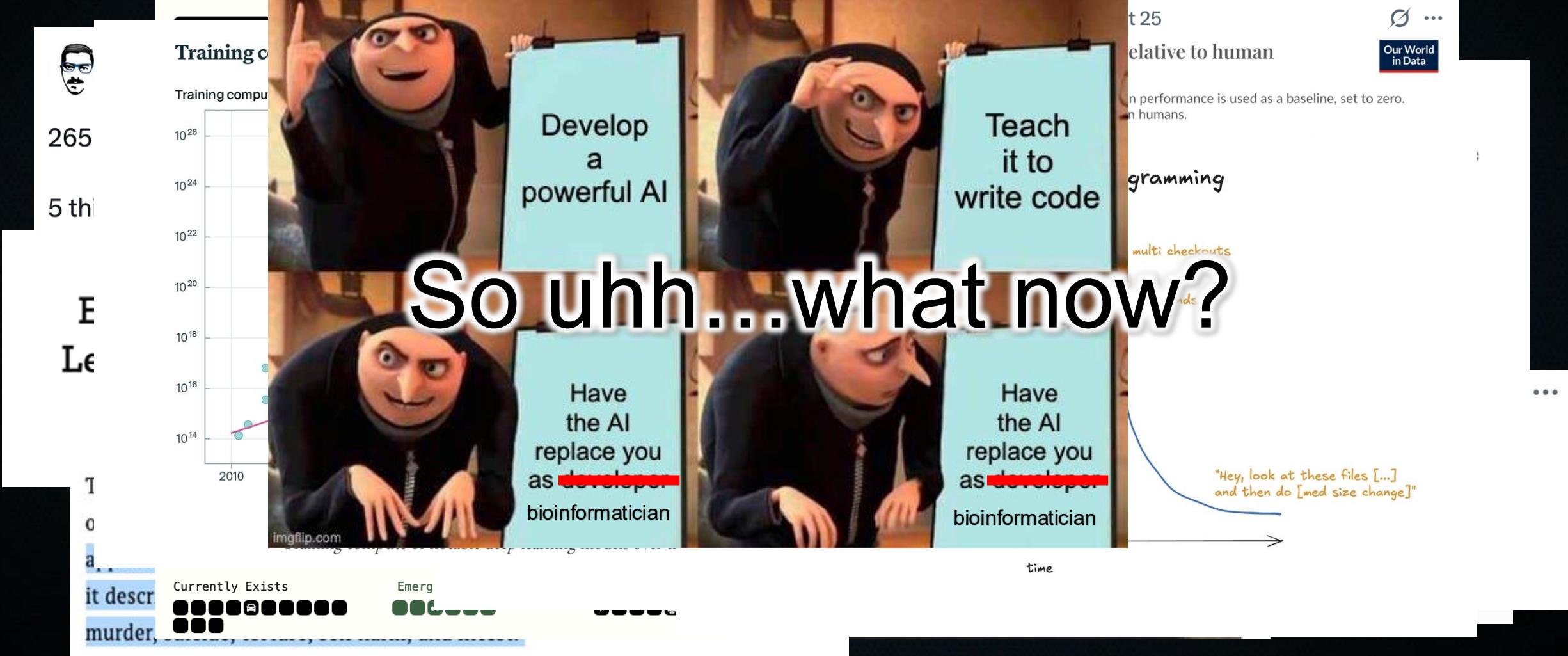
**JUST
DO IT!**



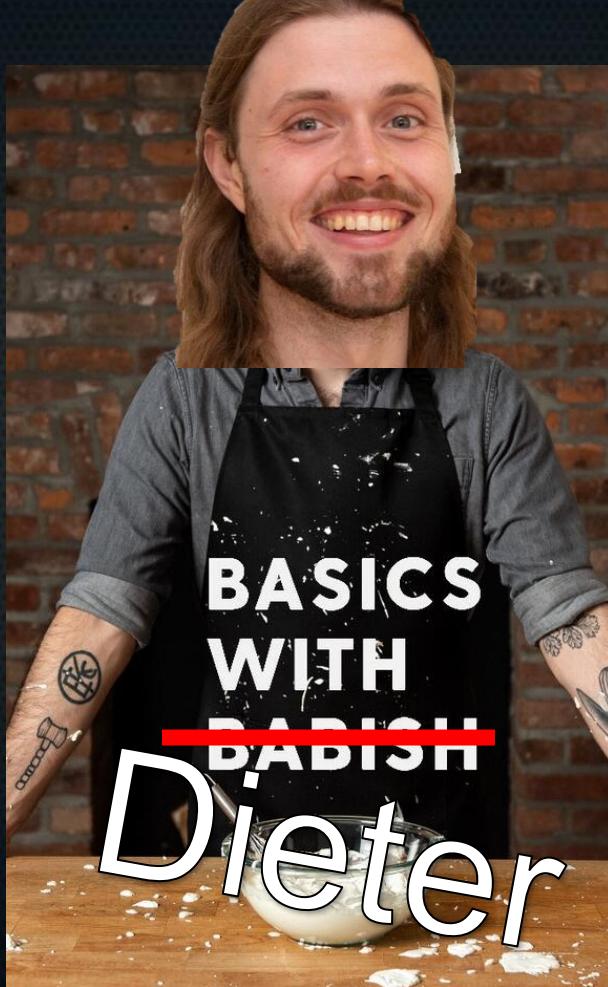
Today

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

Intro



Intro

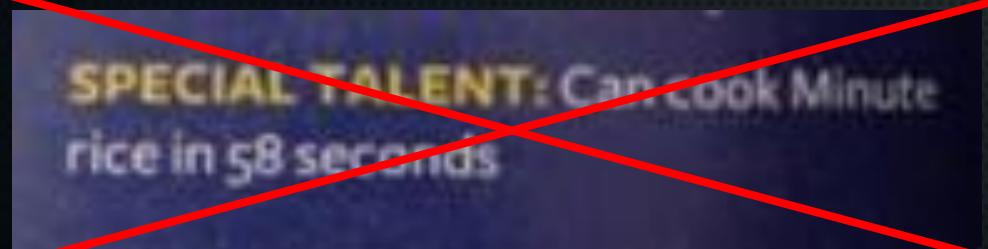


Who am I?

- 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?

The screenshot shows a website page for the UMC Utrecht Center for Molecular Medicine, specifically the De Ridder Group. The header includes the UMC Utrecht logo, the text "UMC Utrecht Center for Molecular Medicine", and "De Ridder Group". On the right side of the header are links for "Research", "Publications", "Group members", "Vacancies", and "Contact". The main content area features a portrait of a man with long hair and a beard wearing a yellow party hat with a cartoon character on it. To the right of the photo is the name "Dieter Stoker" and the title "PhD-Student". Below this is a detailed biography of Dieter Stoker, mentioning his education at Utrecht University, his work in the de Ridder lab, and his interests in machine learning, reading Terry Pratchett, and playing video games. At the bottom of the profile is the email address "D.G.G.Stoker-6@umcutrecht.nl".

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

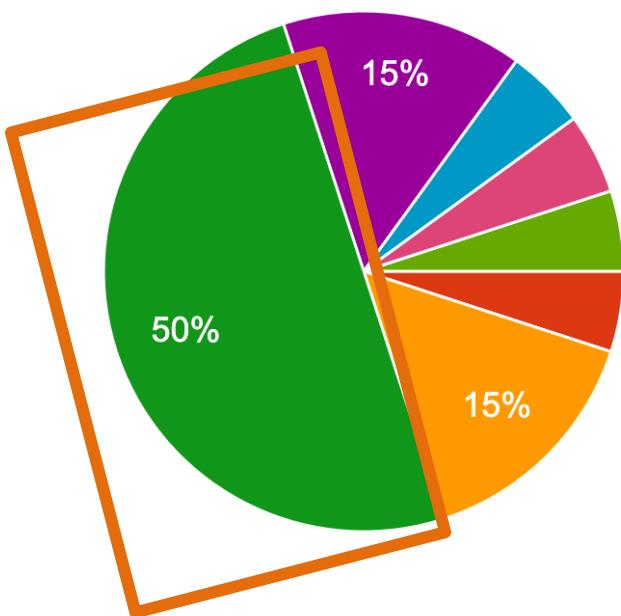


Dieter Stoker,
PhD candidate
man

Who are you?

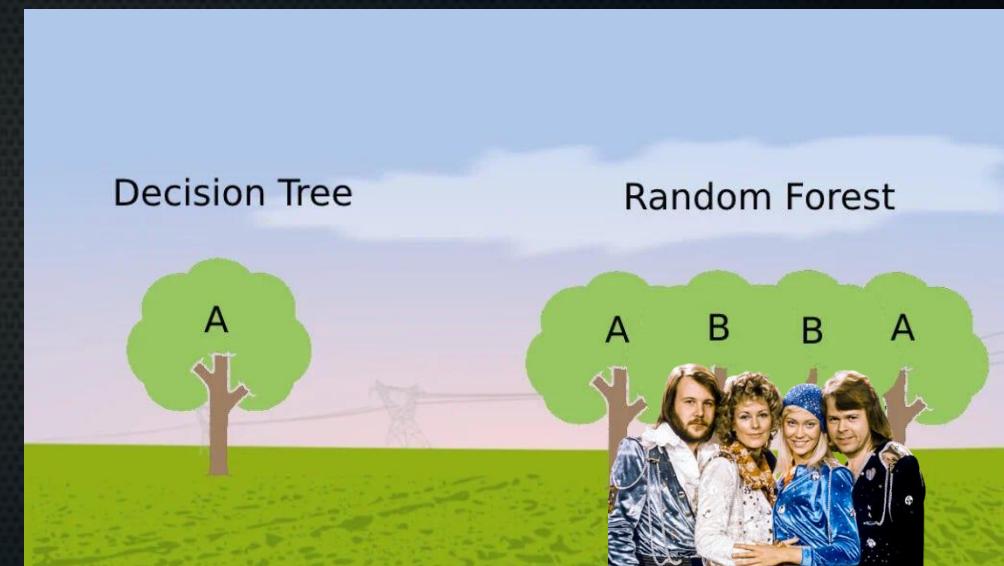
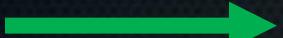
What's your level of ML experience?

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 reimplementations...
- 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 learn...
- Well, I don't think my buddies know "engineering"
- I have worked with the Snowflake machine learning...

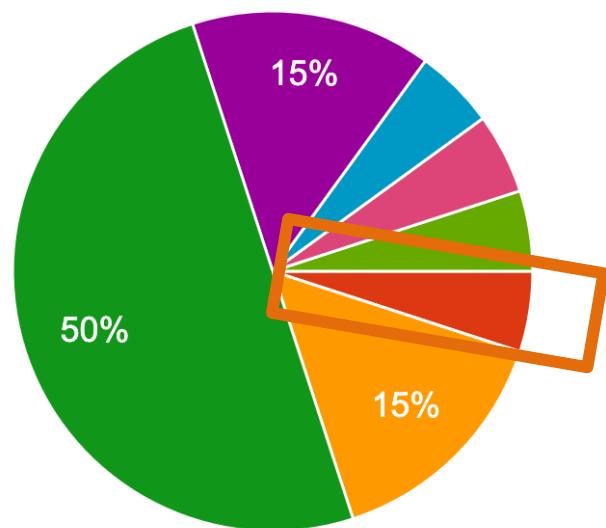
Who are you?



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 Omnisiah 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.



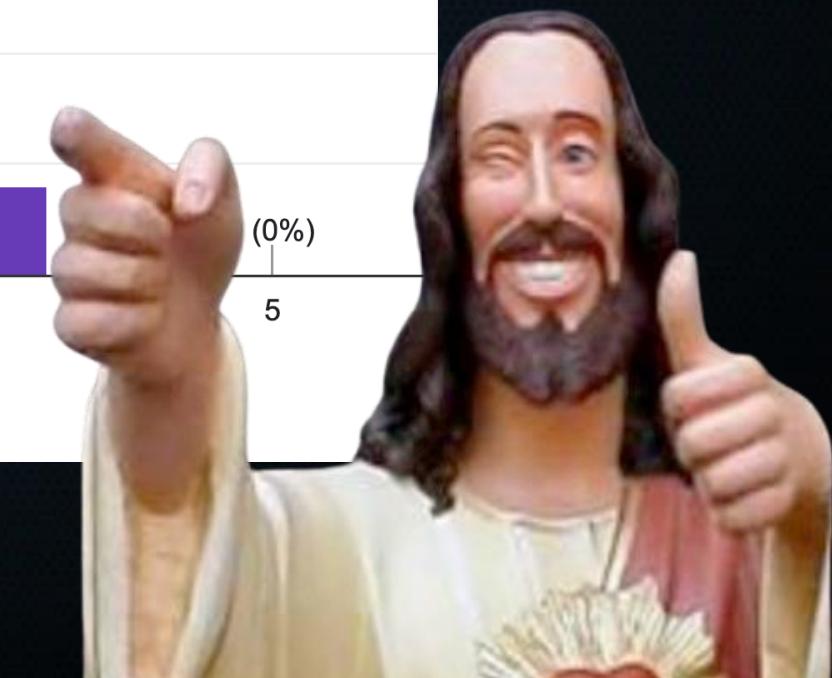
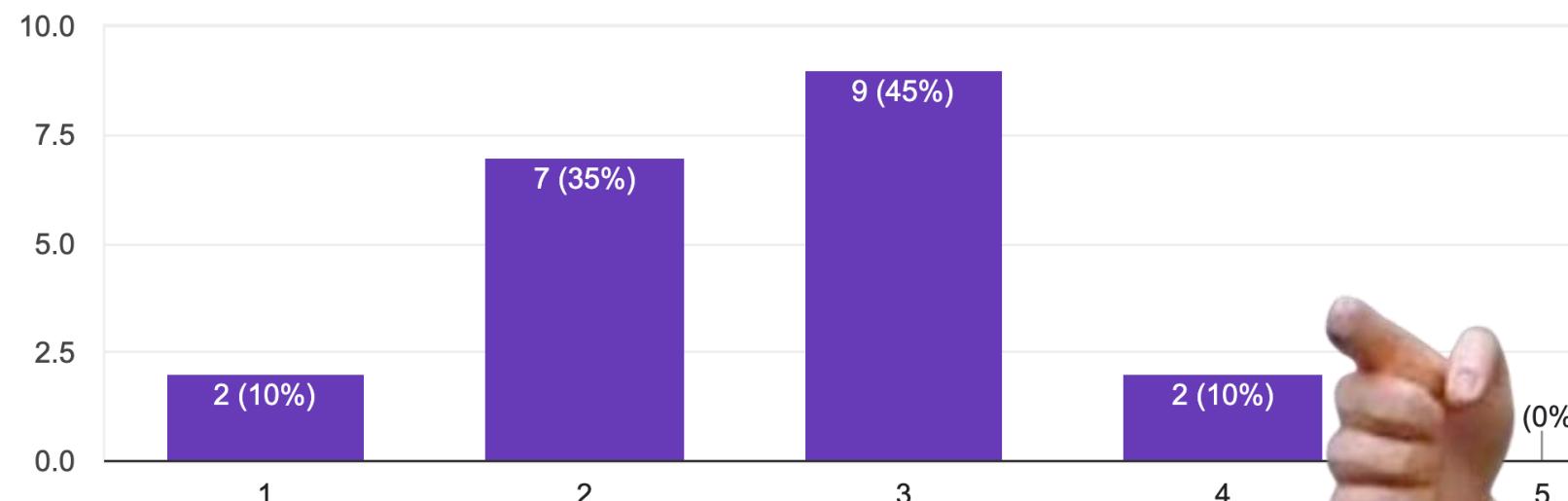
Who are you?



Who are you?

How do you feel about partial derivatives?

20 responses



Who are you?

Article | Published: 14 March 2018

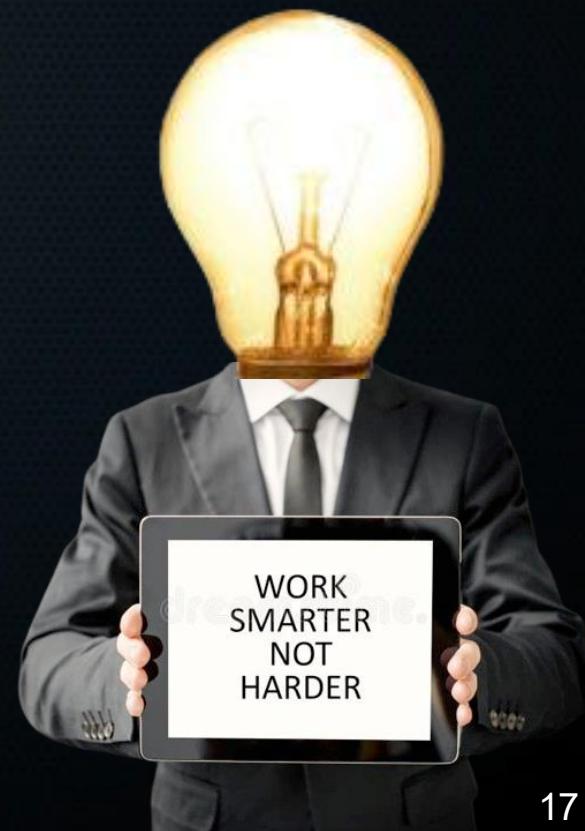
DNA methylation-based classification of central nervous system tumours

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. Braczyński, 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.'



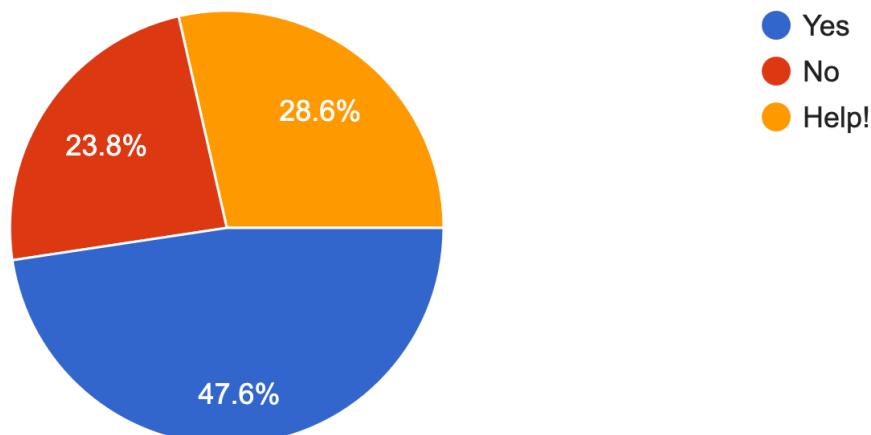
Who are you?



Git or git gud?

Bro, do you even git clone?

21 responses



Who are you?

Deep learning?

W

19

NOOOOOOO!!!!!! YOU CAN'T FOCUS
ON H100s, THEY'RE SO EXPENSIVE
AND OUT OF REACH FOR
OPEN-SOURCE!!! YOU'RE LIMITING
ACCESS TO CUTTING-EDGE
TECHNOLOGY FOR THOSE WHO
INNOVATE FOR THE PUBLIC
GOOD!!! HOW CAN WE ADVANCE
OPEN TECHNOLOGIES IF THE
TOOLS ARE JUST FOR THE
ELITE??? STOPPPPPPPPPPPPPP!



haha gpus go brr



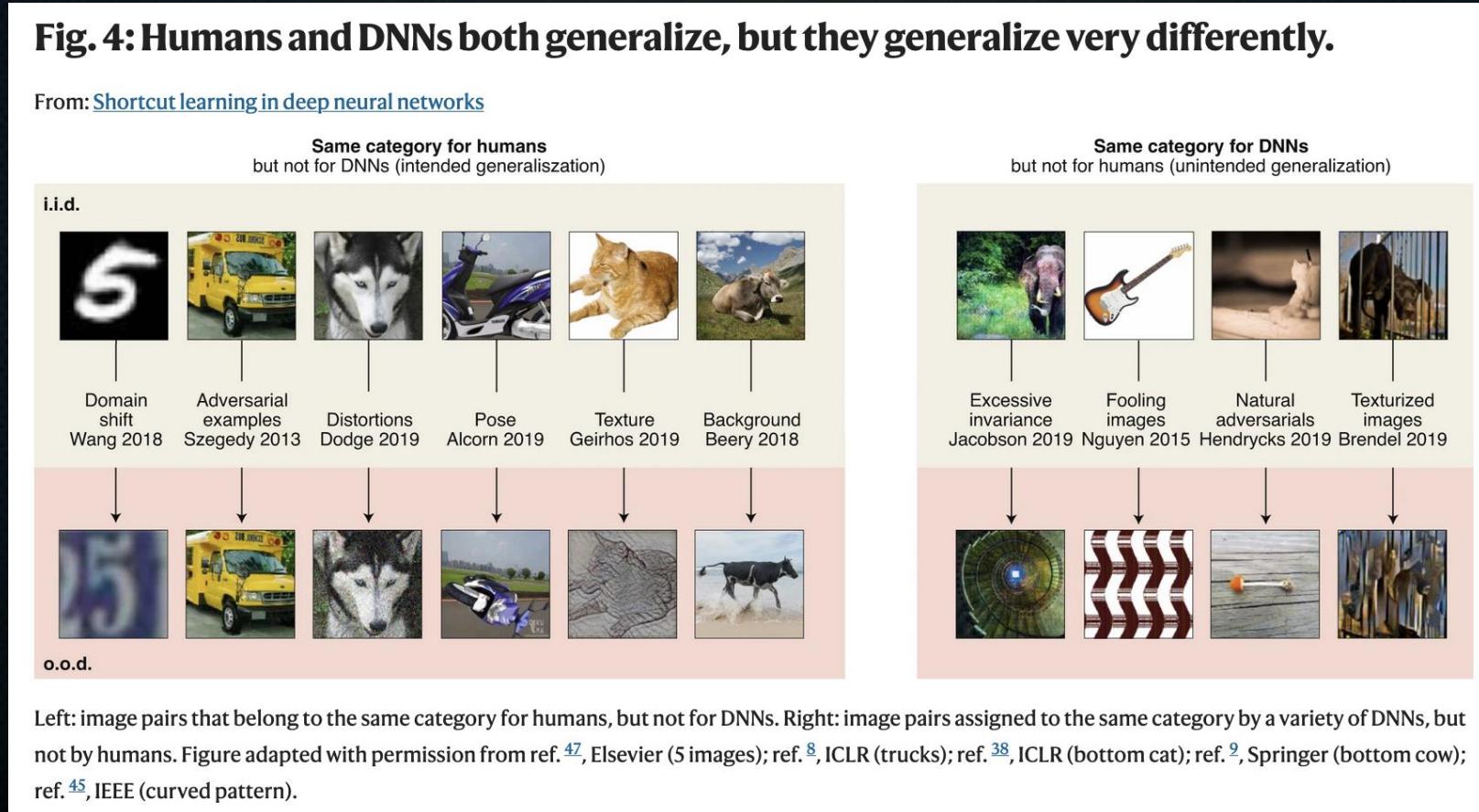
Figure 1: brr

Your questions

What about bias?

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

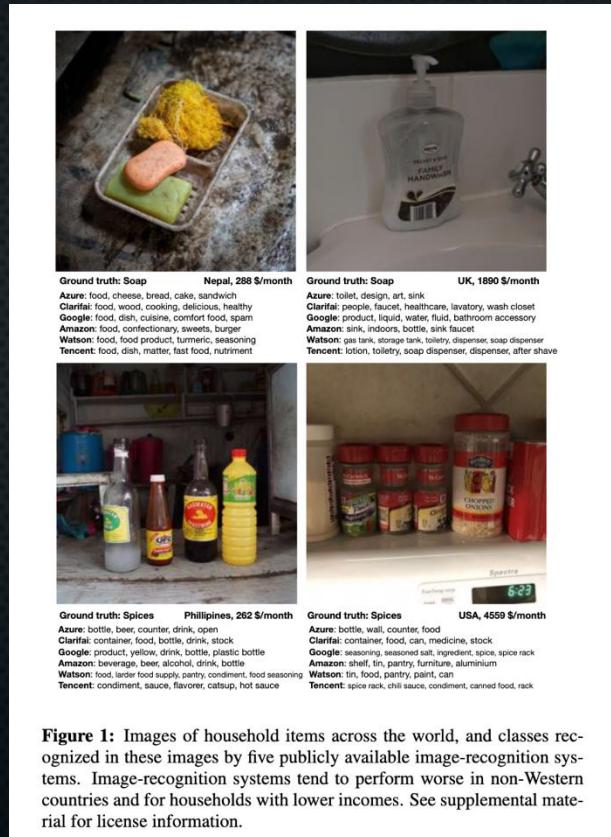
From: [Shortcut learning in deep neural networks](#)



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

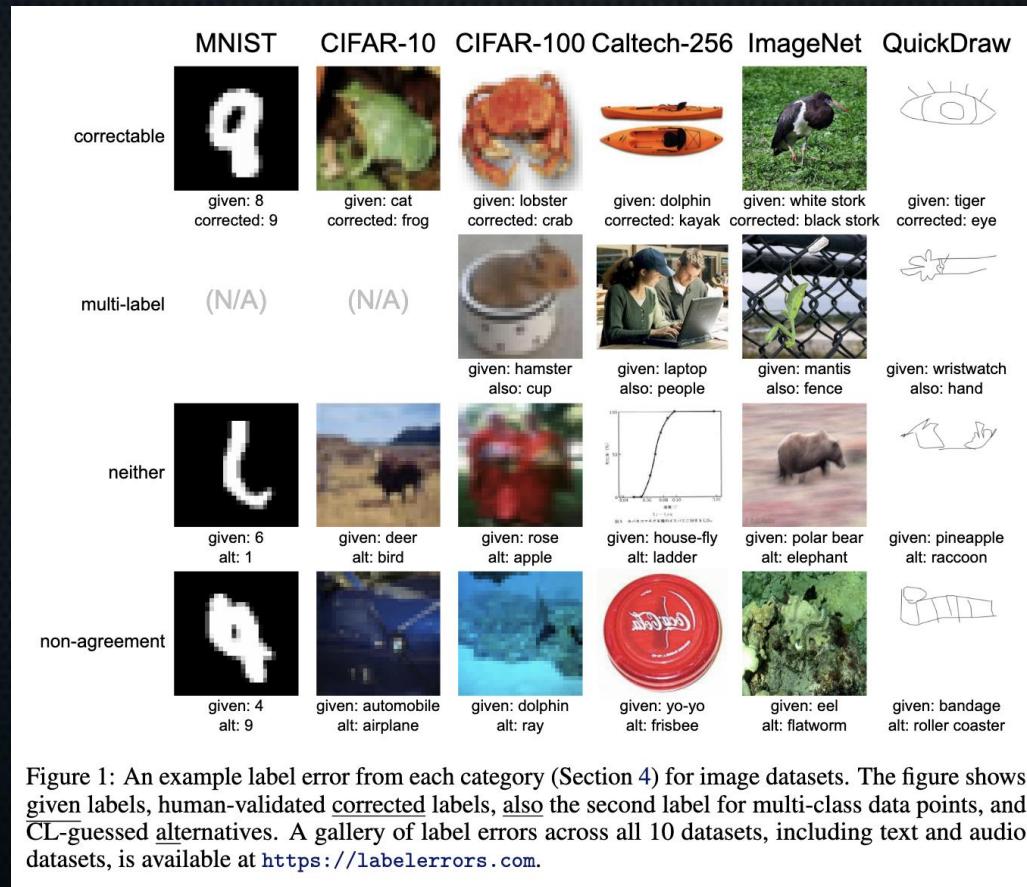
Your questions

What about bias?



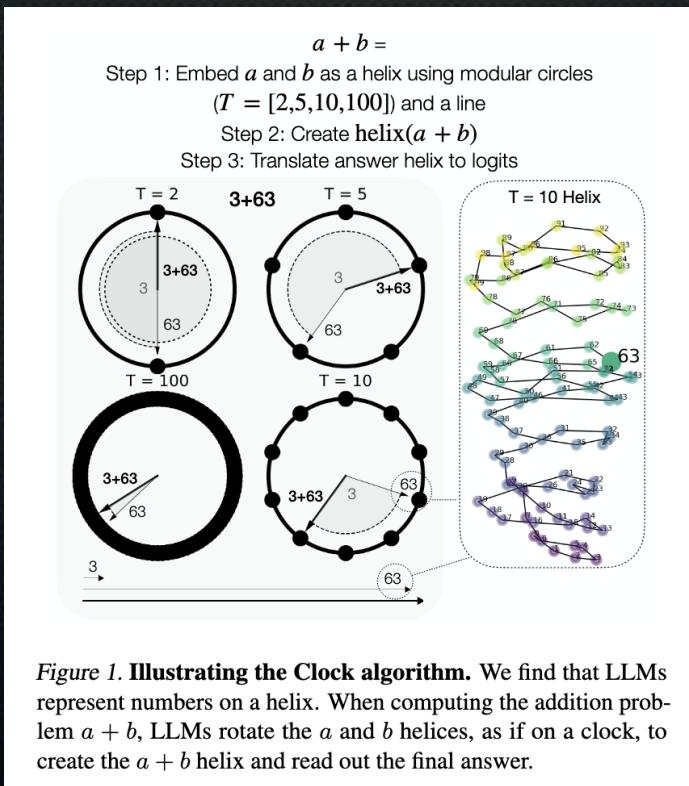
Your questions

What about bias?



Your questions

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



“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

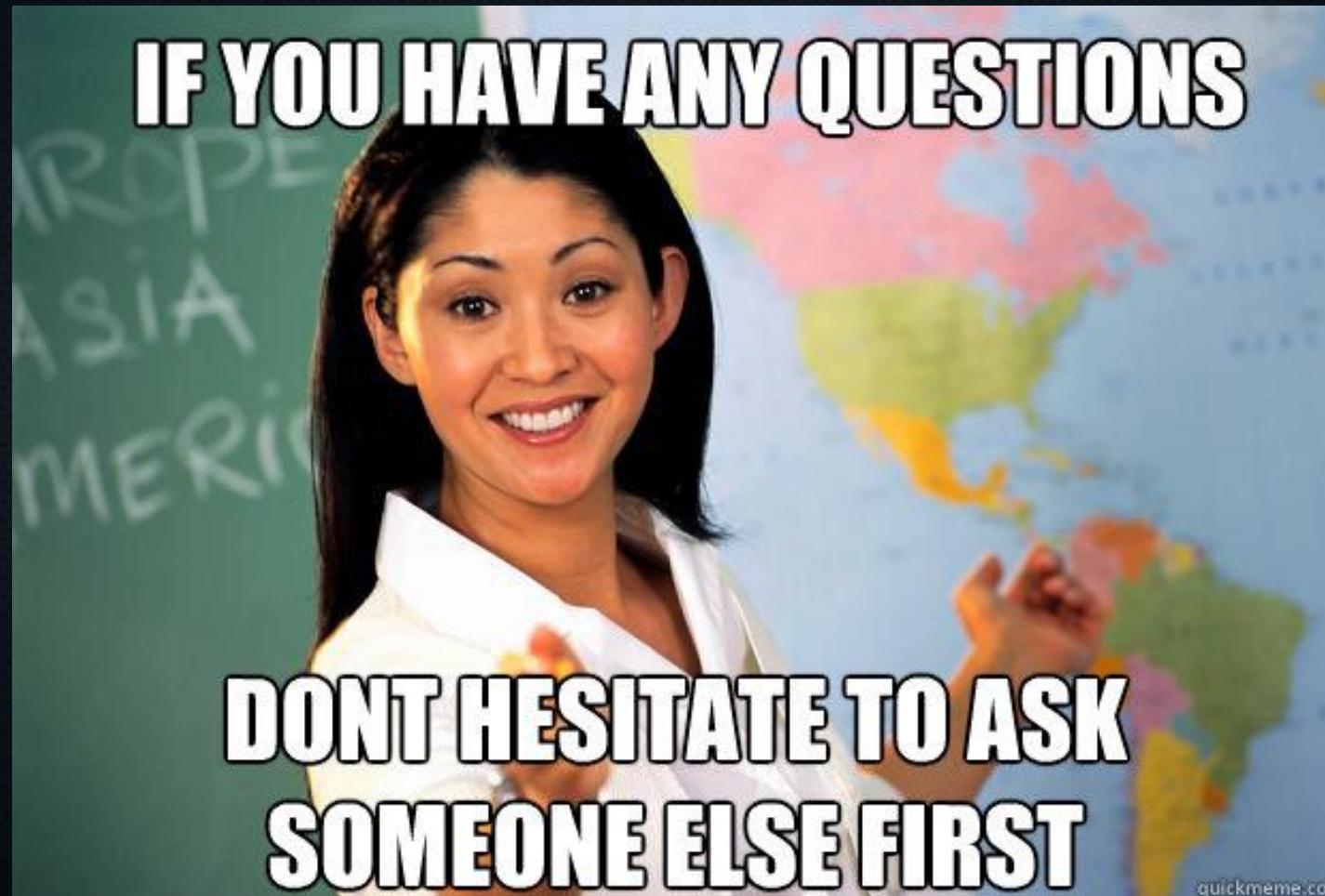
Hello yes diffusion models plz?

The screenshot shows a YouTube video player. At the top, there is a thumbnail of a cartoon character with large eyes and a small body. Below the thumbnail, the video title is "Diffusion models explained. How does OpenAI's GLIDE work?". Under the title, there is a profile picture of a person and the channel name "AI Coffee Break with Letitia" followed by "59.6K subscribers". To the right of the channel info are buttons for "Join", "Notification bell", and "Share". Below the title, there is a large image of a presentation slide. The slide has a dark background with a white header section. The header contains a yellow lightbulb icon, the text "Link in the description below.", the title "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", the authors' names "Alex Nichol * Prafulla Dhariwal * Aditya Ramesh * Prahar Shyam * Pamela Mishkin * Bob McGrew * Ilya Sutskever * Mark Chen", and the date "CVPR 2021 22 Dec 2021". The main content of the slide is divided into two sections: "Abstract" and "Figure 1". The "Abstract" section discusses the capabilities of diffusion models, mentioning classifier-free guidance and its ability to generate photorealistic images. It also notes that the model can produce multiple concepts in a single image. The "Figure 1" section shows a grid of 12 generated images with their corresponding text prompts. The images include a cat playing checkers, a sunset over a canyon, a psychedelic hamster dragon, a superhero Albert Einstein, a boat in Venice, a fox in a starry night, a red cube on a blue cube, a panda in a stained glass window, a space elevator drawing, a futuristic city in synthwave style, a pixel art corgi pizza, and a foggy New York City skyline.



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

More?



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

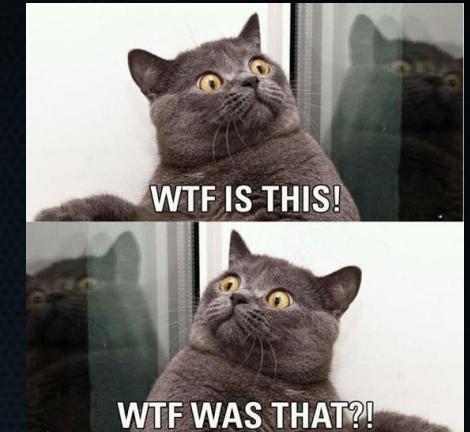
- Morning/early afternoon:
 - Lectures of ~45-60 minutes, interspersed with (2) short practical(s).
- Rest of the day:
 - Somewhat longer afternoon practical
- Taken together:
 - Lecture (09:00-09:45)
 - Short practical 1 (09:45-10:45)
 - Lecture (10:45-11:30)
 - Short practical 2 (11:30-12:30)
Lecture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

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



**BRUTE
FORCING**

**MACHINE
LEARNING**

**DEEP
LEARNING**

TL;DR Machine Learning

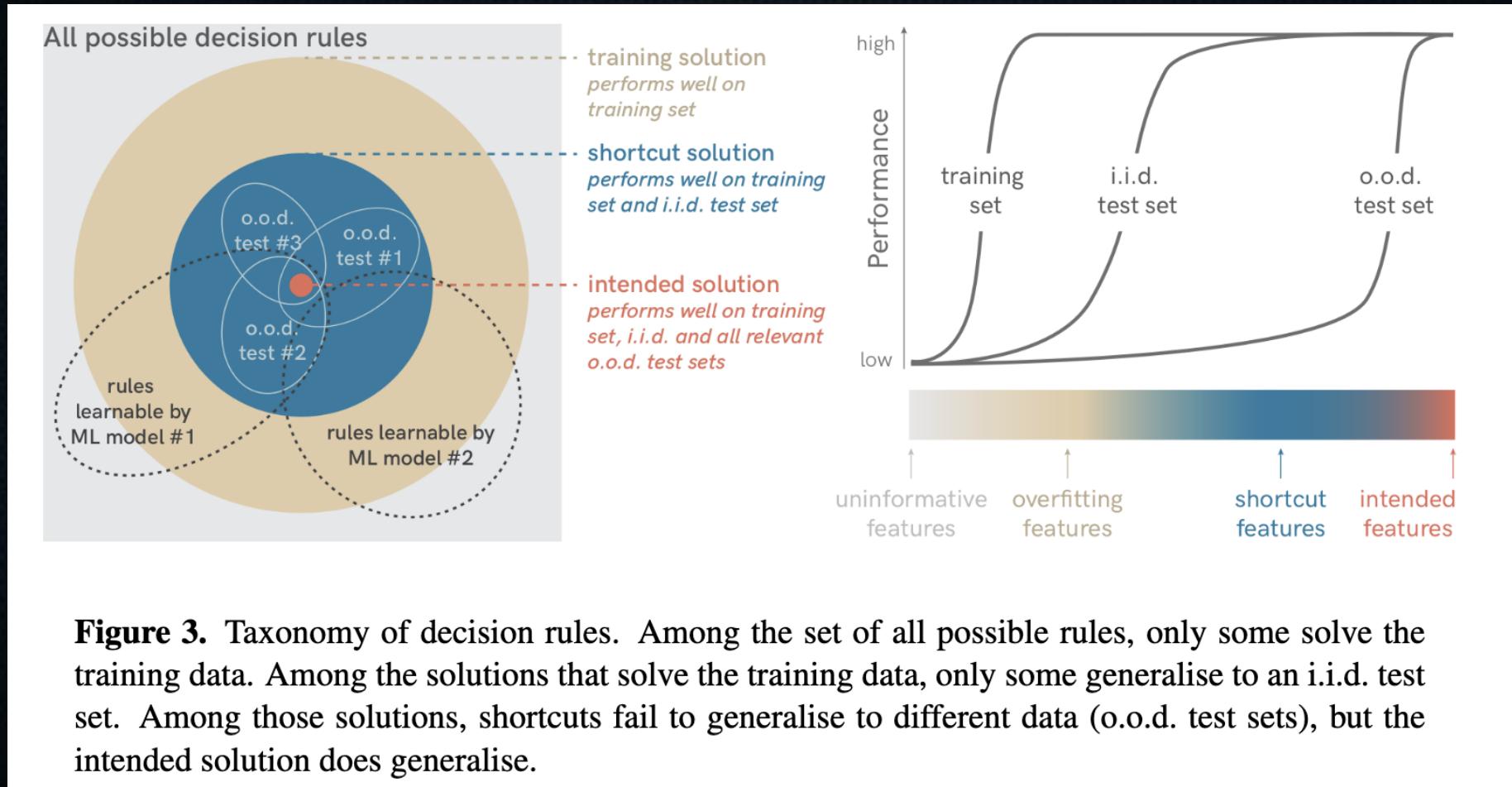


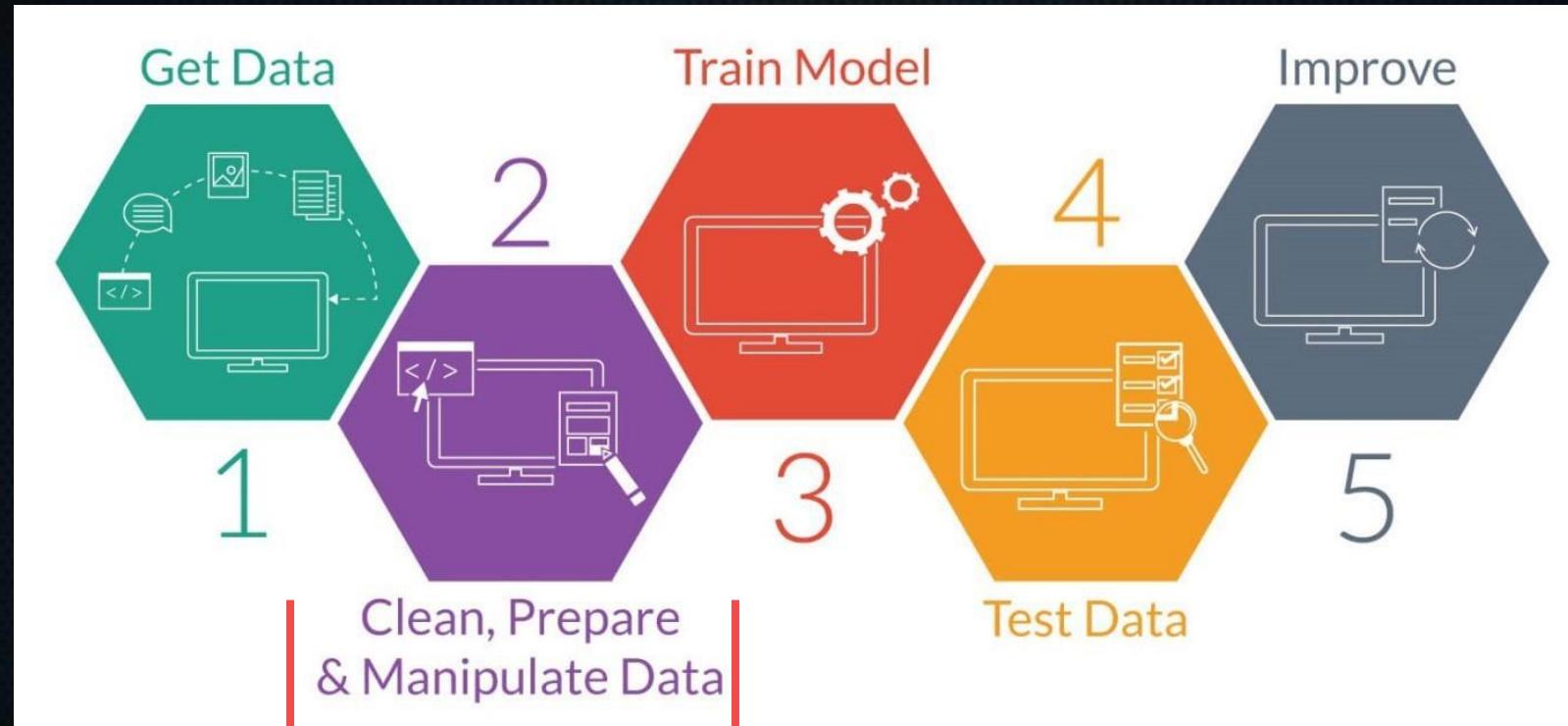
Figure 3. Taxonomy of decision rules. Among the set of all possible rules, only some solve the training data. Among the solutions that solve the training data, only some generalise to an i.i.d. test set. Among those solutions, shortcuts fail to generalise to different data (o.o.d. test sets), but the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



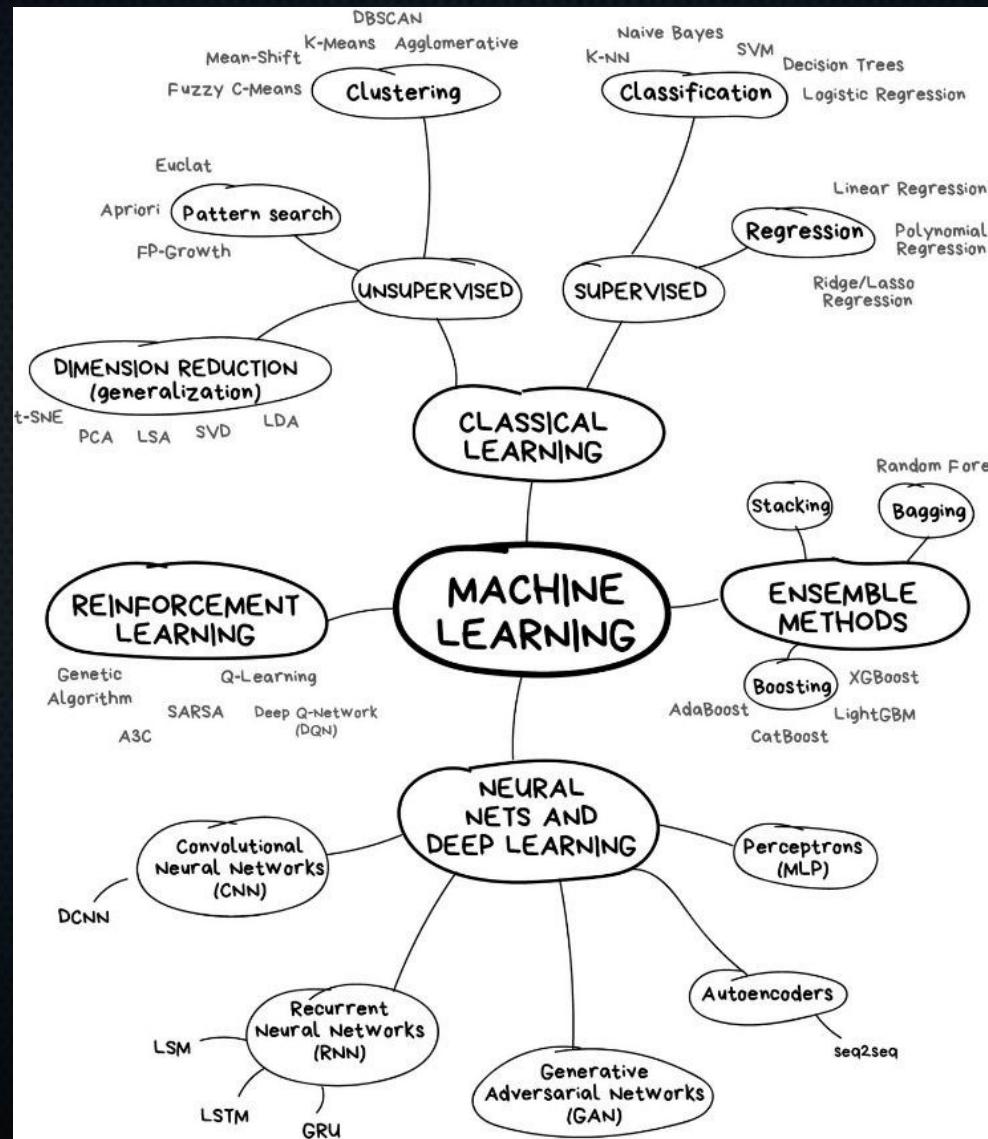
This presentation

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

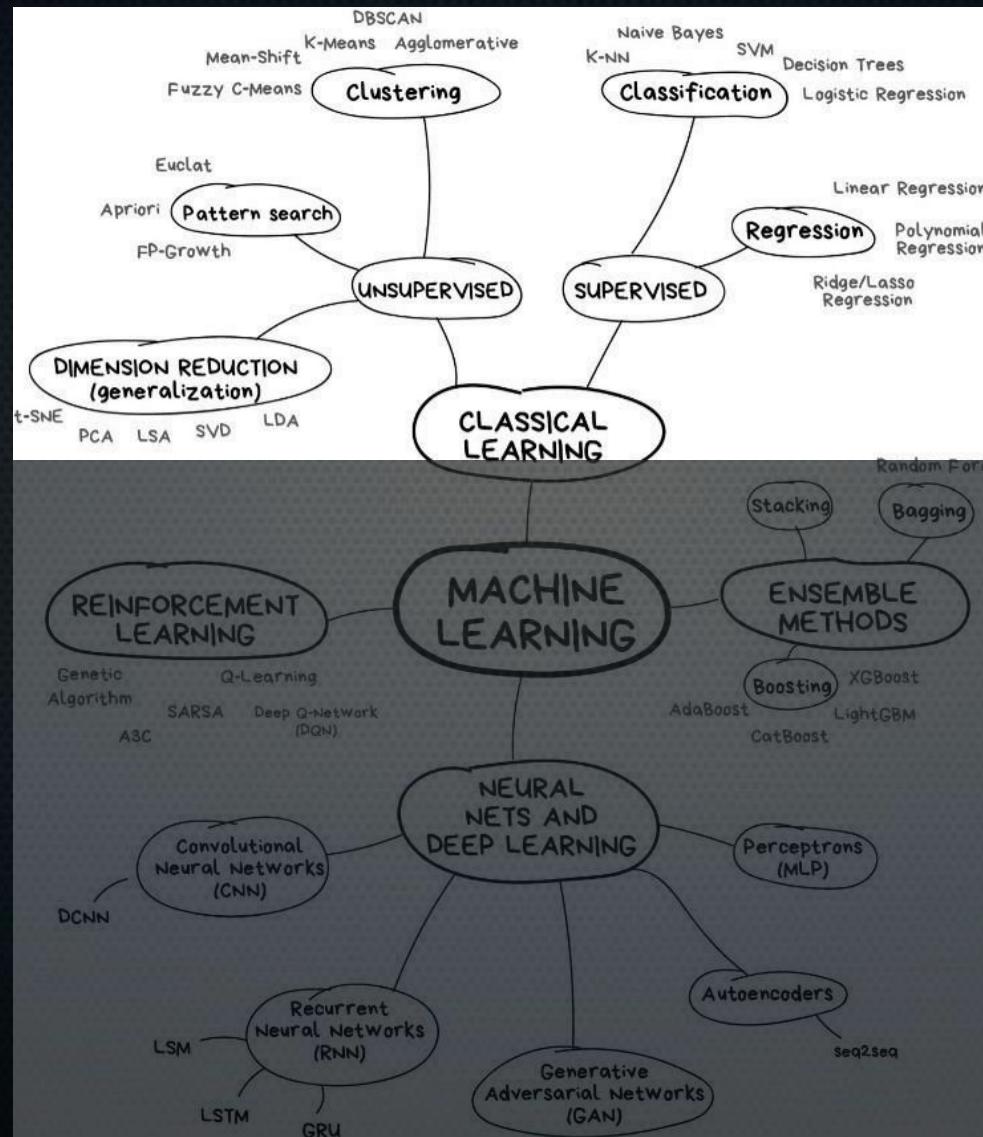
The ugly truth about ML



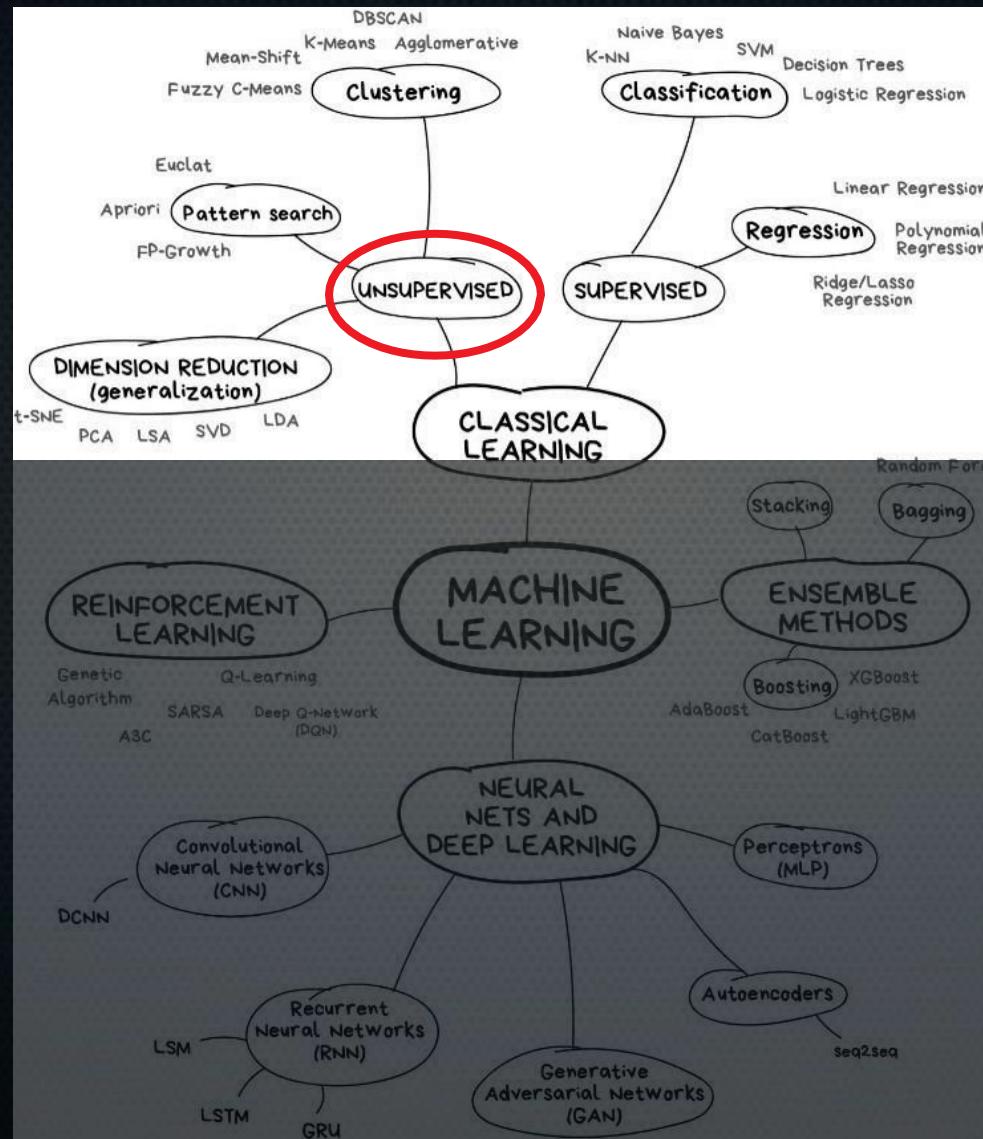
Map of Machine Learning



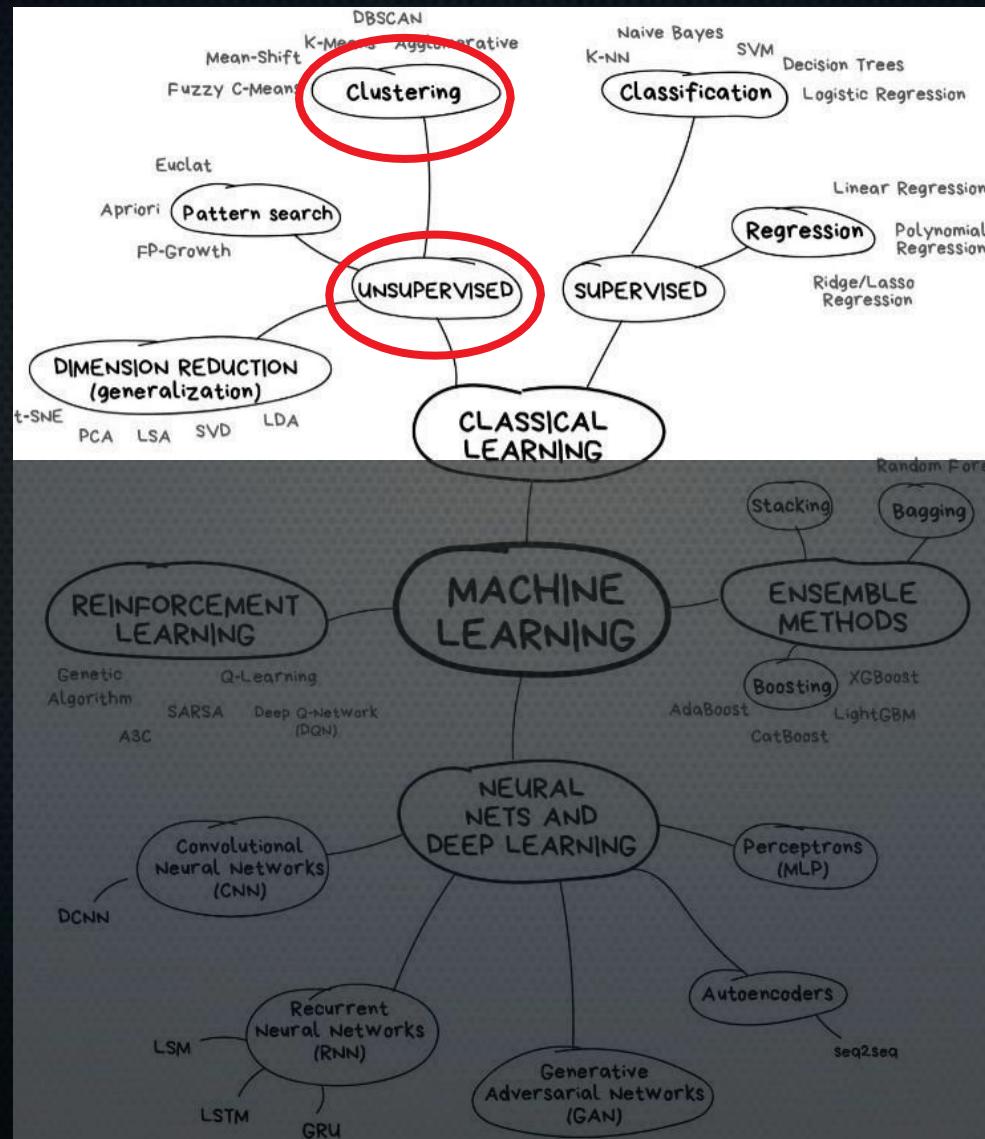
Map of Machine Learning



Map of Machine Learning

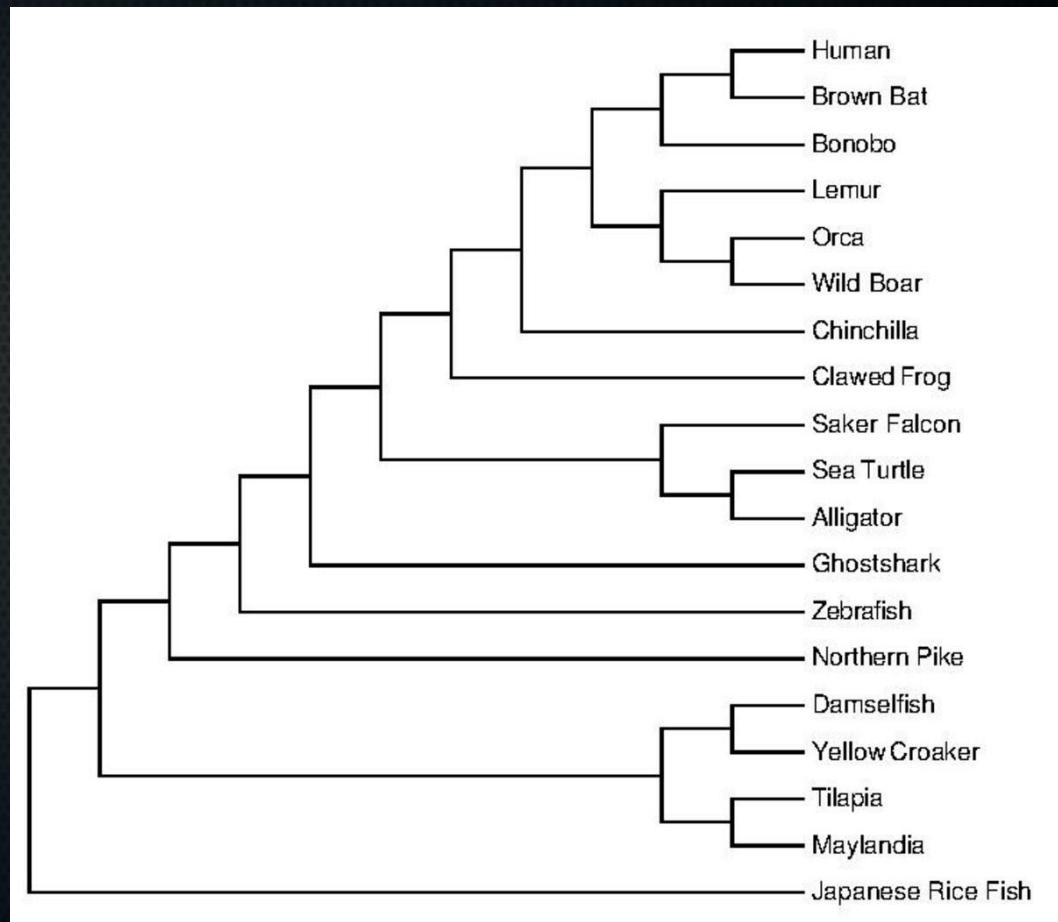


Map of Machine Learning



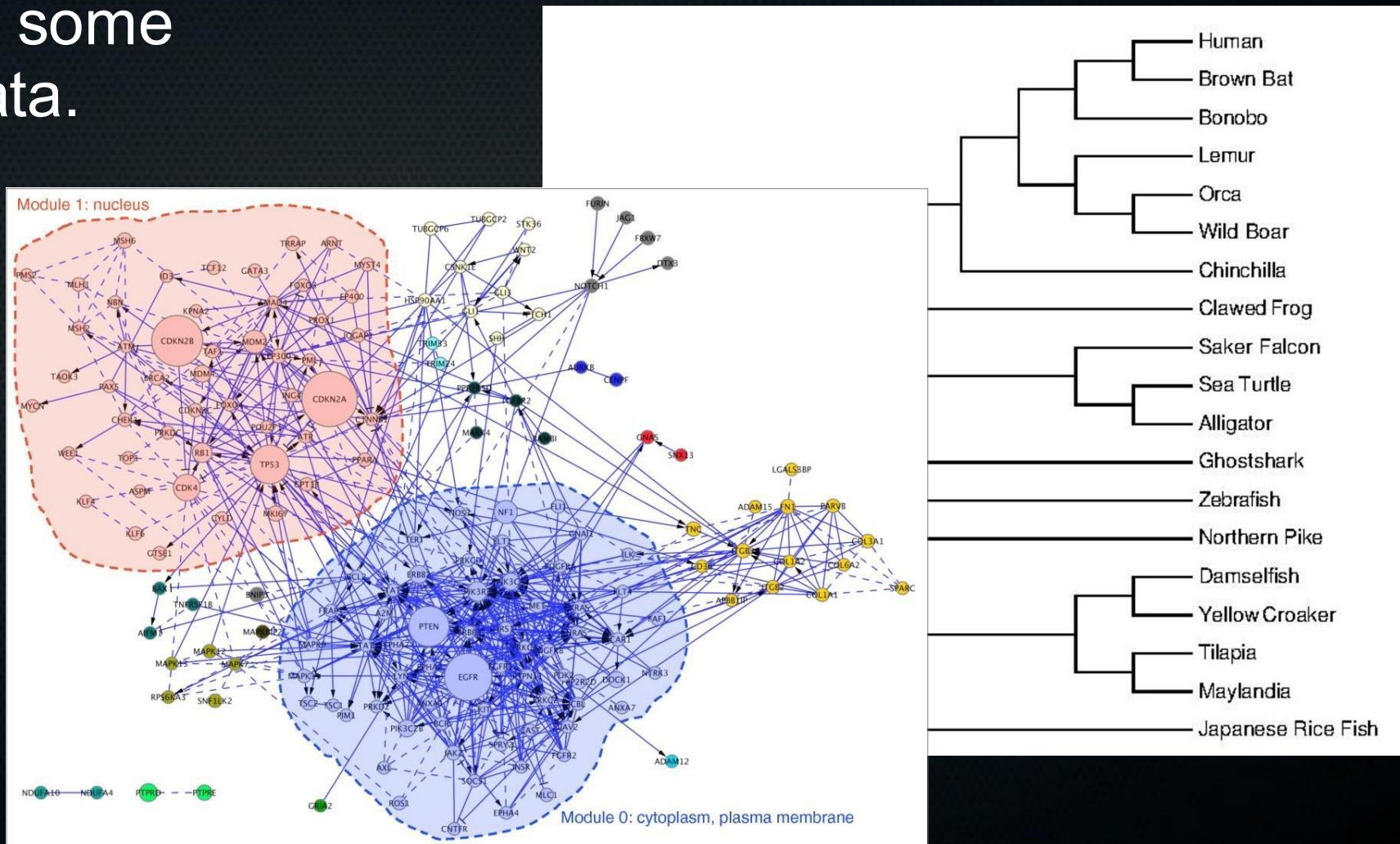
Unsupervised learning: clustering

- Automatically find some structure in the data.



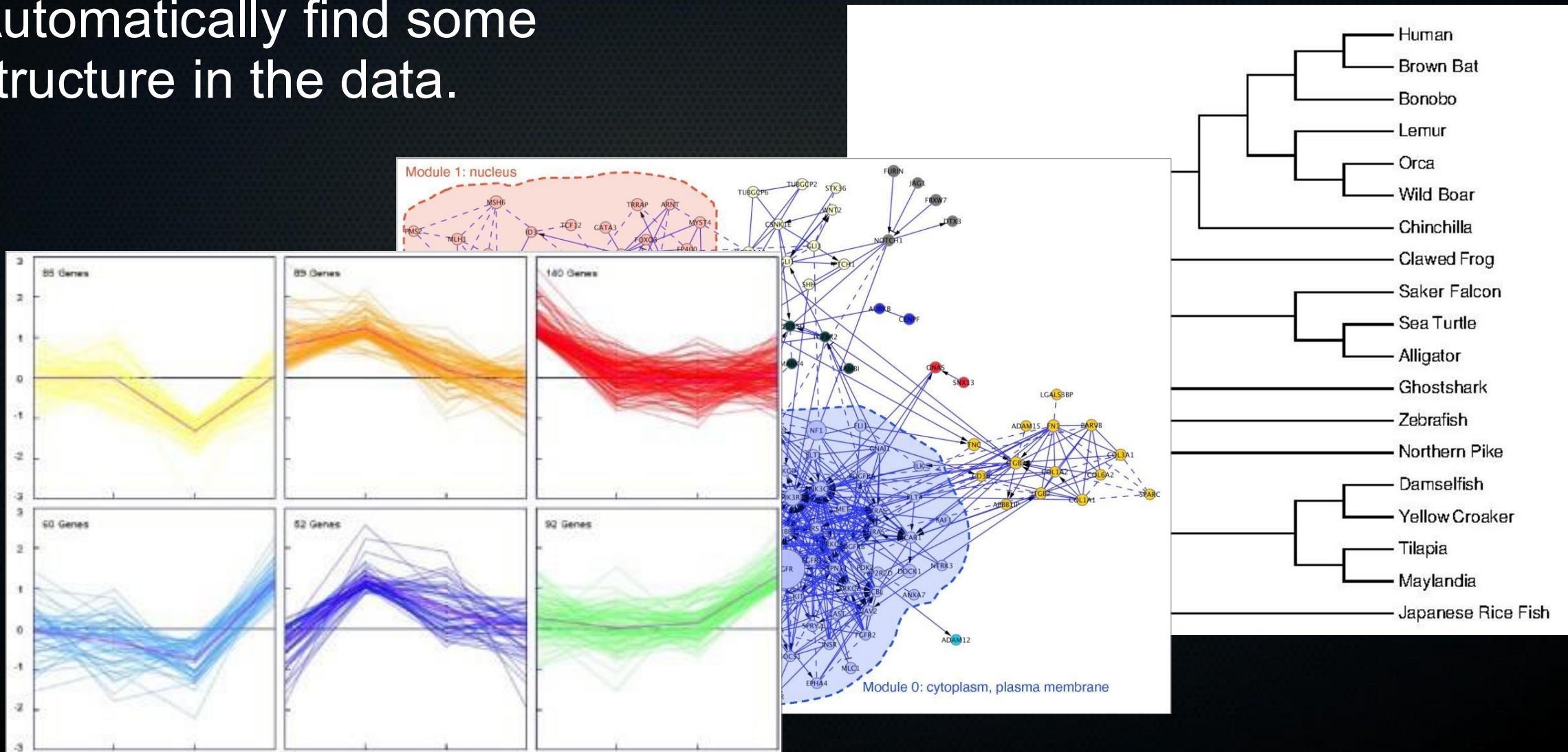
Unsupervised learning: clustering

- Automatically find some structure in the data.



Unsupervised learning: clustering

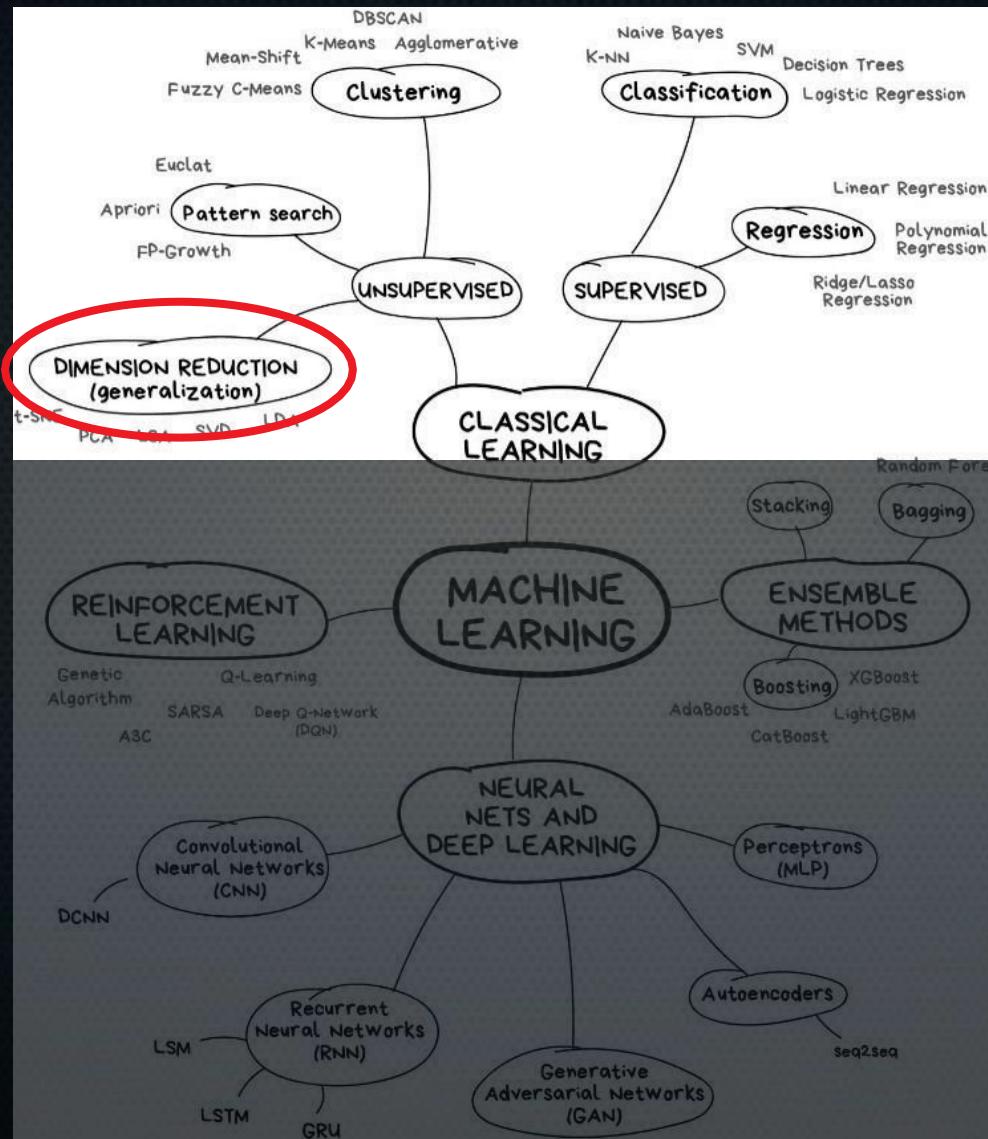
- Automatically find some structure in the data.



Unsupervised learning: clustering

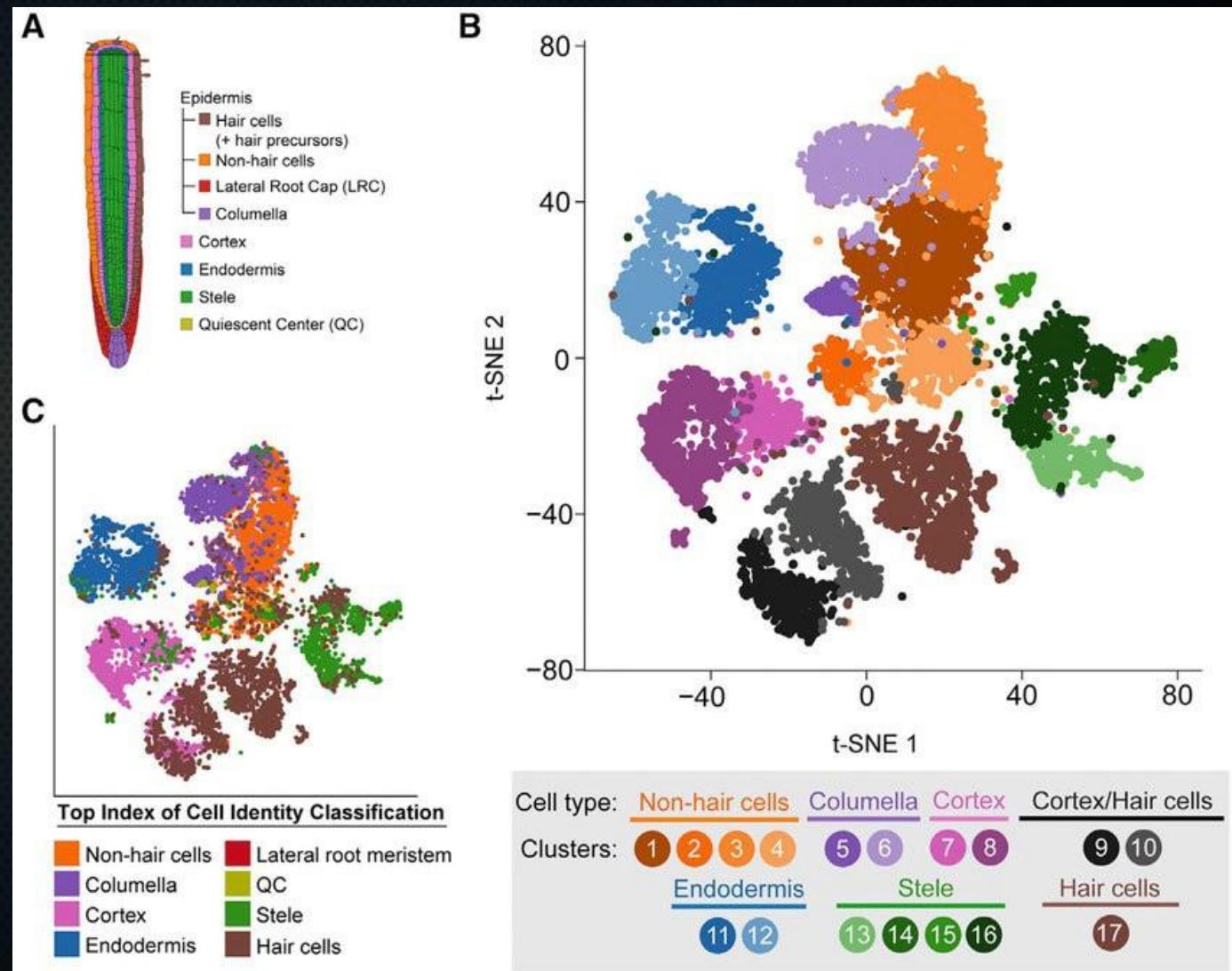
- No right or wrong:
 - Back-and-forth between different clustering algorithms, your knowledge, and the data.
 - You don't *know* correct clustering.

Map of machine learning



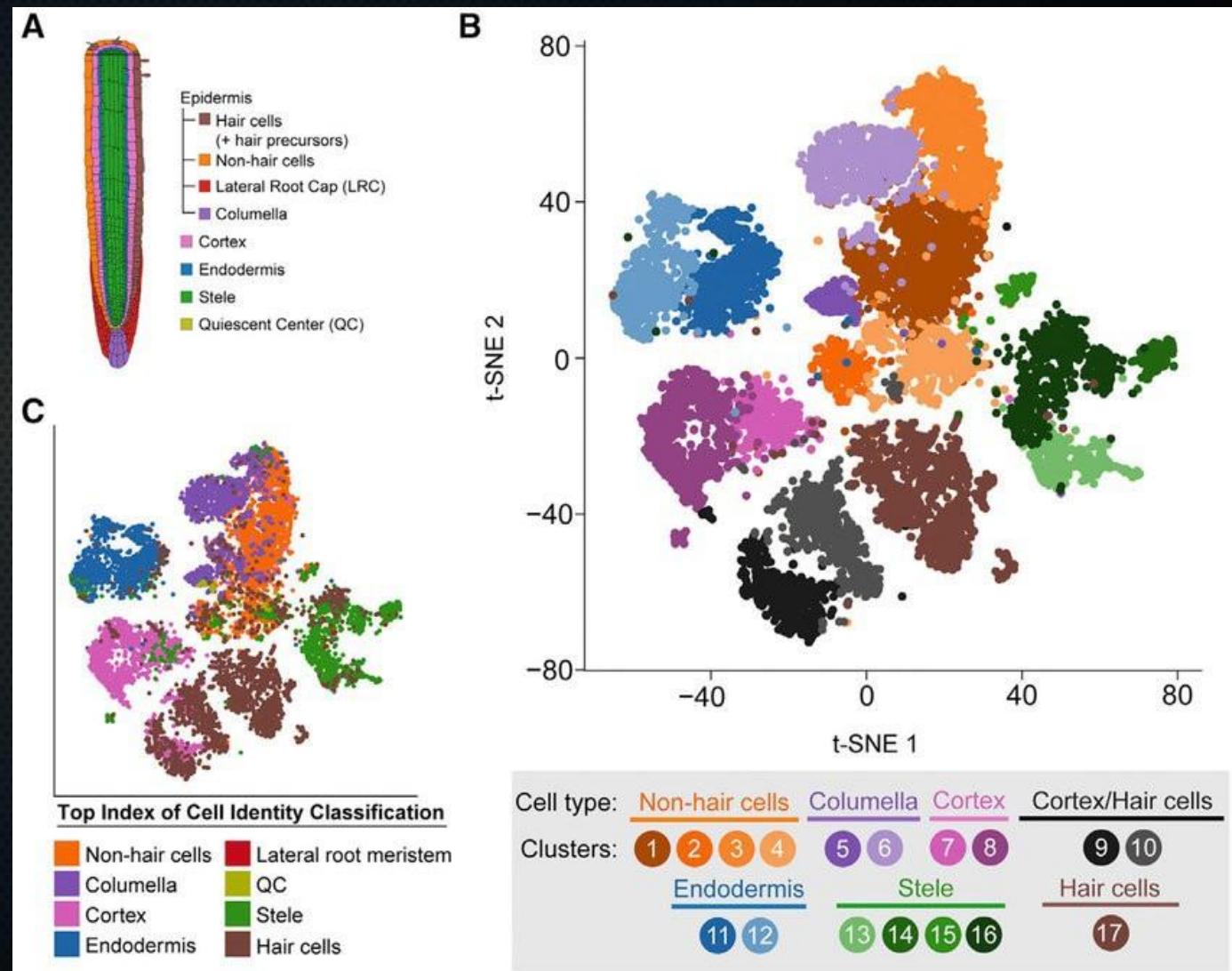
Unsupervised learning: dimensionality reduction

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



Unsupervised learning: dimensionality reduction

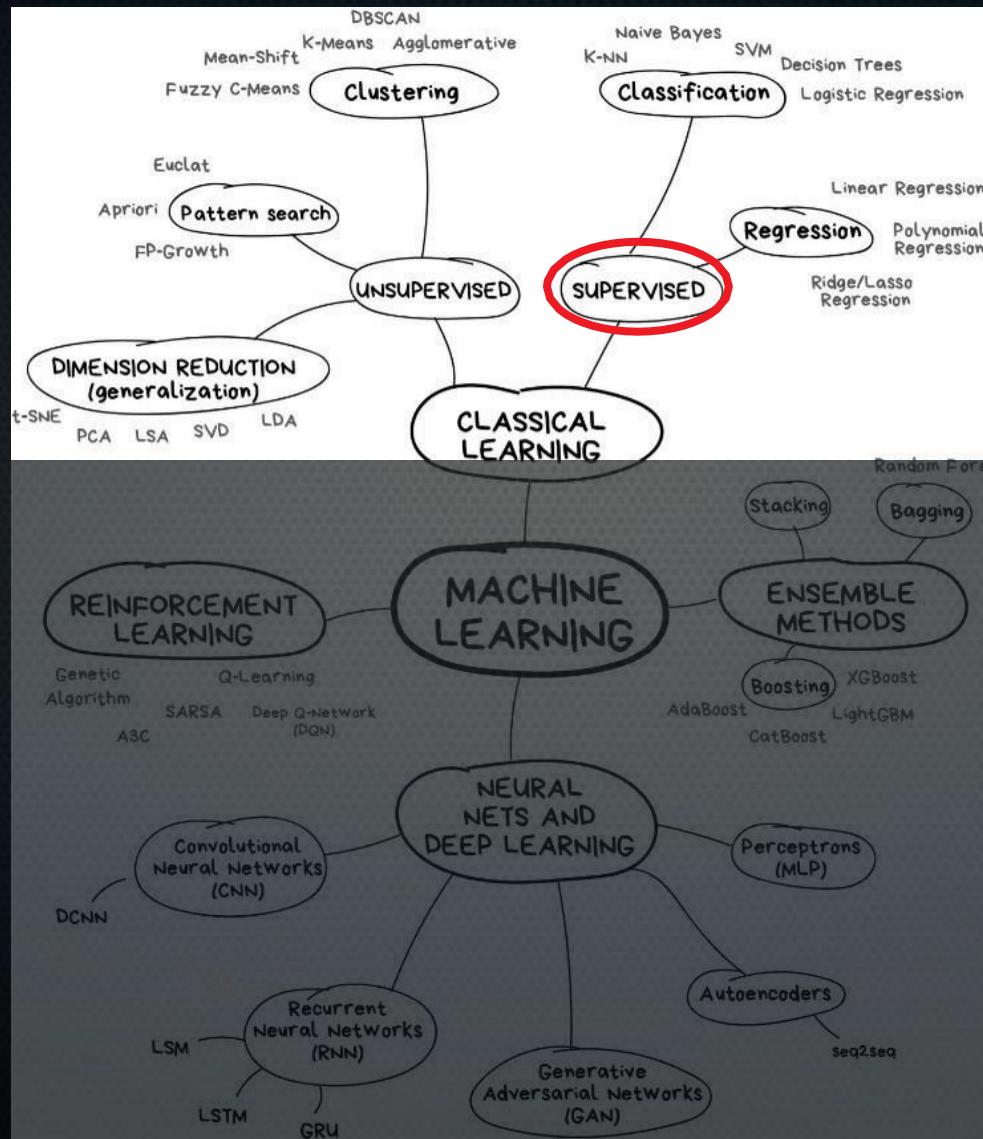
- Single-cell RNAseq of 12,198 *Arabidopsis* root cells.
- How do they differ?
- Visualise differences in all RNAs between all these cells in 2 dimensions.
- Used in conjunction with clustering (colours)



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

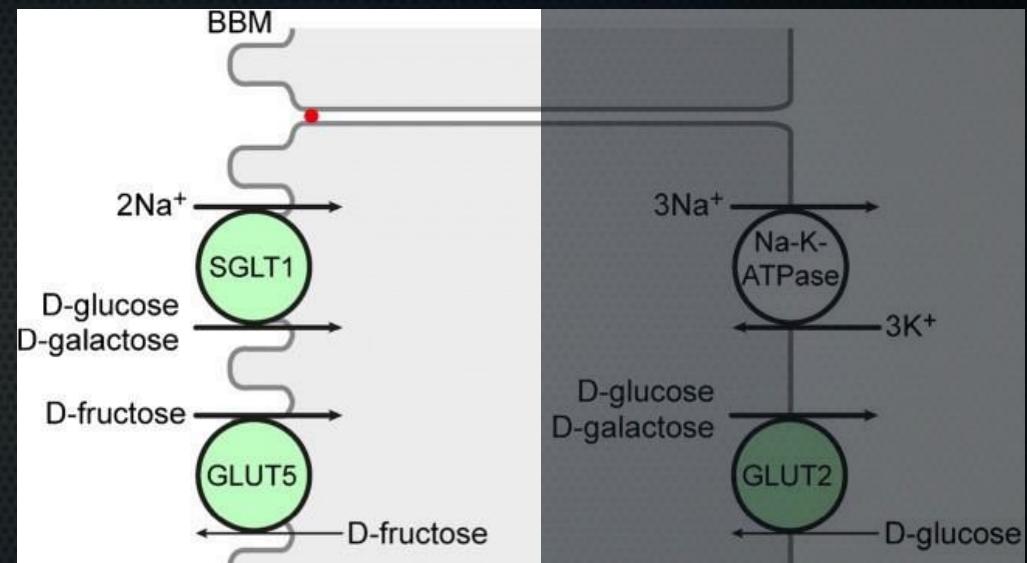


Supervised learning

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

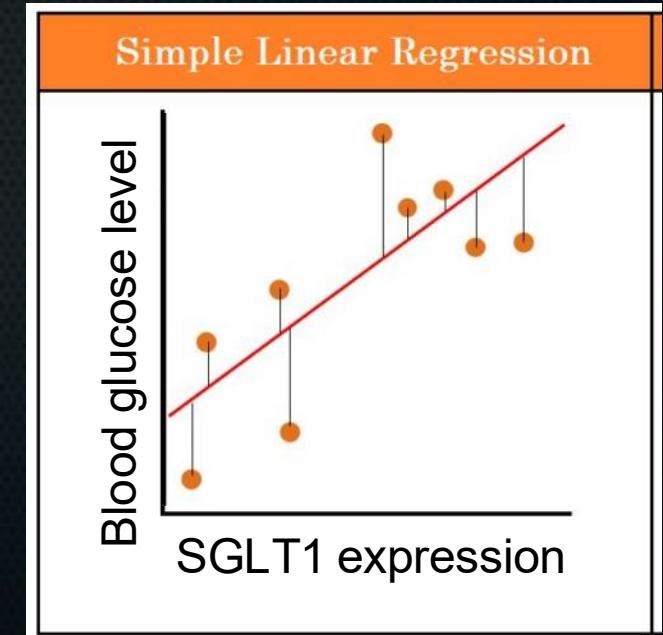
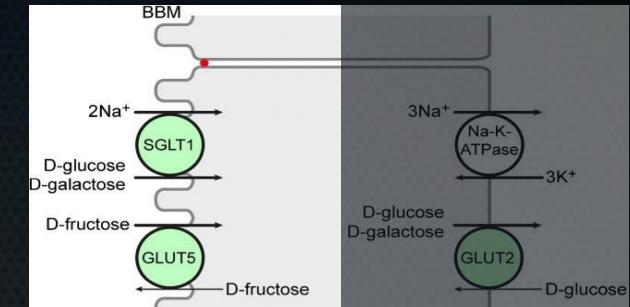
Supervised learning: regression

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



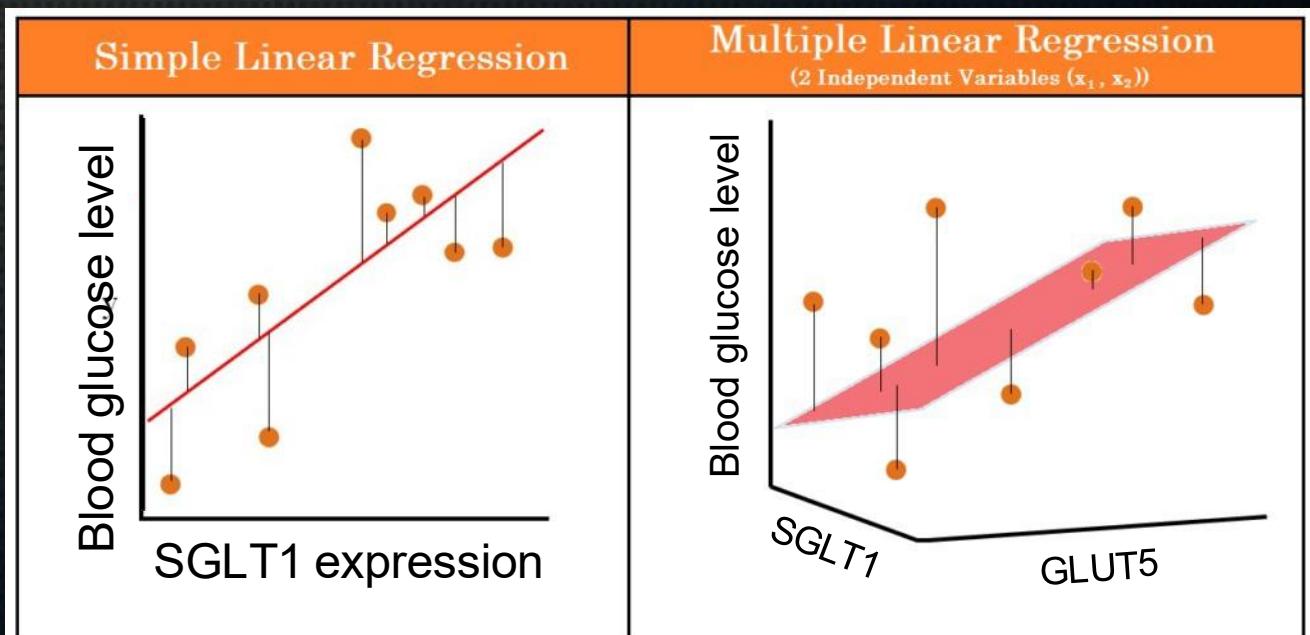
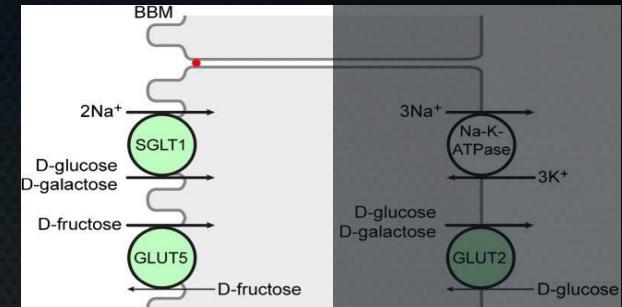
Supervised learning: regression

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



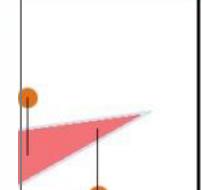
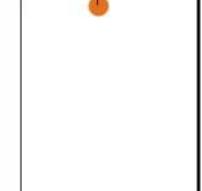
Supervised learning: regression

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



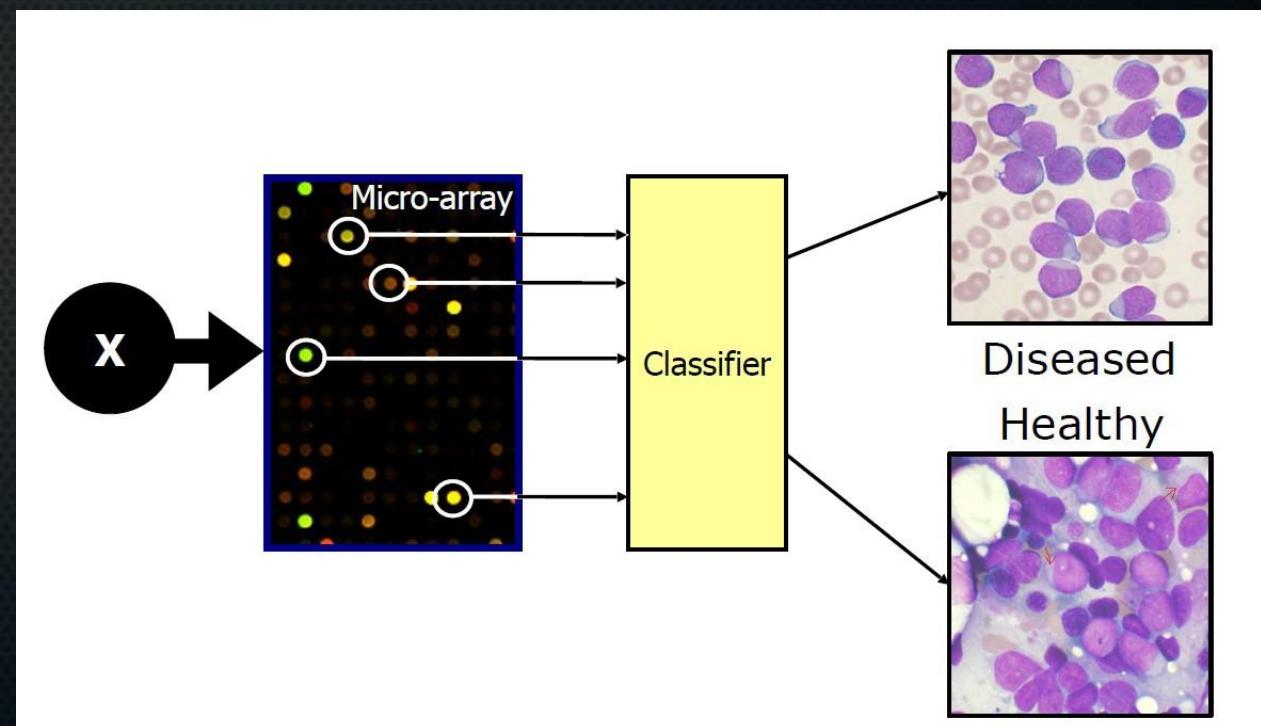
Supervised learning: regression

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

	Simple Linear Regression	Multiple Linear Regression (2 Independent Variables (x_1, x_2))
Simple Linear Regression	$y = b_0 + b_1 x_1$	
Multiple Linear Regression	$y = b_0 + b_1 x_1 + b_2 x_2 + \dots + b_n x_n$	
Polynomial Linear Regression	$y = b_0 + b_1 x_1 + b_2 x_1^2 + \dots + b_n x_1^n$	

Supervised learning: classification

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



Source: Jeroen de Ridder

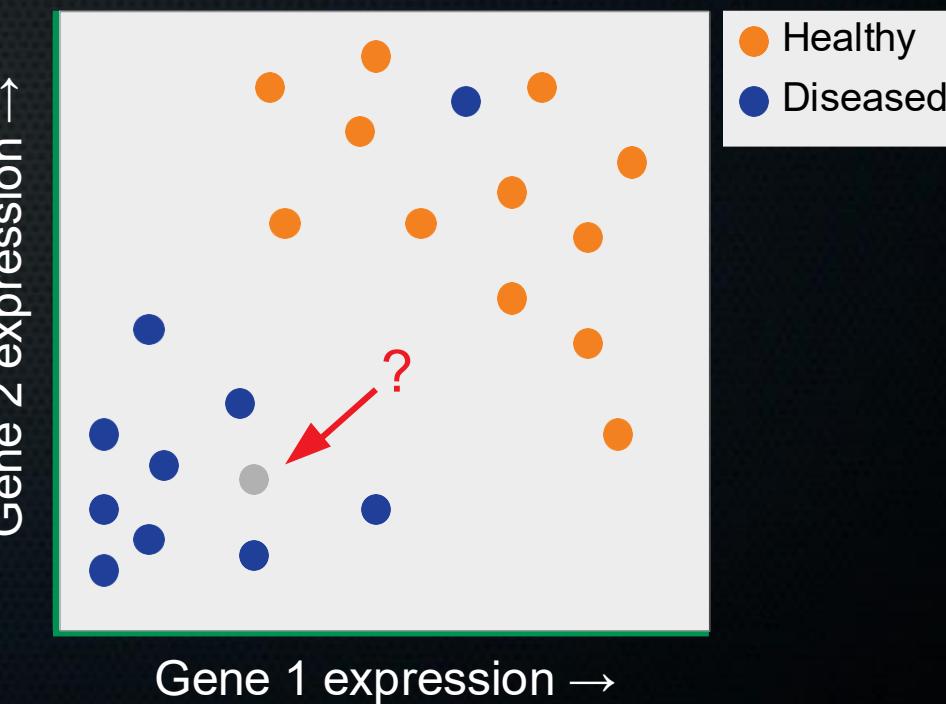
Supervised learning: classification

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



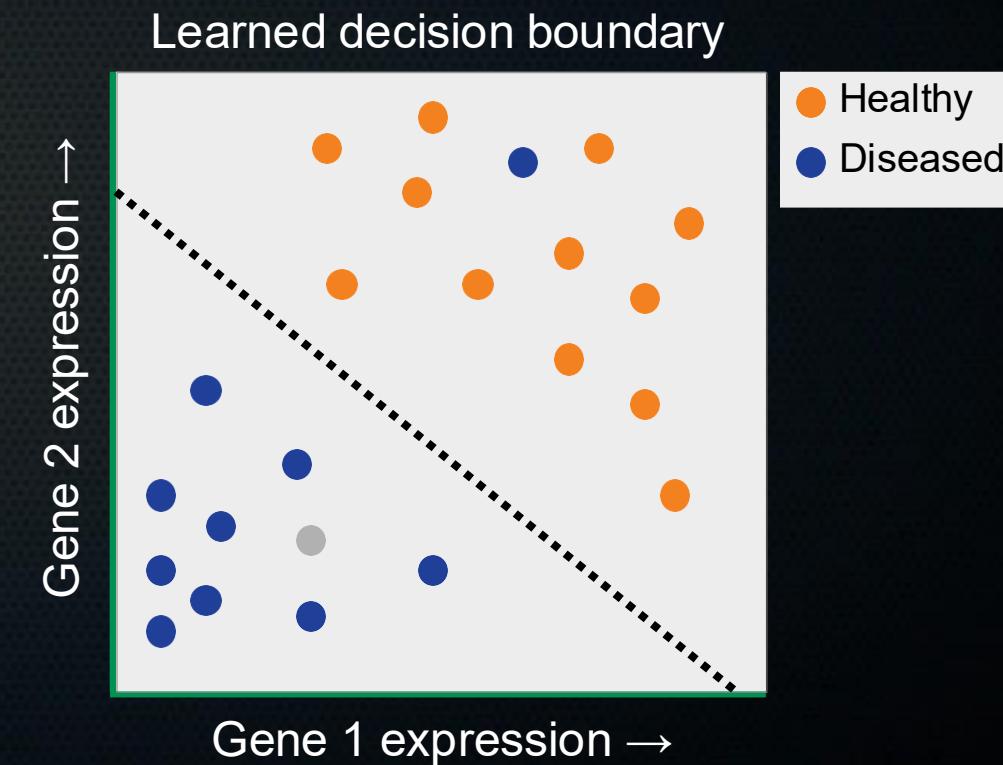
Supervised learning: classification

- 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.

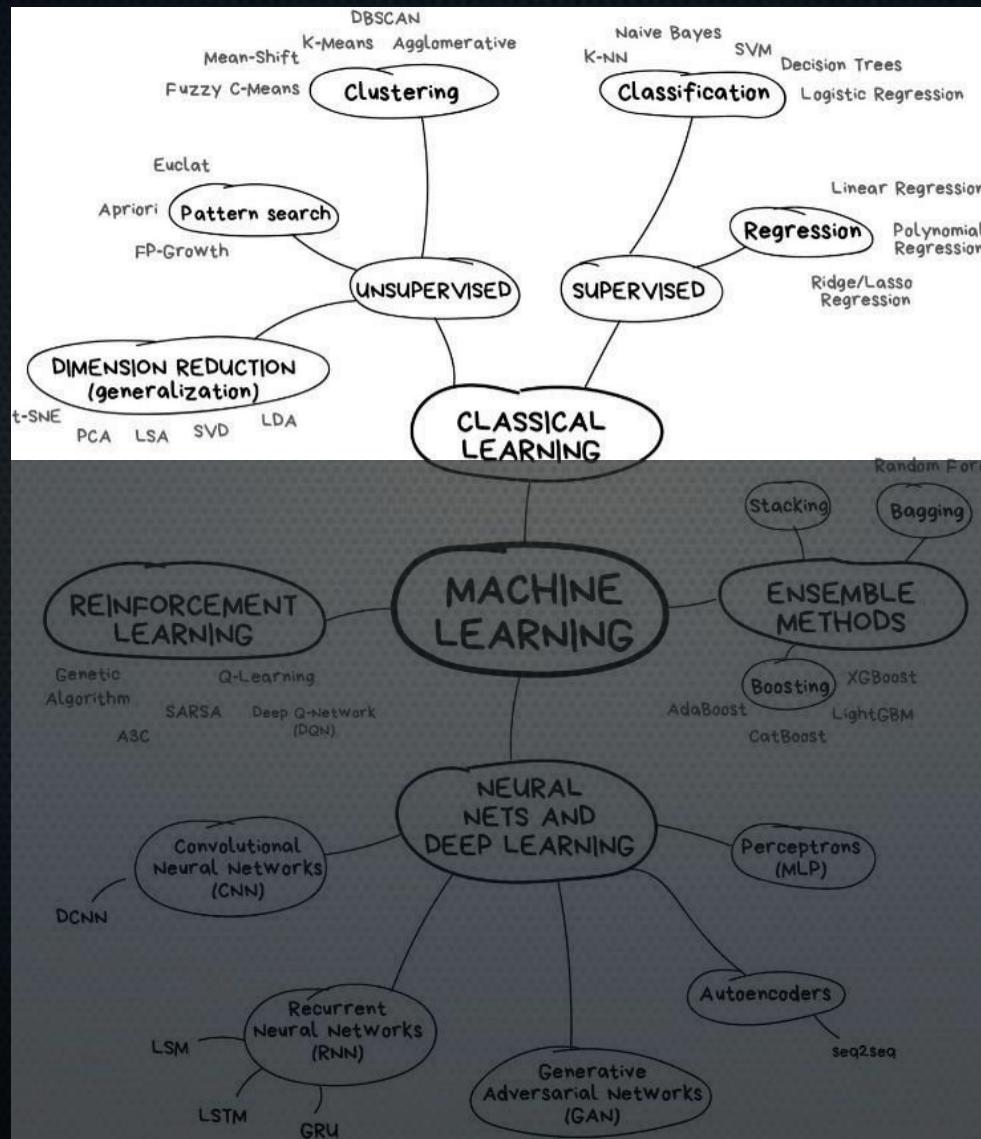


Supervised learning: classification

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



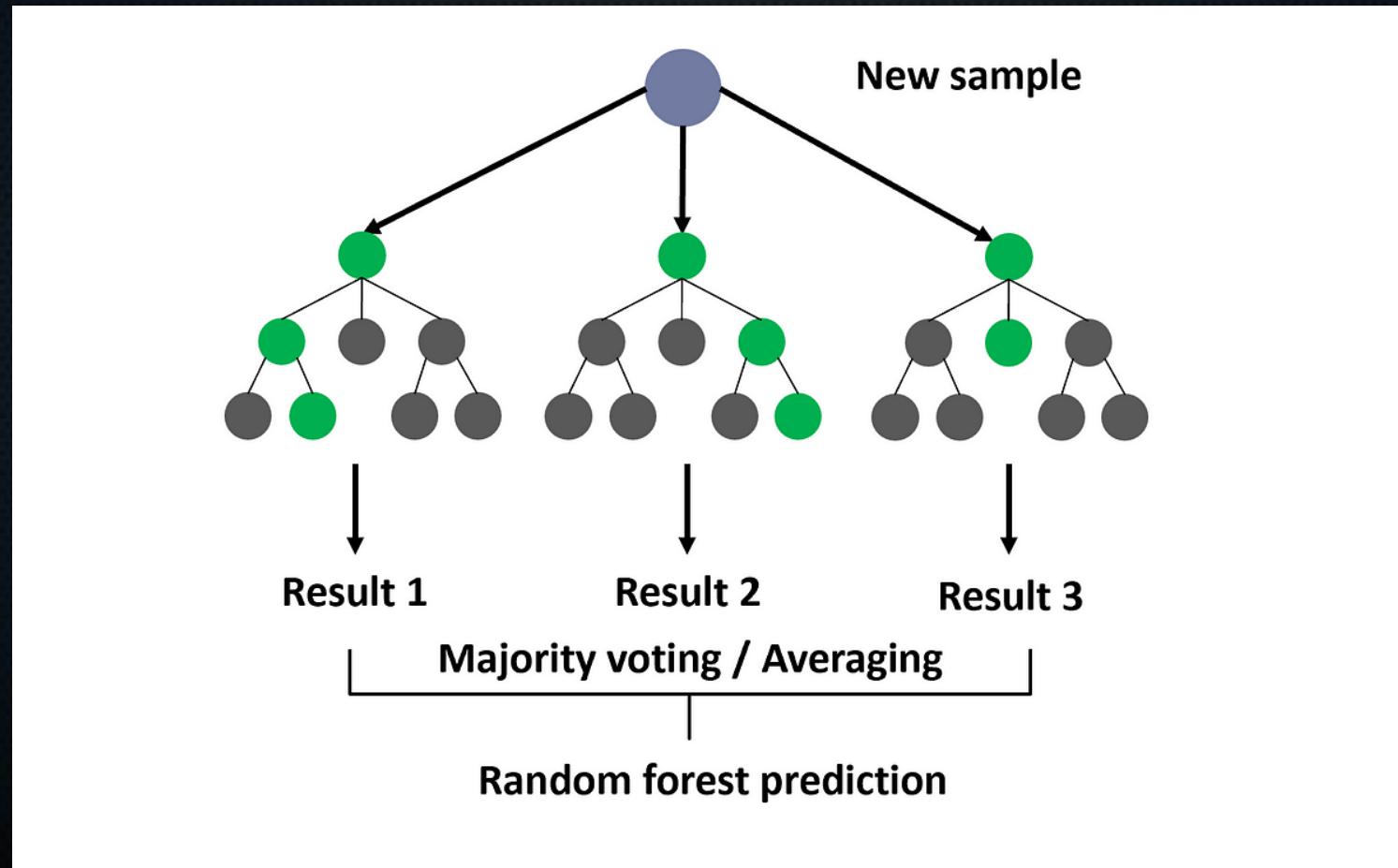
Map of Machine Learning



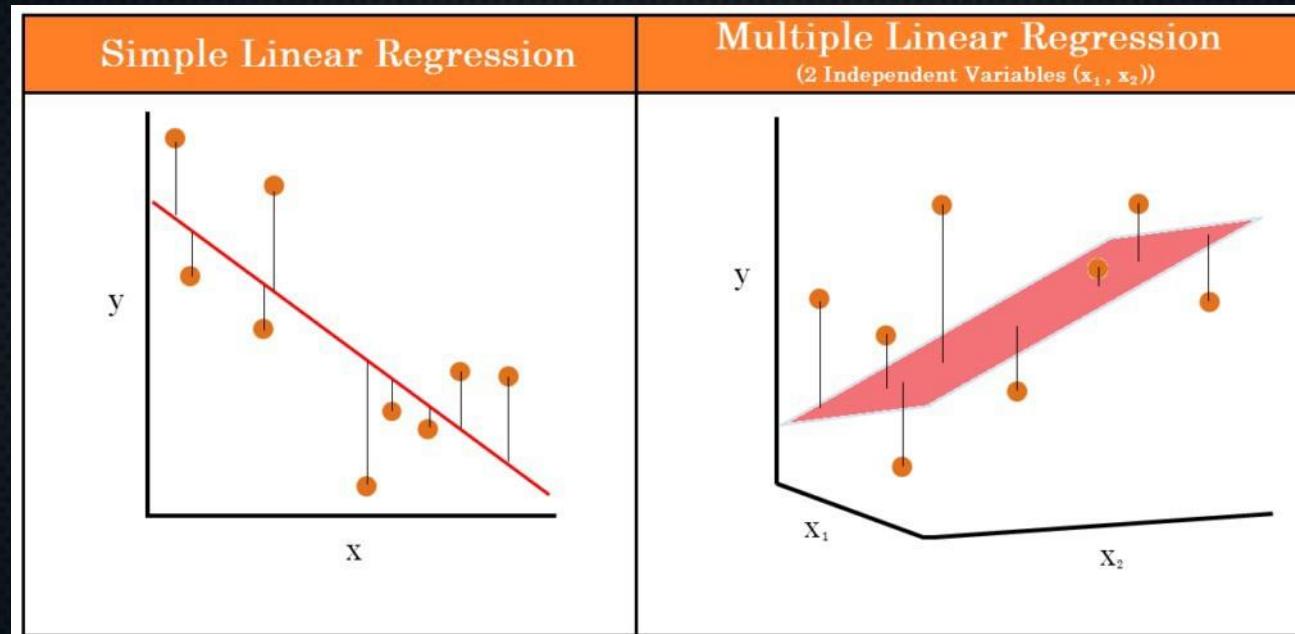
Summary

- 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.

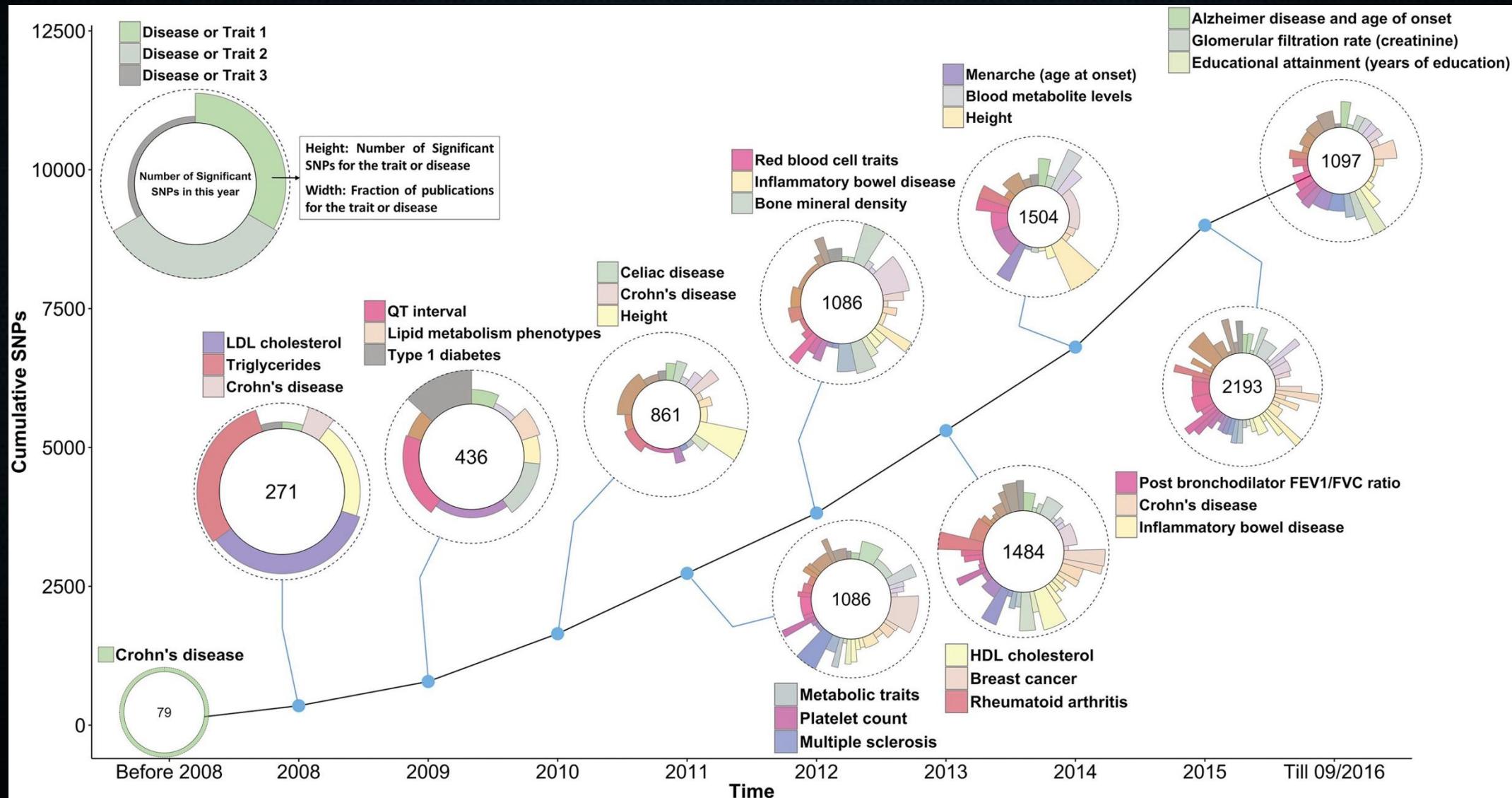
Regression or classification?



Linear regression

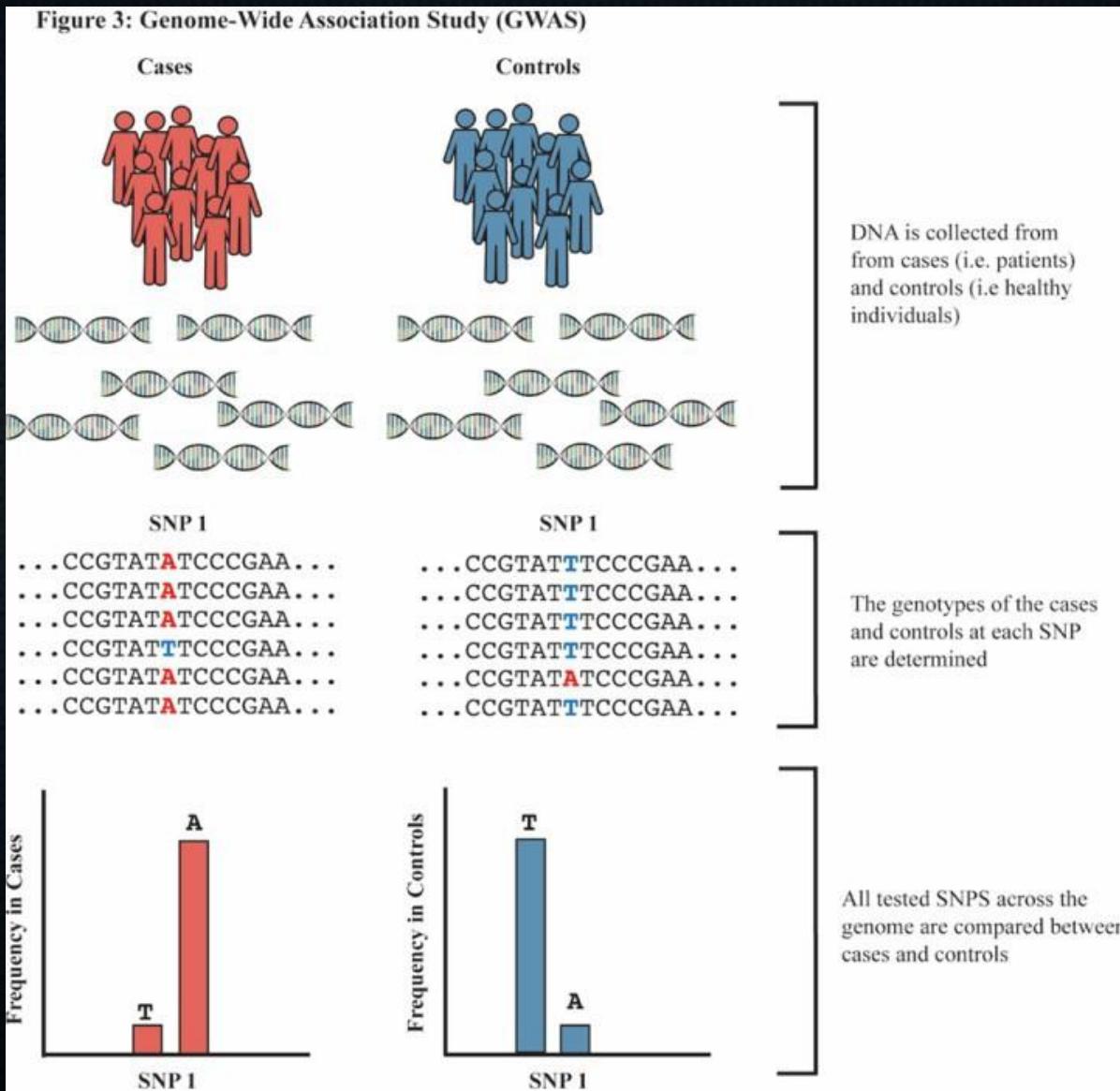


Linear regression: GWAS

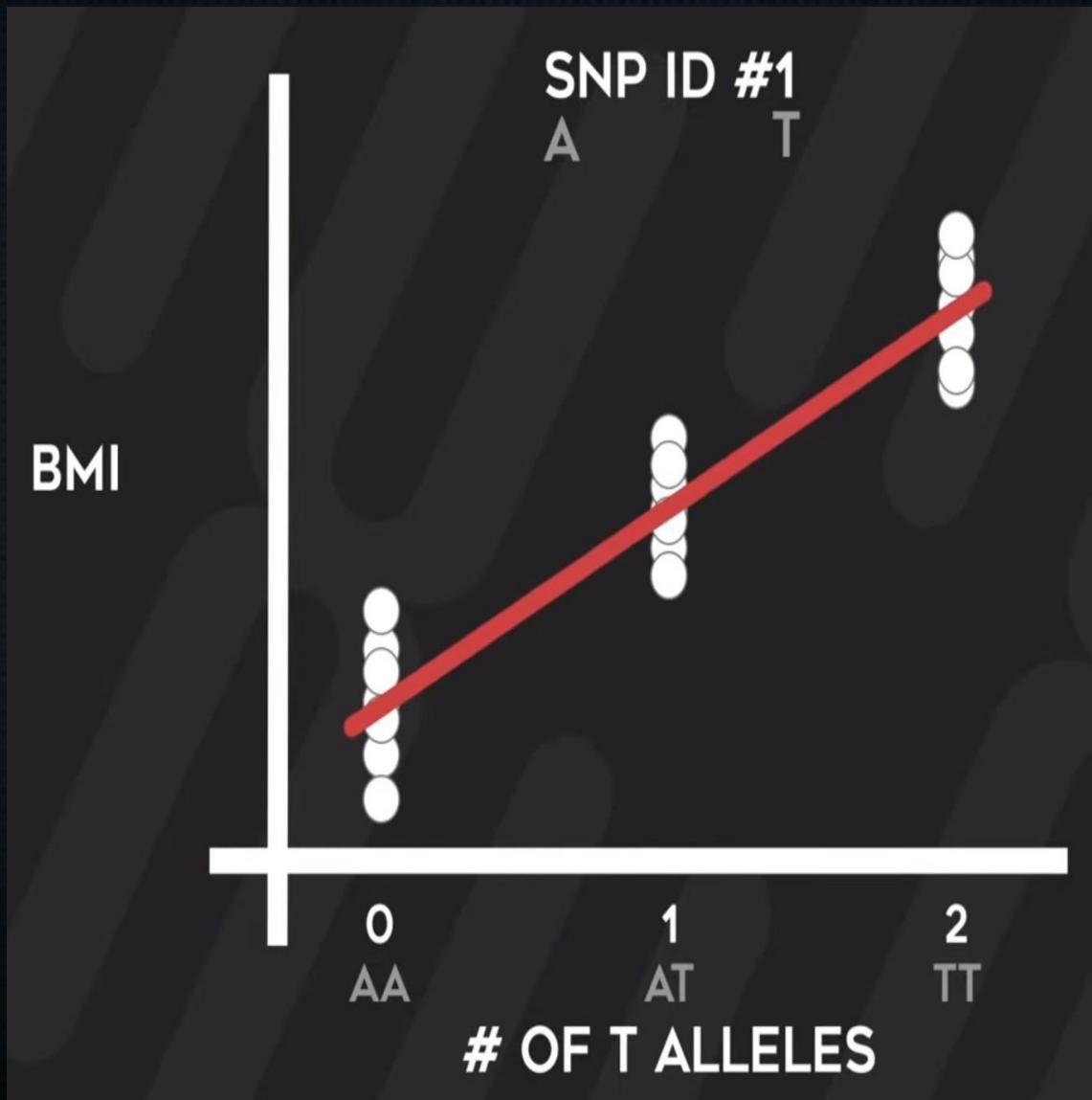


Linear regression: GWAS

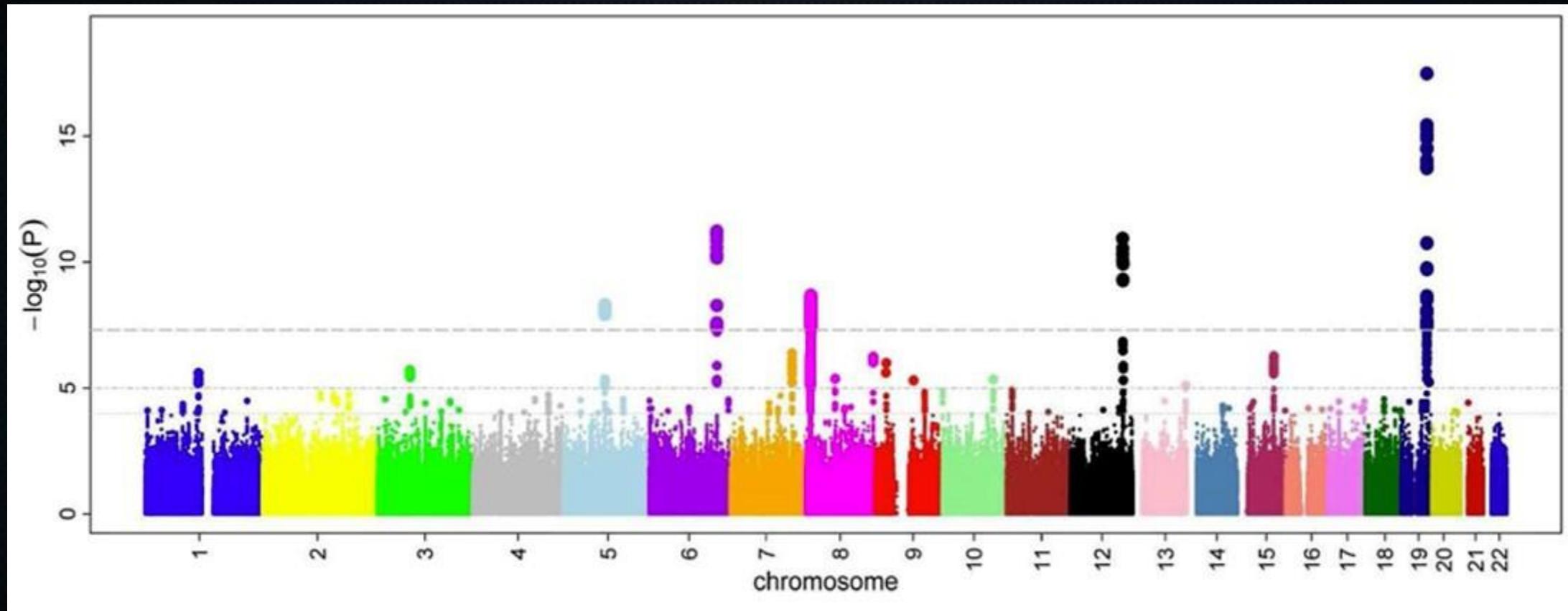
Figure 3: Genome-Wide Association Study (GWAS)



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.

AJHG

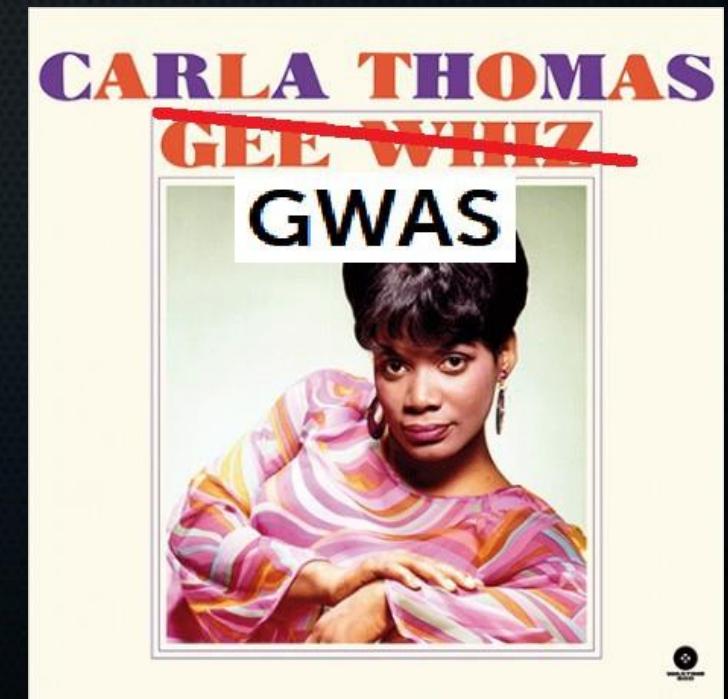
Volume 110, Issue 2, 2 February 2023, Pages 179-194

Review

15 years of GWAS discovery: Realizing the promise

Abdel Abdellaoui ¹   , Loic Yengo ² , Karin J.H. Verweij ¹ , Peter M. Visscher ²

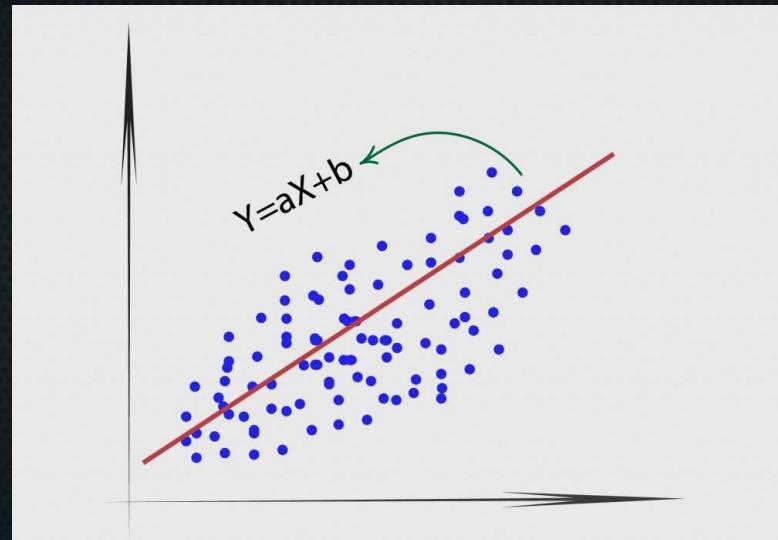




Univariate linear regression

$$y=ax+b$$

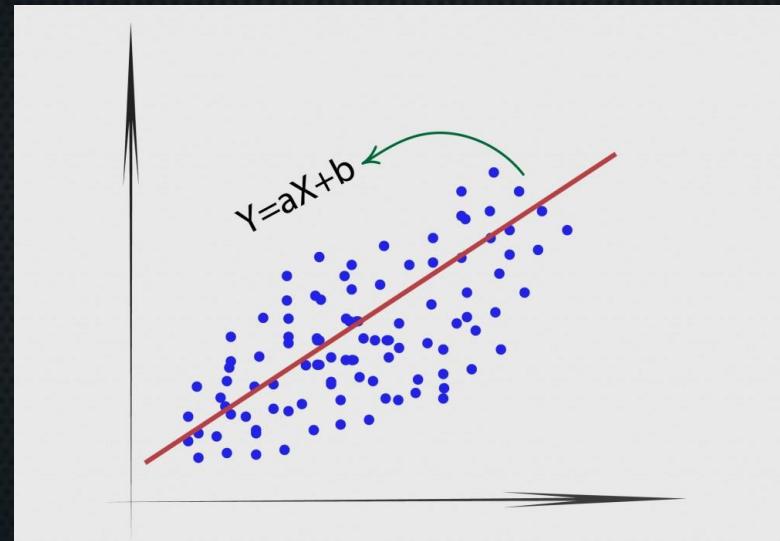
$$y=ax+b$$



Univariate linear regression

$$y=ax+b$$

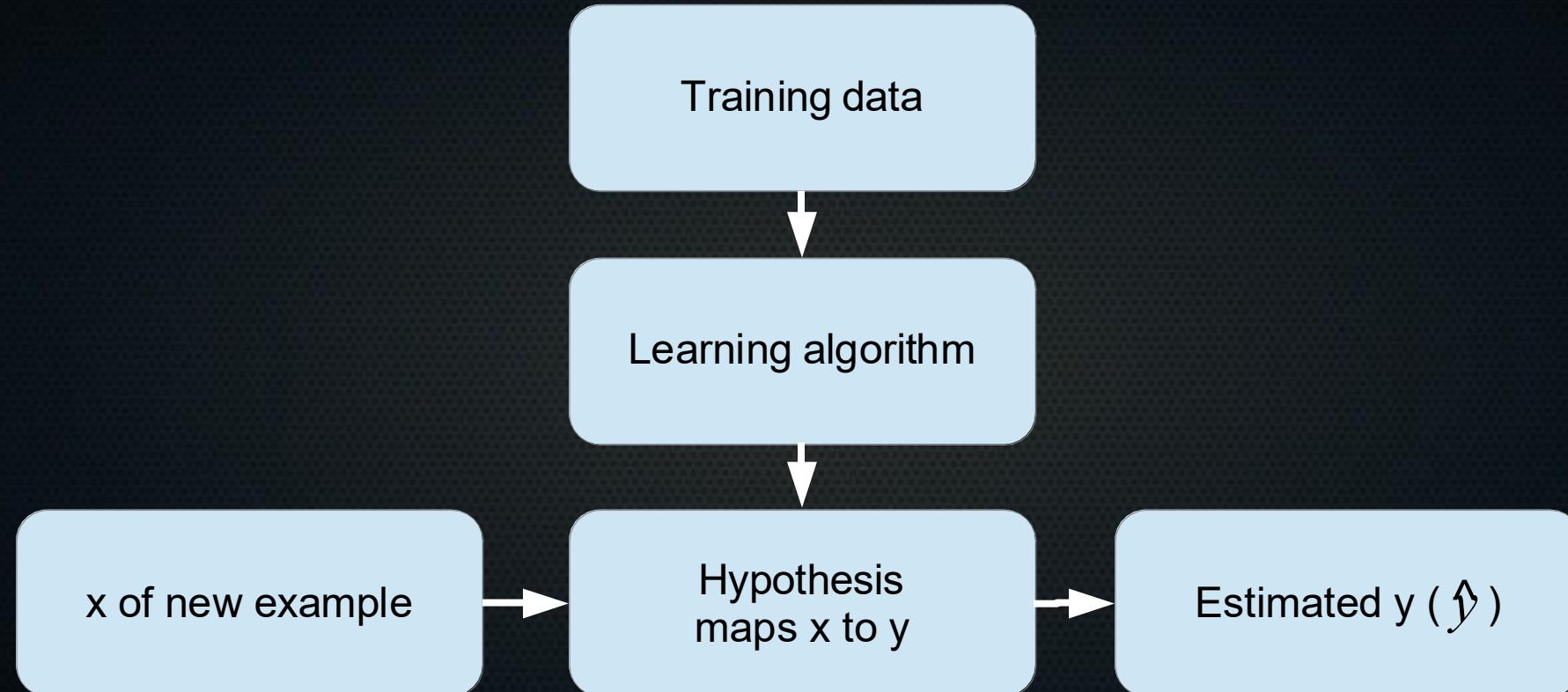
$$y=ax+b$$



$$y=\theta_0+\theta_1x$$

Process

$$y = ax + b$$



Source: Andrew Ng, Coursera

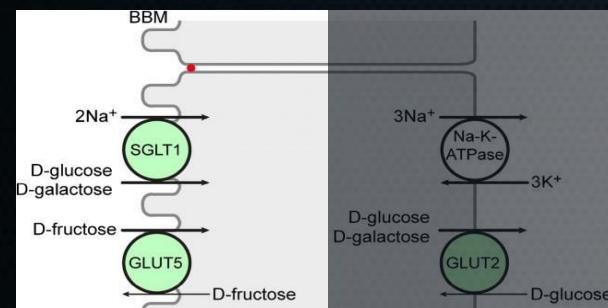
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

Terminology

alert! #StatQuest BAM

$$y=ax+b$$

Training data

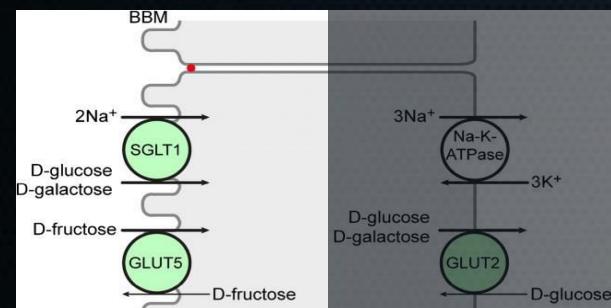


Sample #	SGLT1 expression level (arbitrary units relative to housekeeping gene)	Blood glucose level (mg/dL)
1	3	80
2	8	130
3	12	170
4	2	89

Terminology

$$y = ax + b$$

Training data



$m = \# \text{ of training examples}$

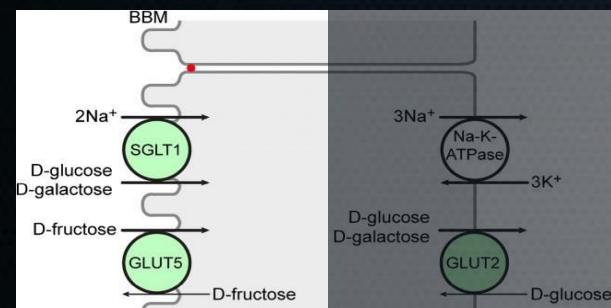
m

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

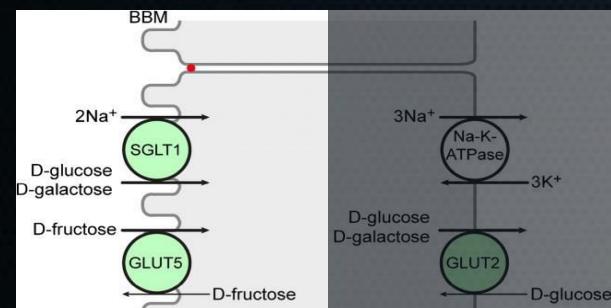
m

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

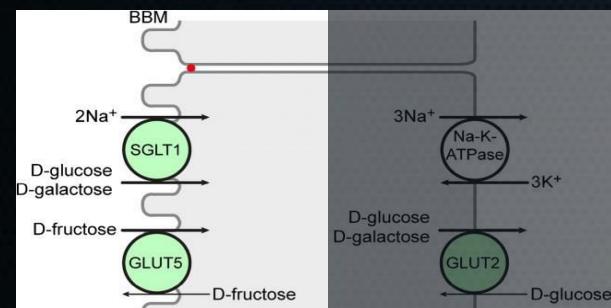
m

n

Terminology

$$y = ax + b$$

Training data



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

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

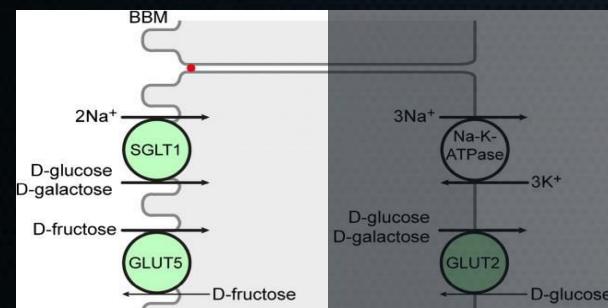
n

y

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

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

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

n

y

A green arrow points from the text $(x^{(3)}, y^{(3)})$ to the third row of the table. The values 12 and 170 in the third row are circled in red.

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

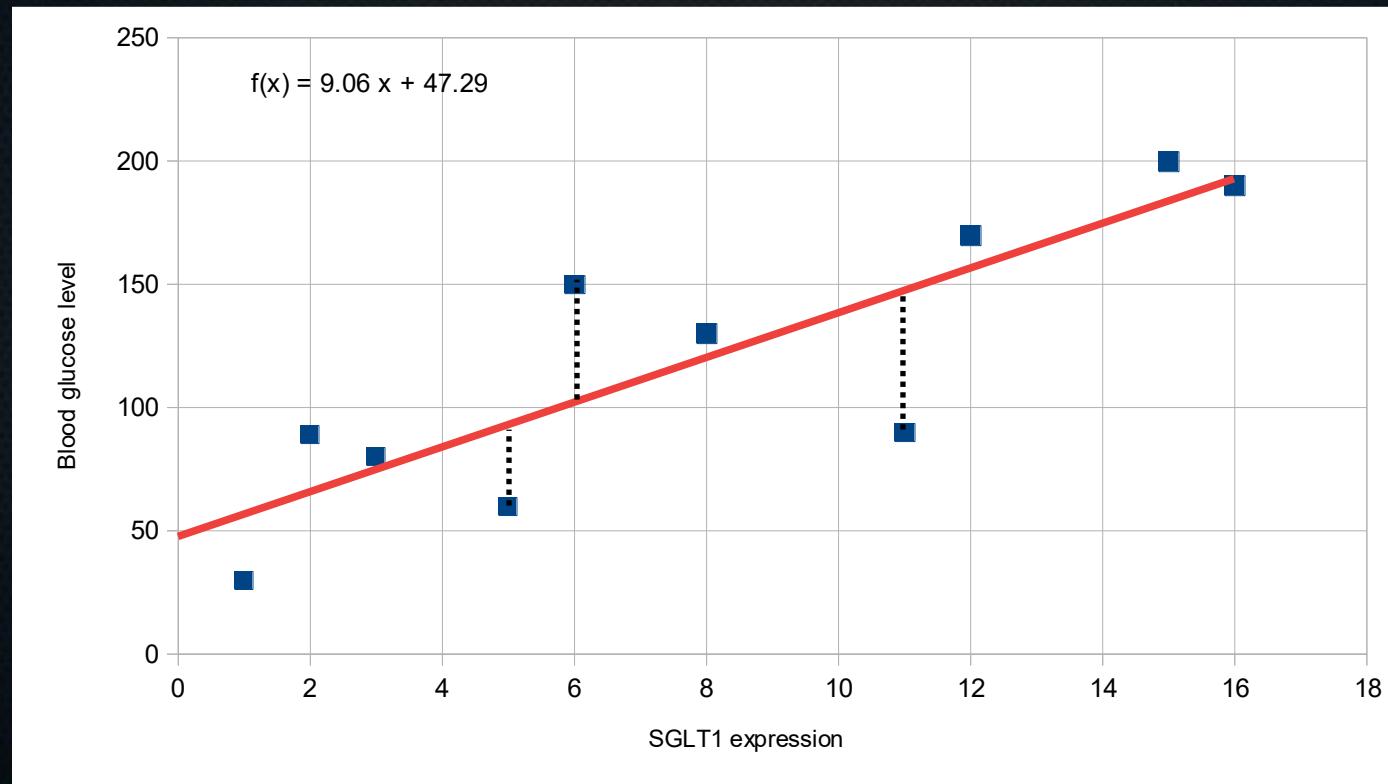
- How to learn theta's from data?
- Two parts
 - **How wrong are we for given parameters?**
 - How do we update our parameters, given how wrong we are?

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

Cost function

- How wrong are we for given parameters?

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

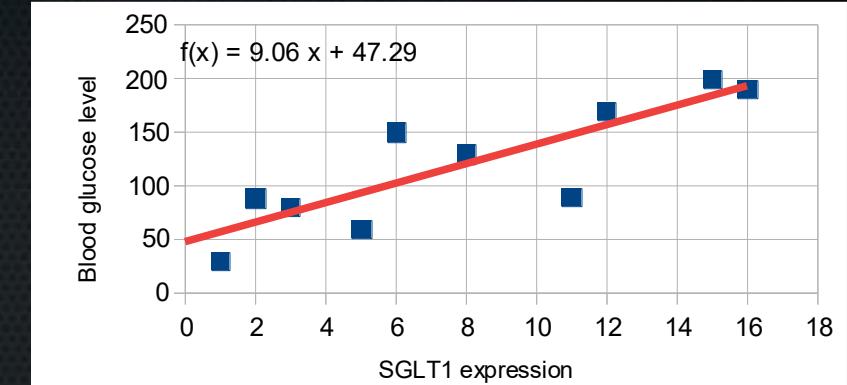


Cost function

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

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

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



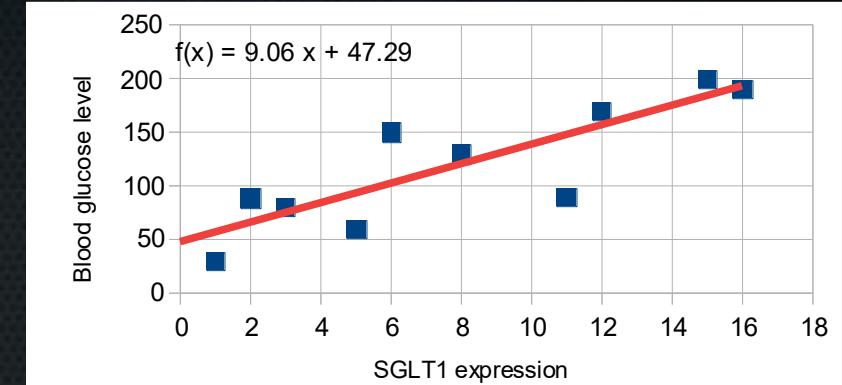
Cost function

- 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

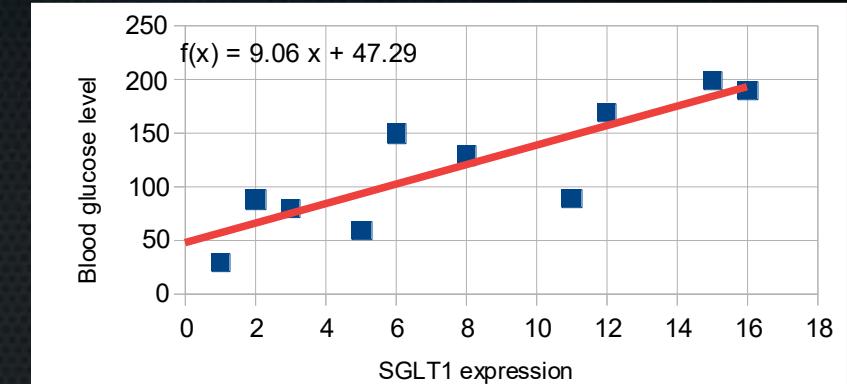
- How wrong are we for given parameters?
- Cost function:

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

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

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

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



Cost function

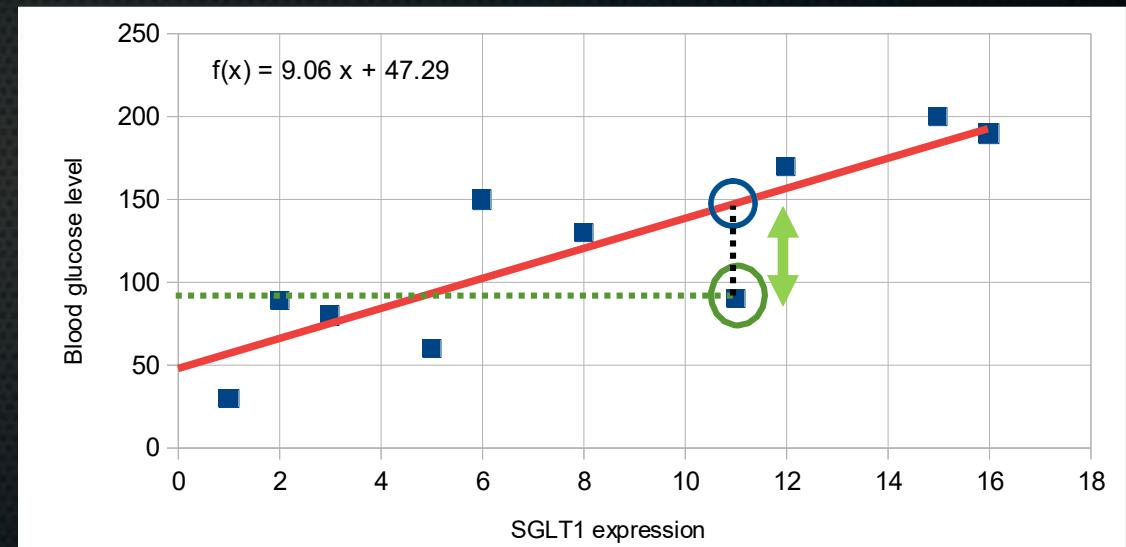
- How wrong are we for given parameters?
- Cost function:

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

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

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

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



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:

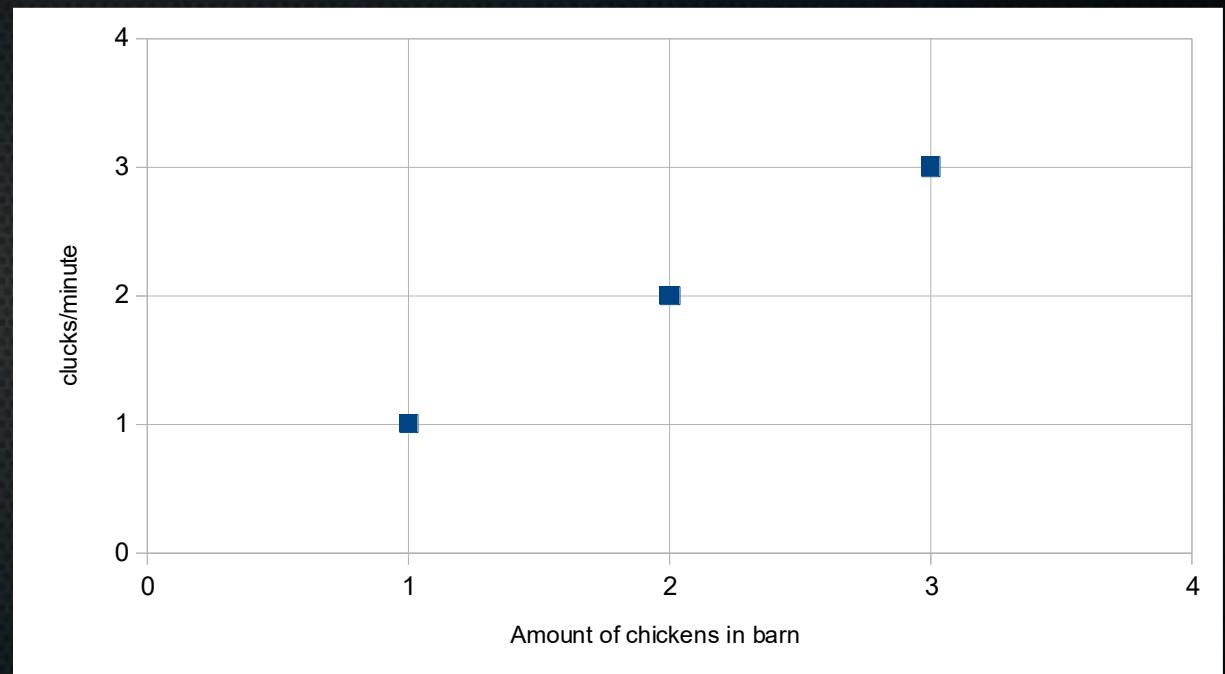
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (\theta_0 + \theta_1 \cdot x^{(i)} - y^{(i)})^2$$

- 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?

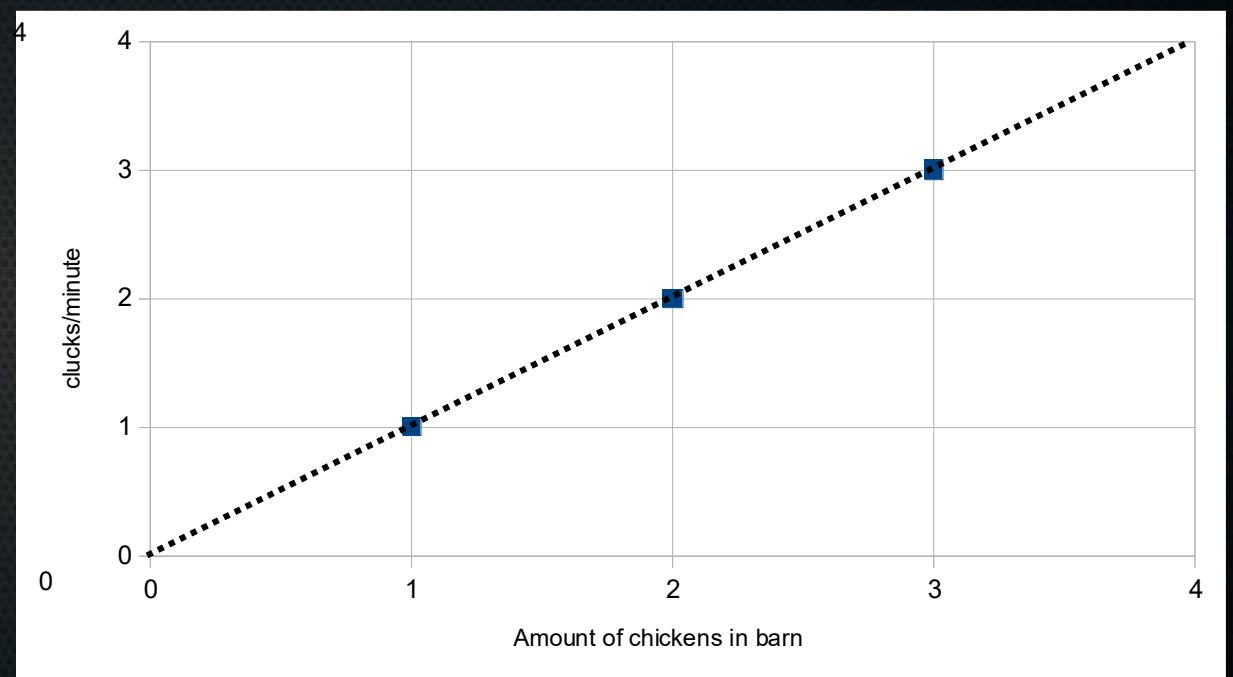


Cost function illustration

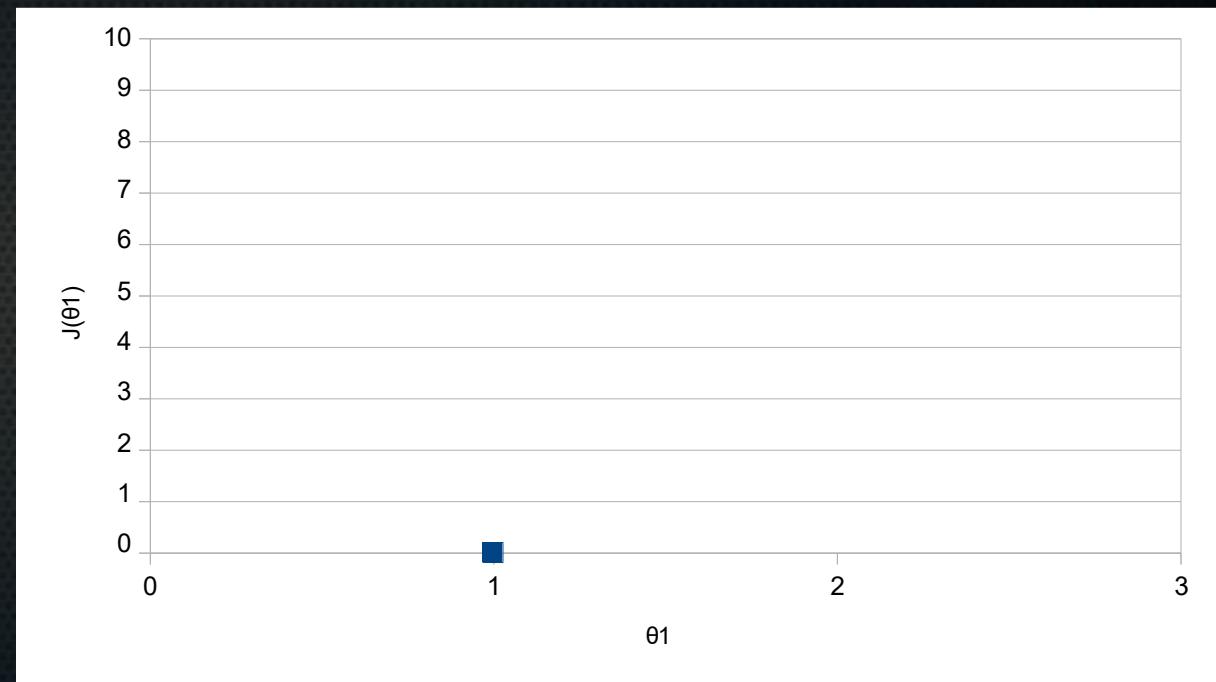
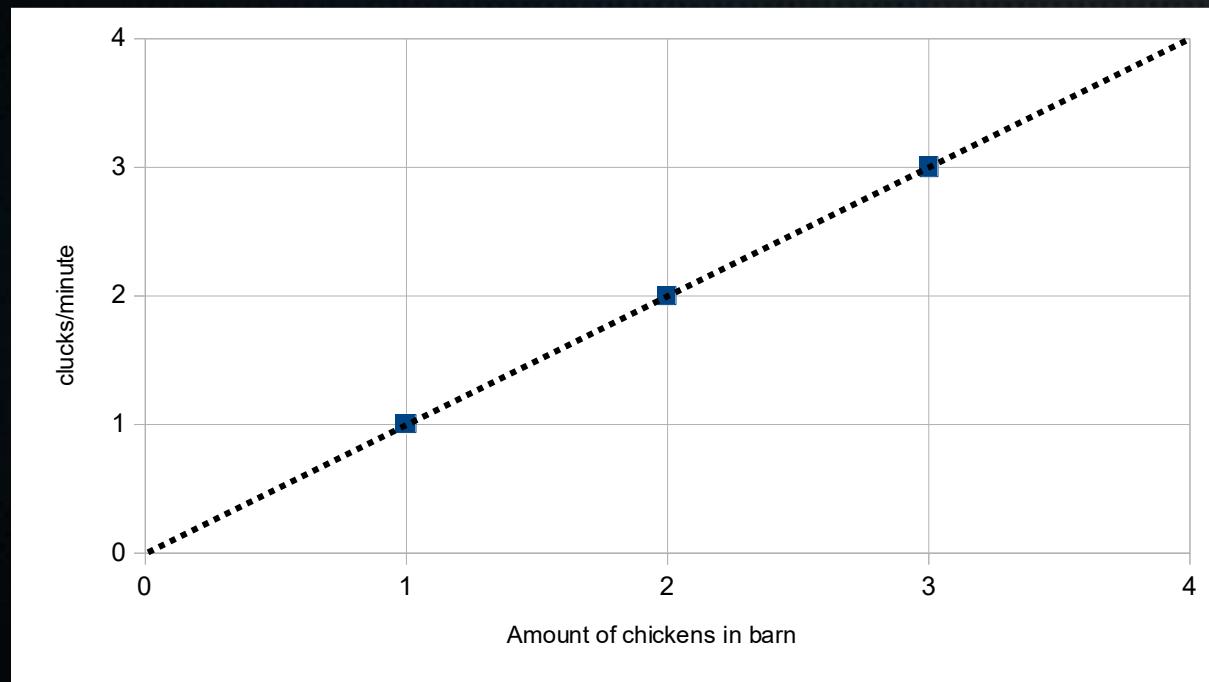
- Let's say theta₀ = 0 (so the intercept is 0). What is J(theta₁) for different values of theta₁?
- Theta₁ = 1

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

$$J(\theta_1) = 0$$



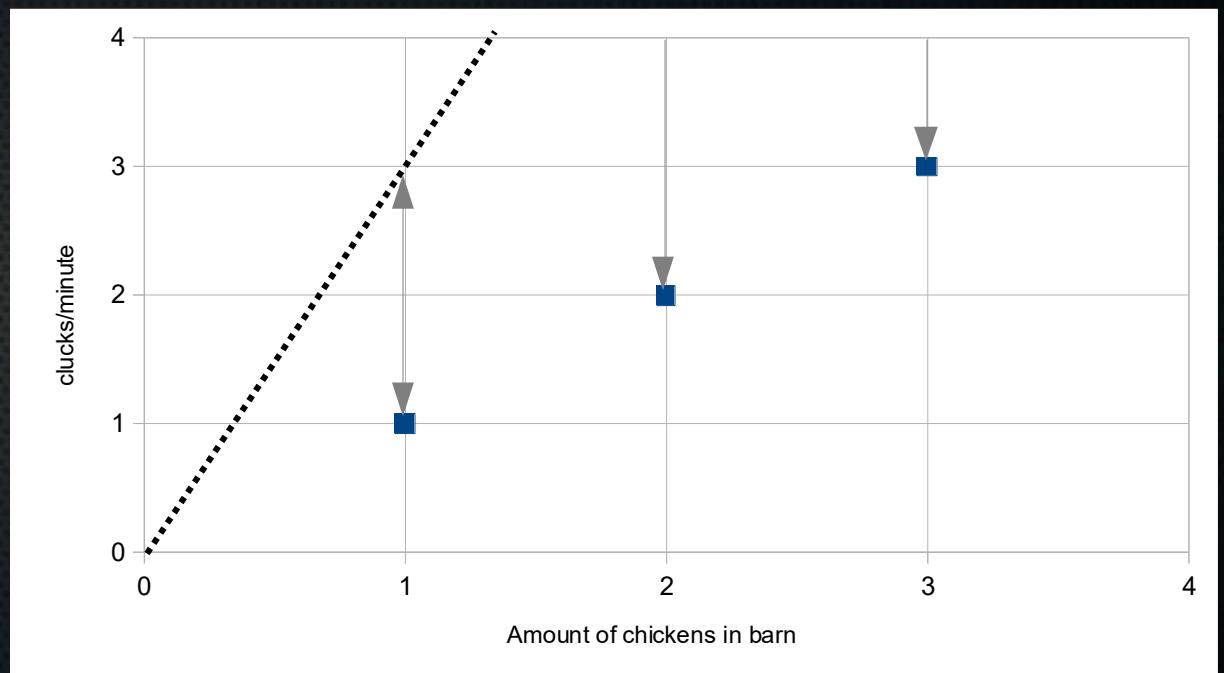
Cost function illustration



Cost function illustration

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

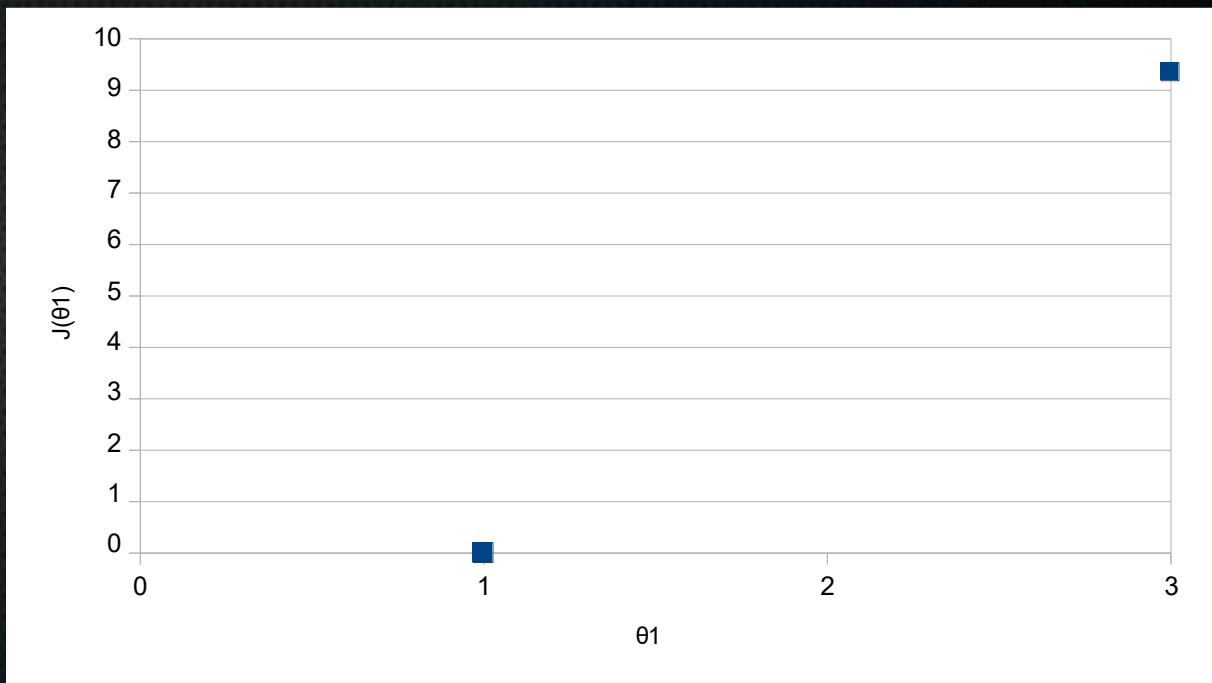
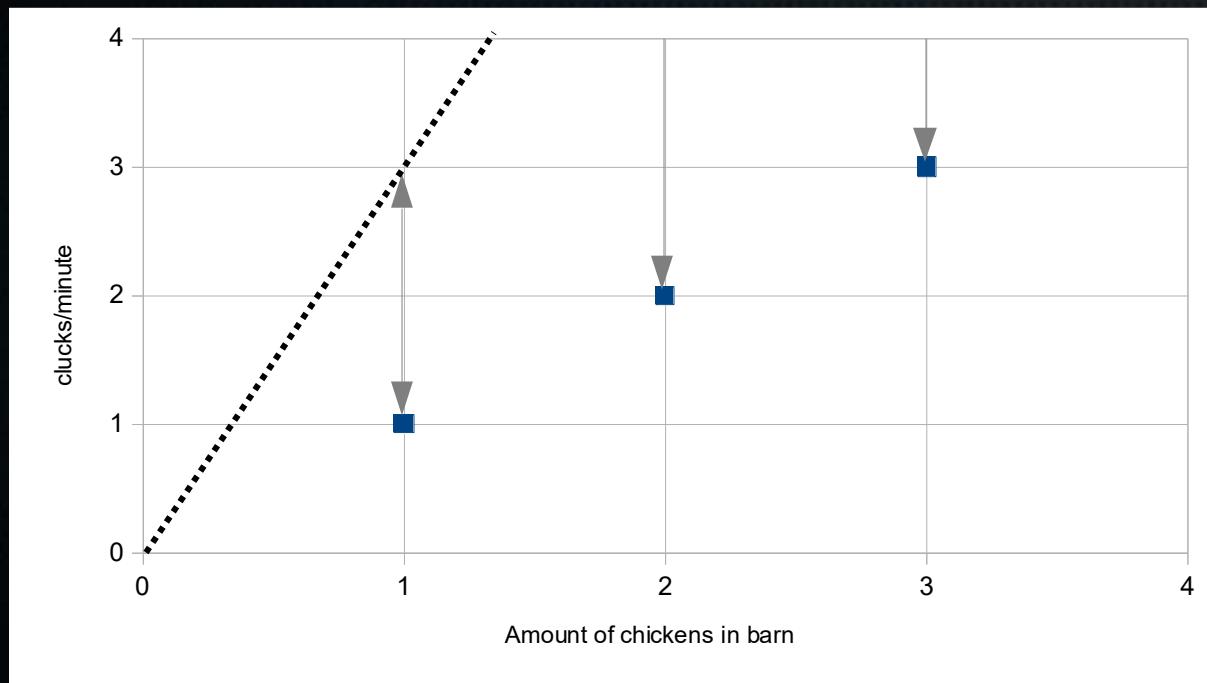
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^3 (\theta_1 \cdot x^{(i)} - y^{(i)})^2$$



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



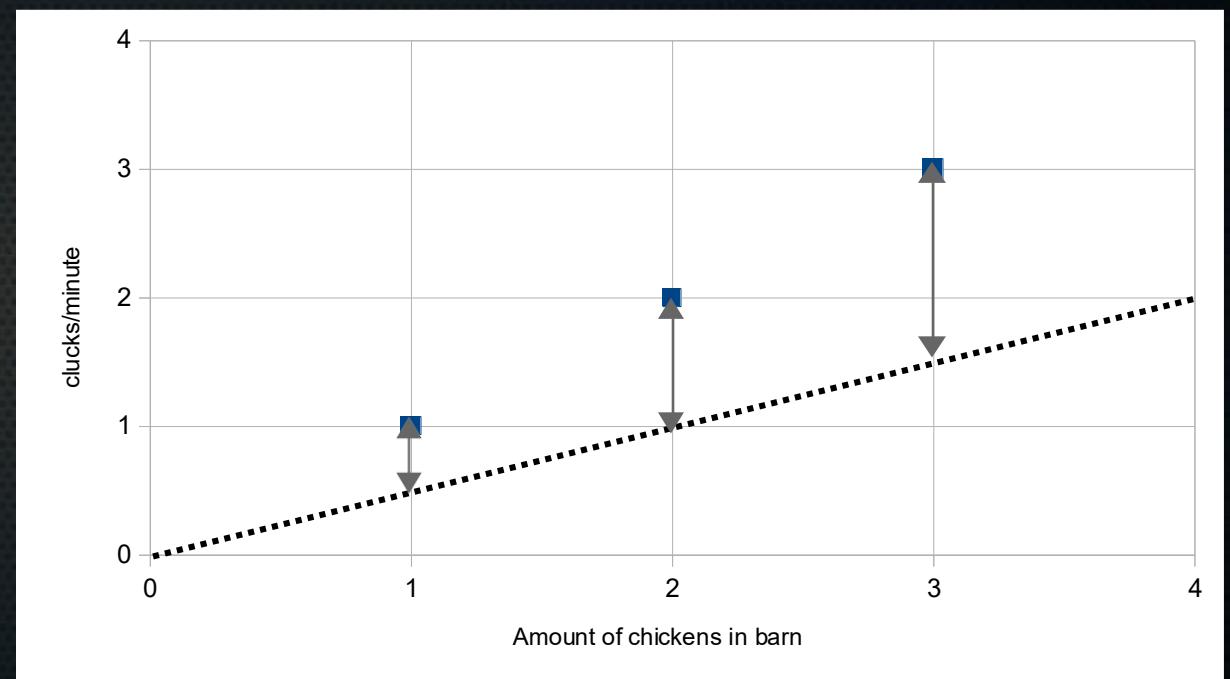
Cost function illustration



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

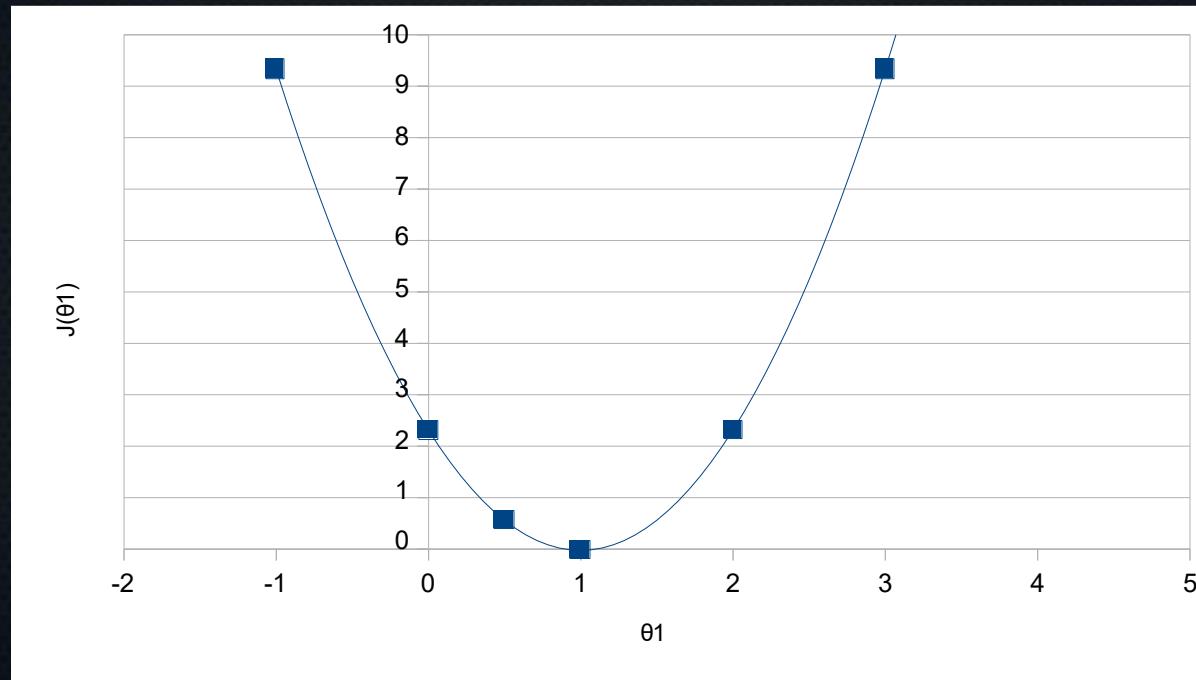
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^3 (\theta_1 \cdot x^{(i)} - y^{(i)})^2$$



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



Cost function illustration

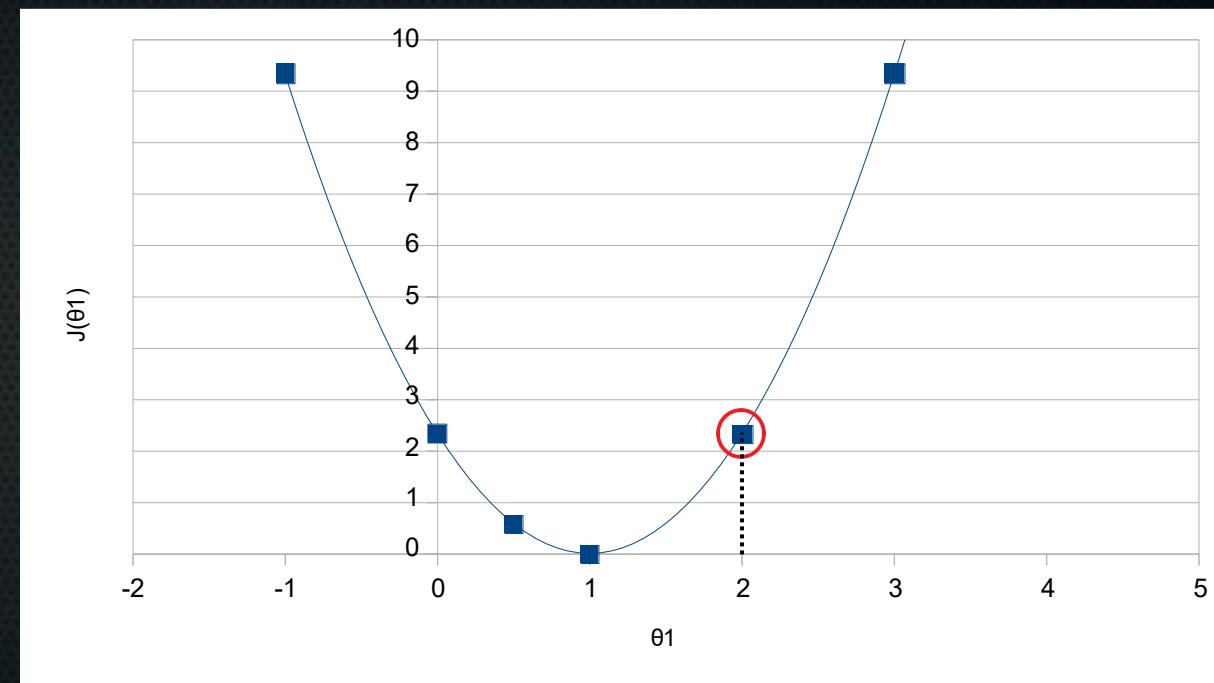


Cost function and gradient descent

- 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?***

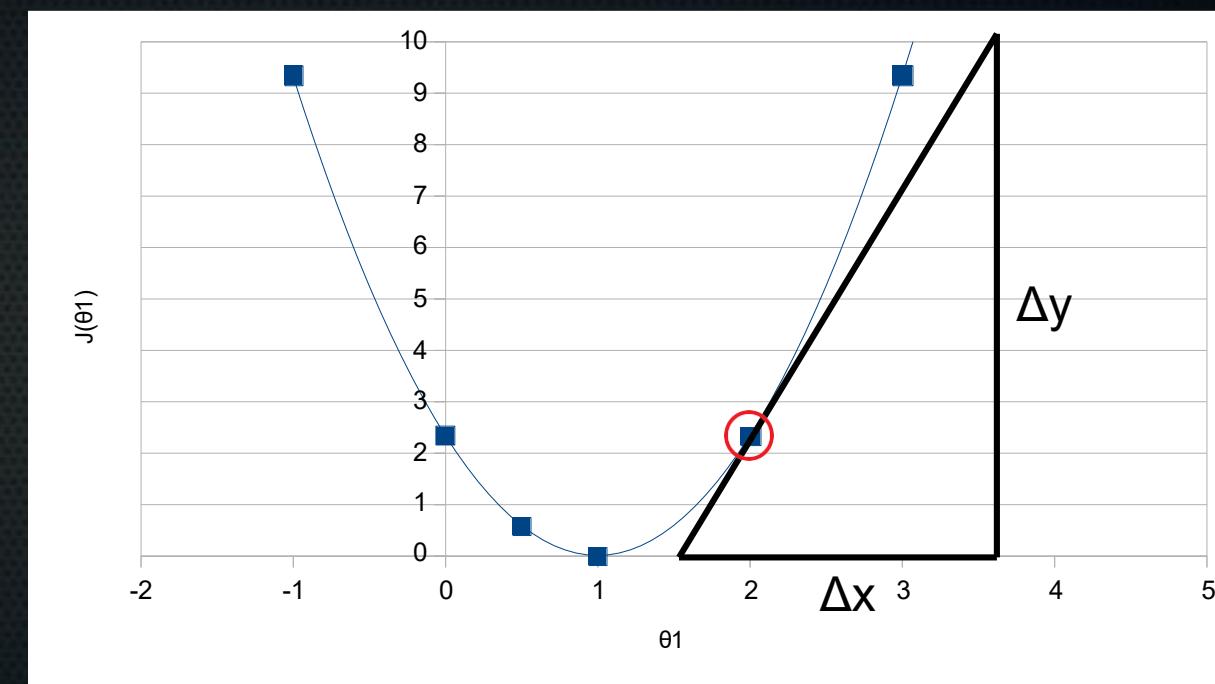
Gradient descent

- Want to minimise
- Where to go?



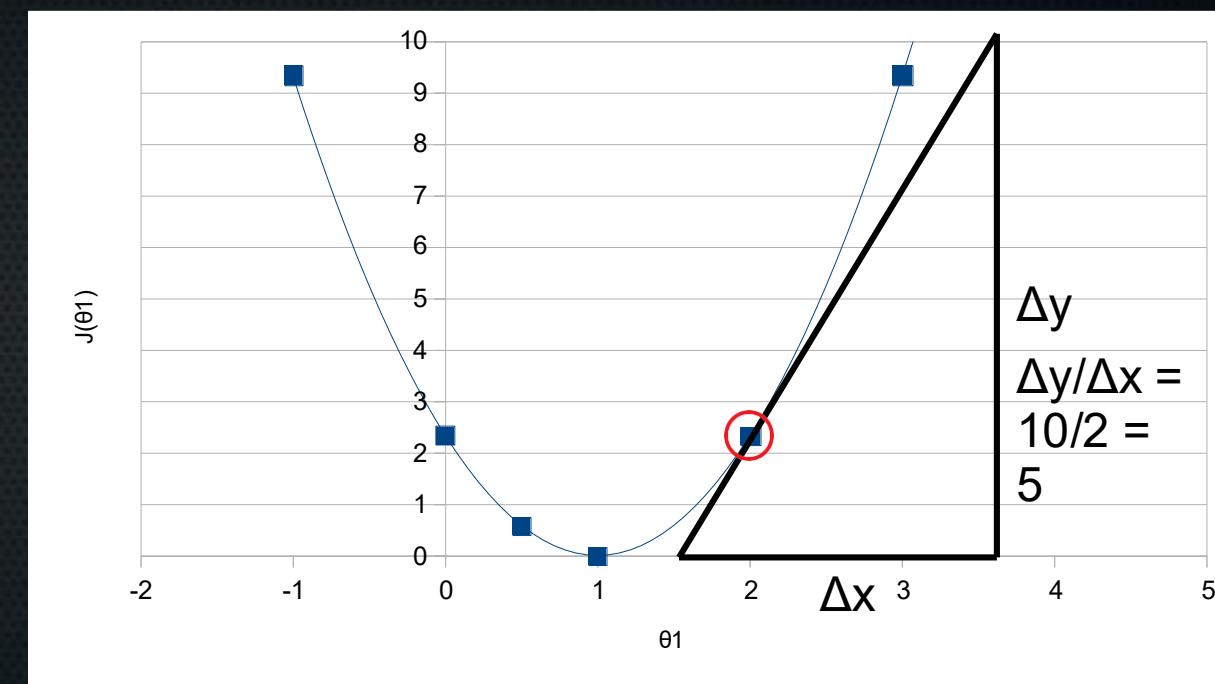
Gradient descent

- Want to minimise
- Where to go?



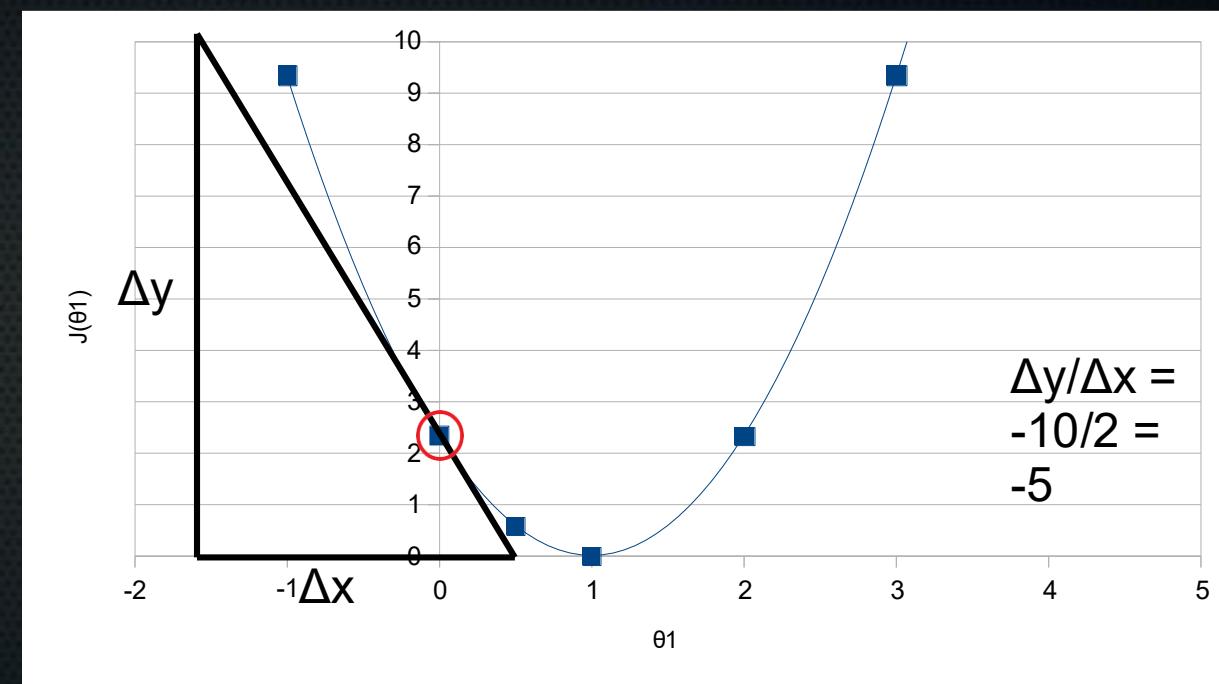
Gradient descent

- Want to minimise
- Where to go?



Gradient descent

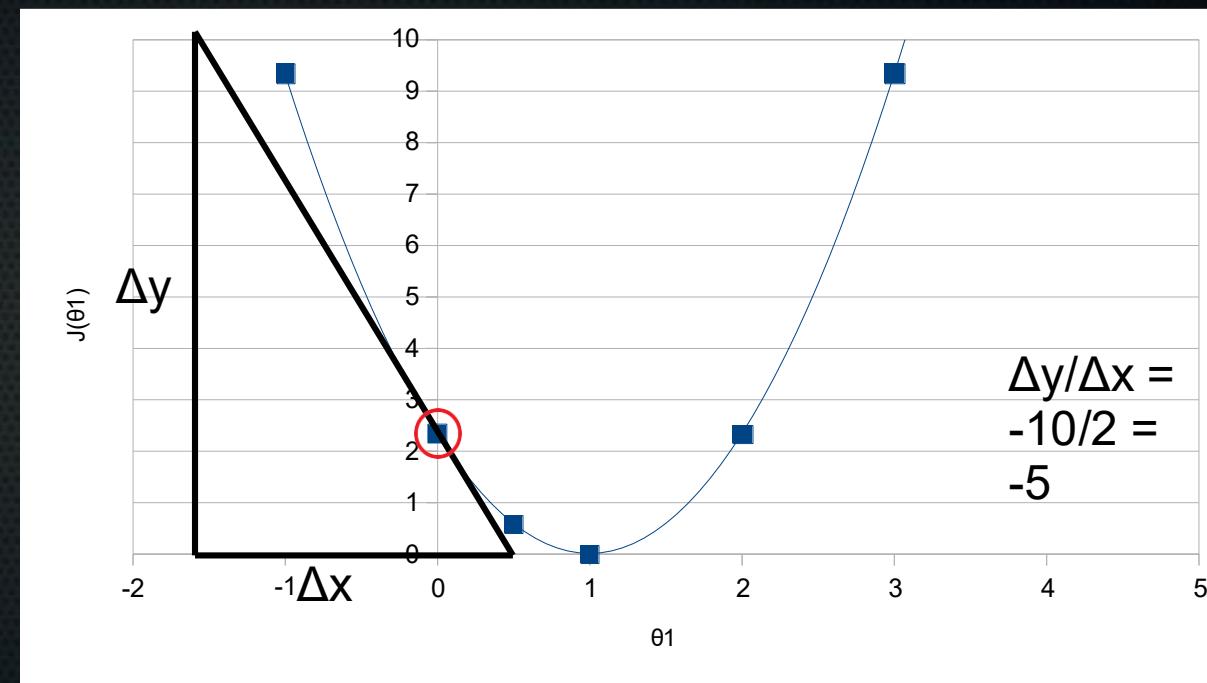
- Want to minimise
- Where to go?



Gradient descent

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

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



Gradient descent

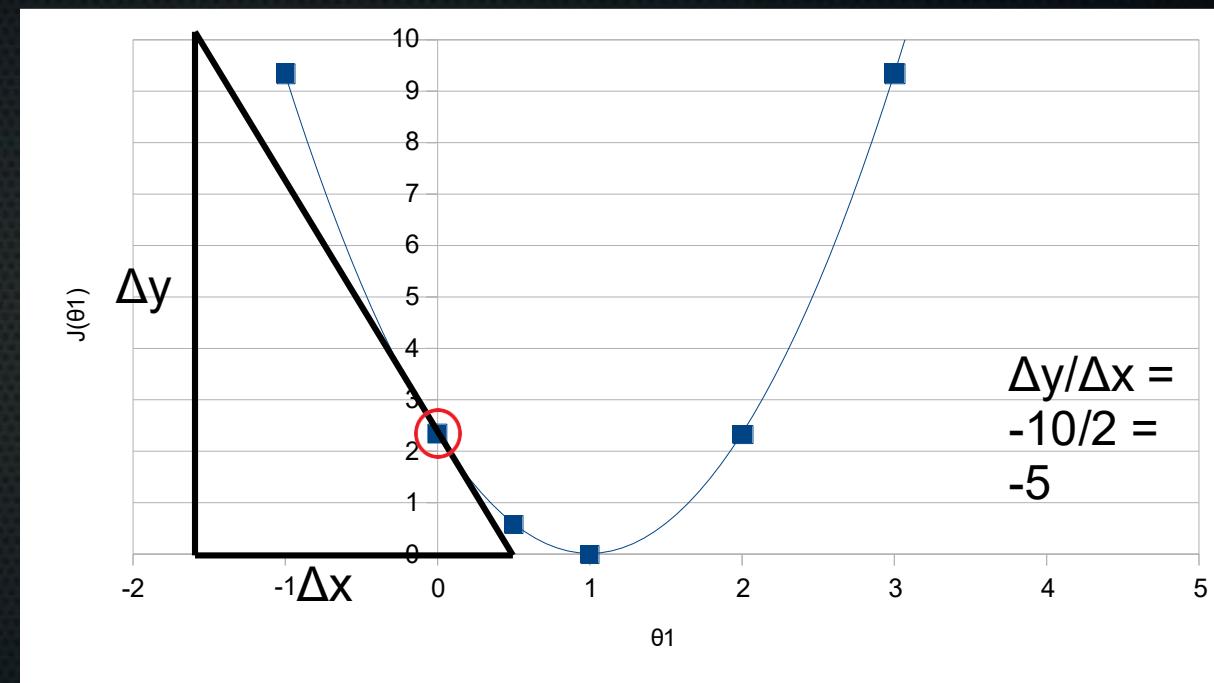
- Want to minimise
- Where to go?
- Change current theta1 as follows:

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

$$a = 0.1$$

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

$$\theta_1 = 0.5$$



Gradient descent

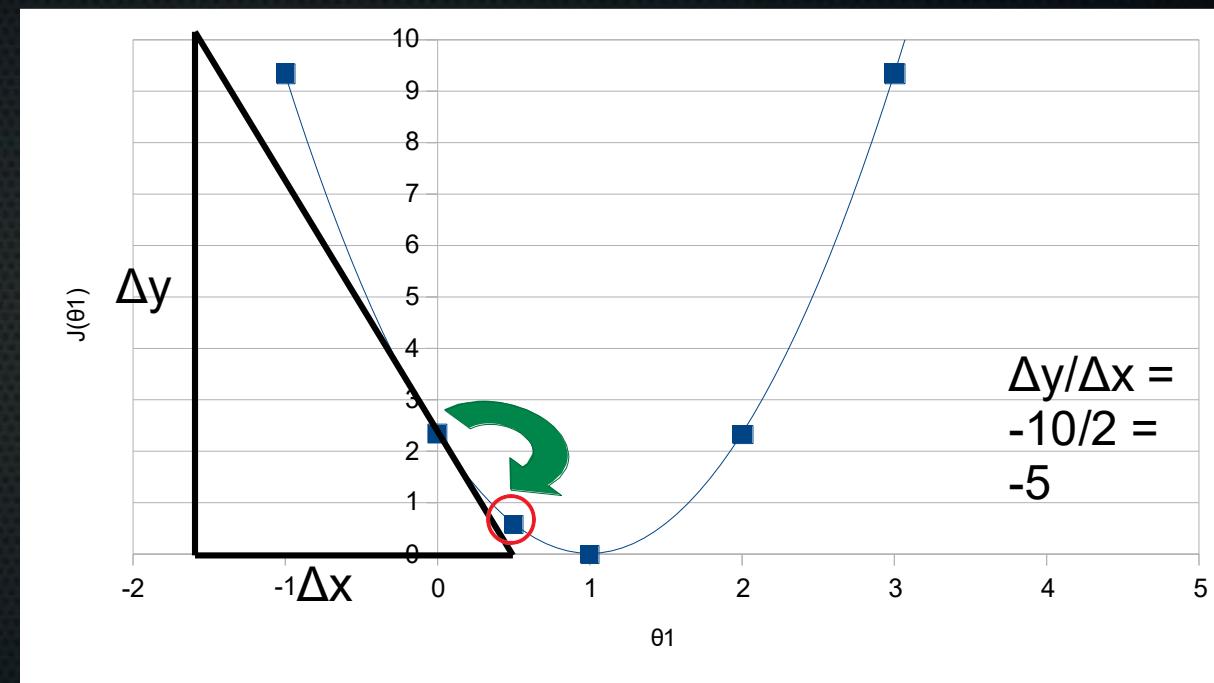
- Want to minimise
- Where to go?
- Change current theta1 as follows:

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

$$a = 0.1$$

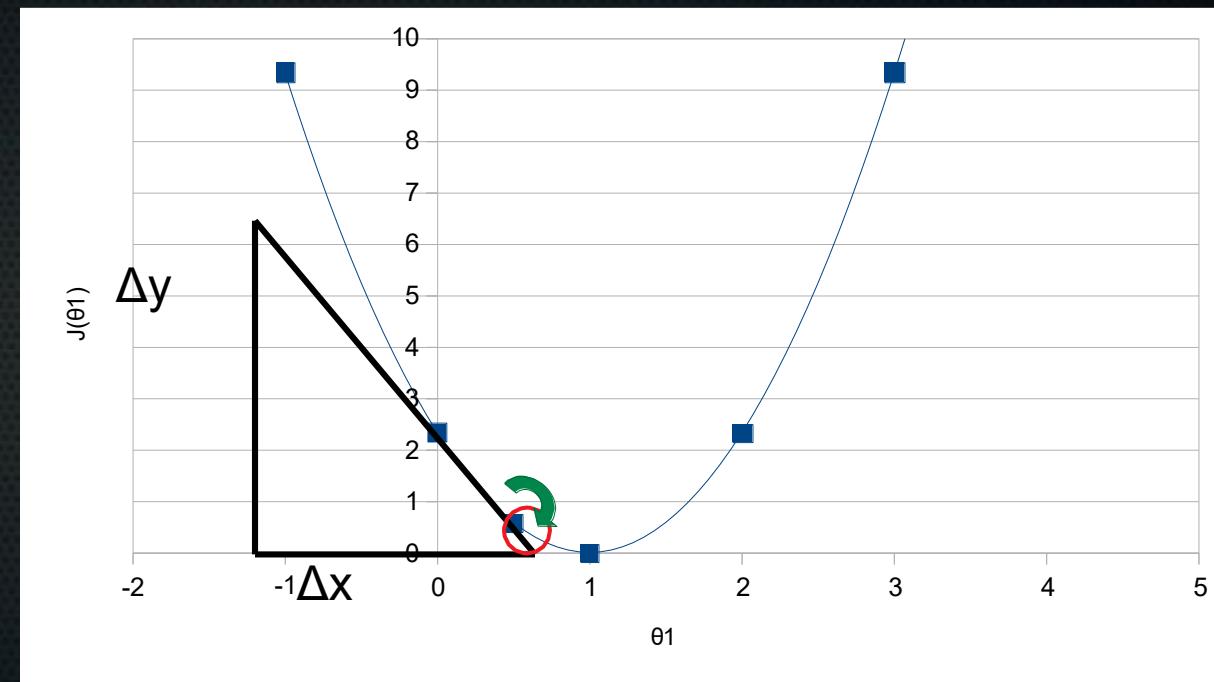
$$\theta_1 = 0 - 0.1 \cdot -5$$

$$\theta_1 = 0.5$$



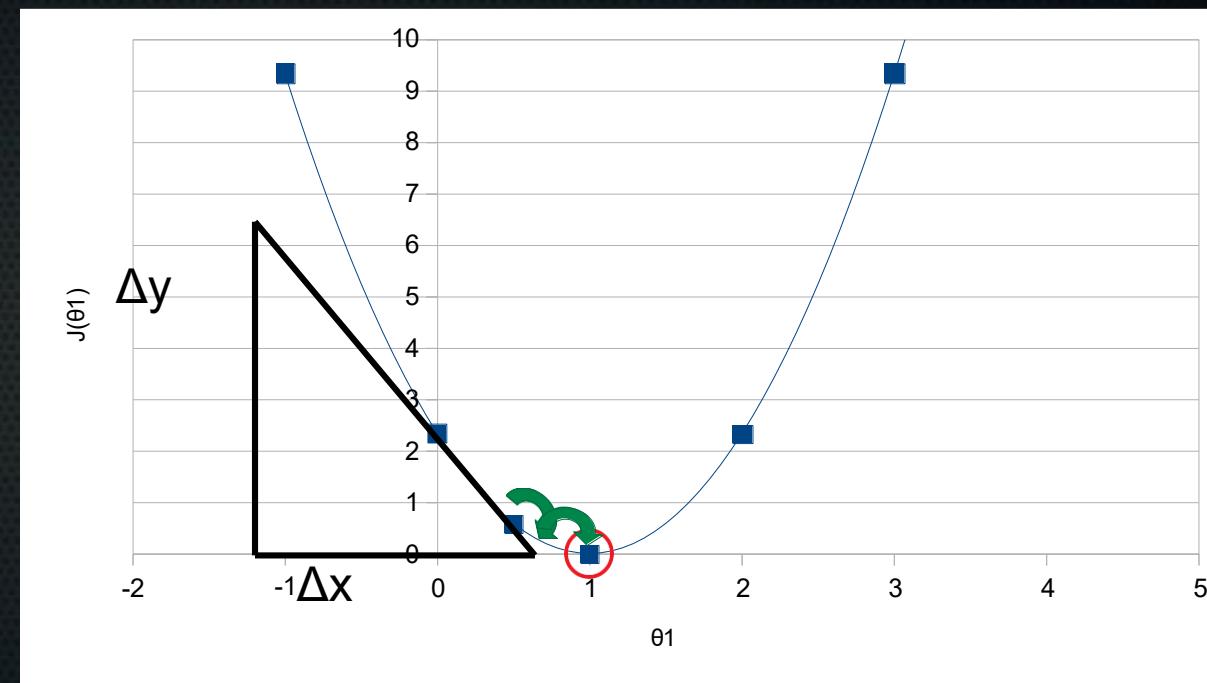
Gradient descent

- 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

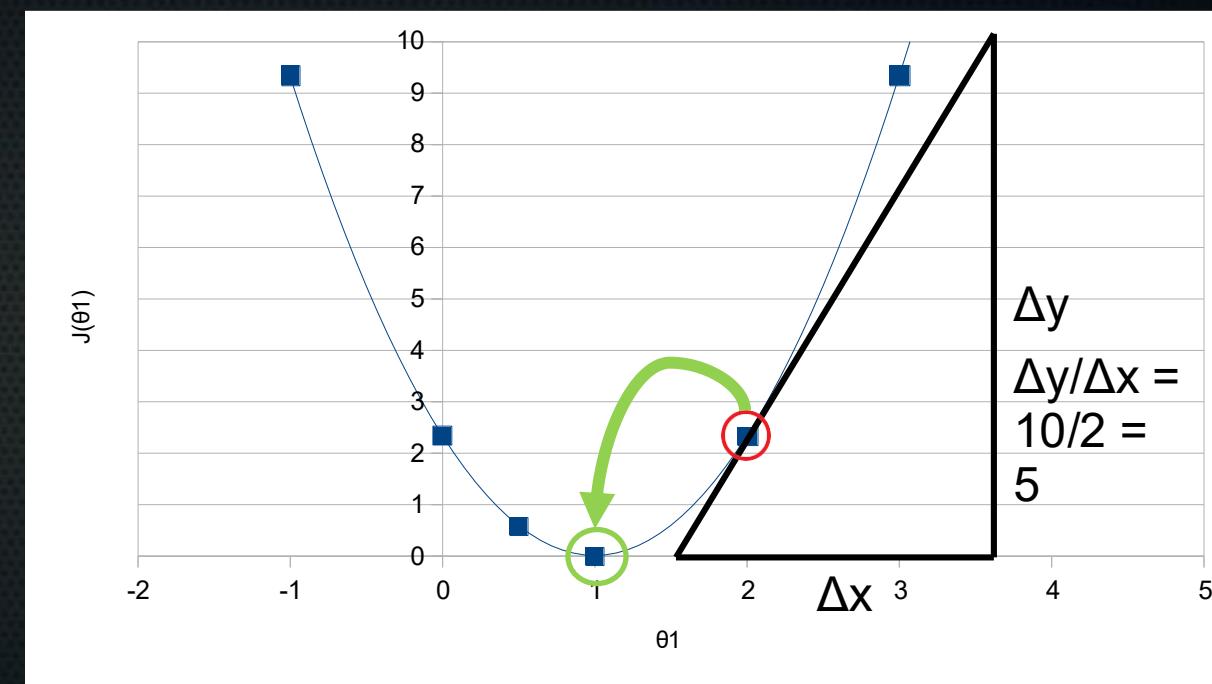
- Works also from other direction.

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

$$a = 0.2$$

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

$$\theta_1 = 1$$

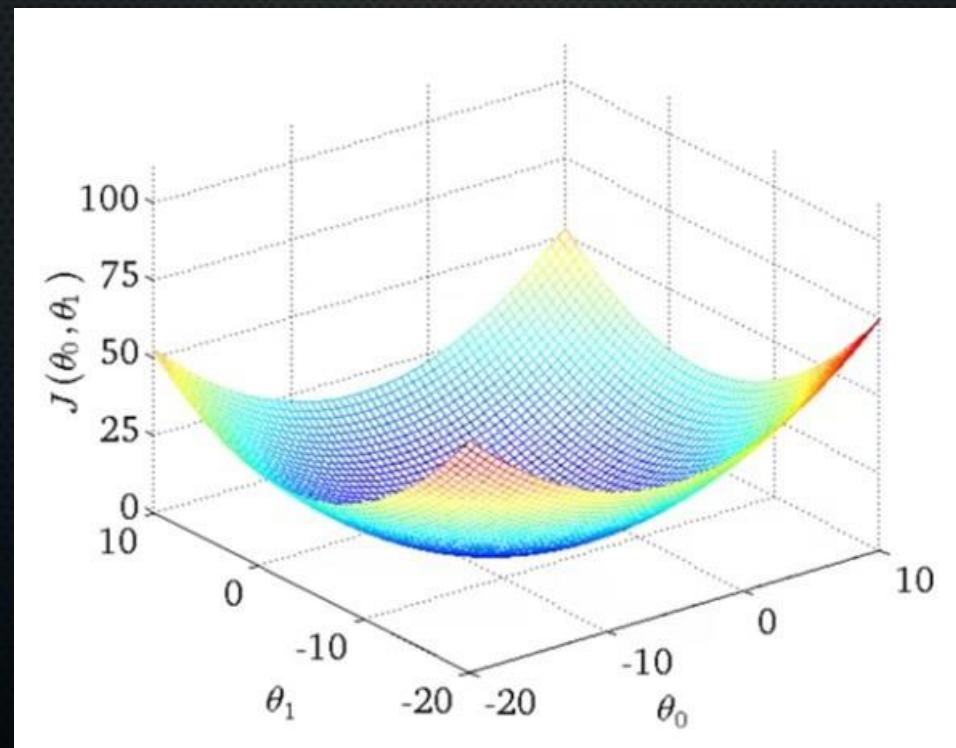


Gradient descent

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

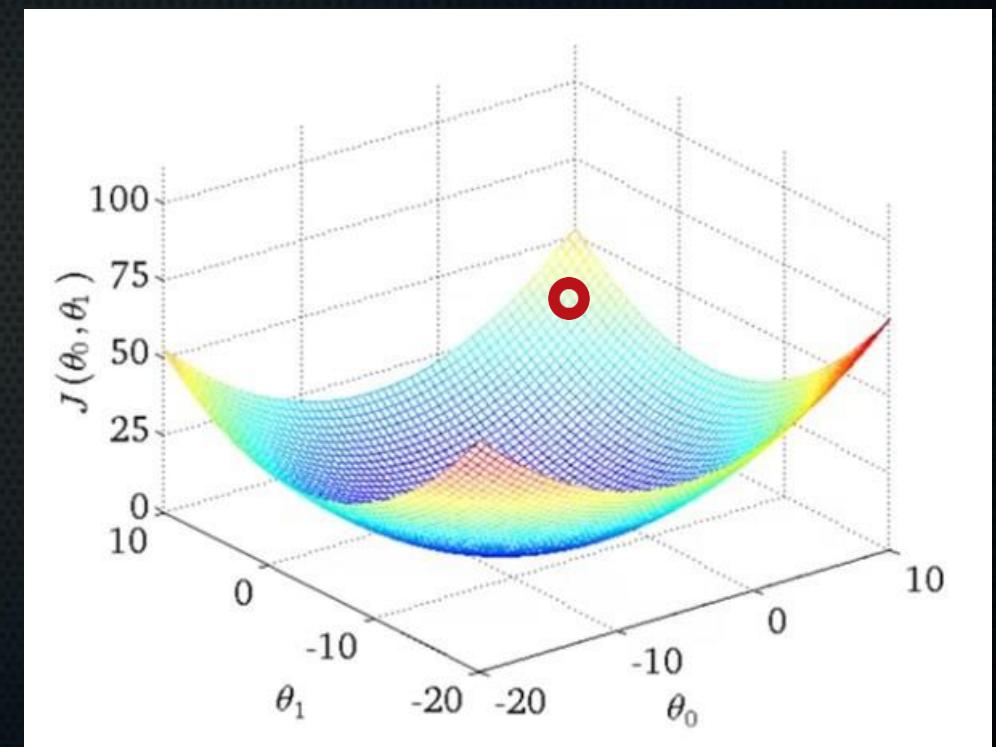
Gradient descent

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



Gradient descent: partial derivatives

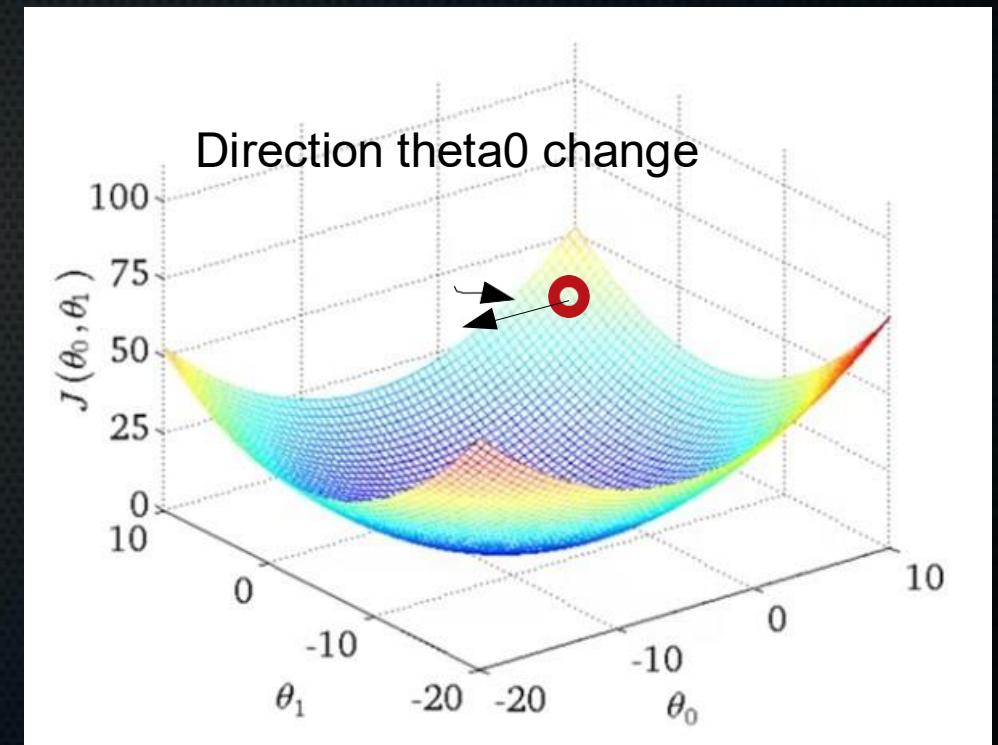
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional.
Want to fit intercept *and* slope.
- Use partial derivatives instead:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

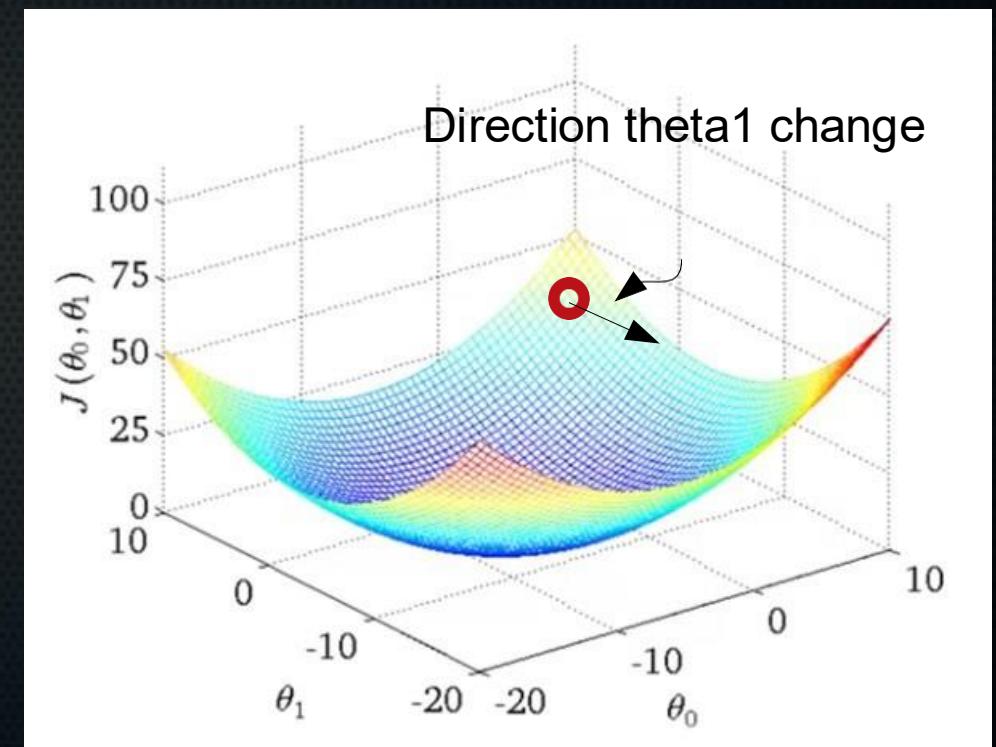
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional.
Want to fit intercept *and* slope.
- Use partial derivatives instead:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

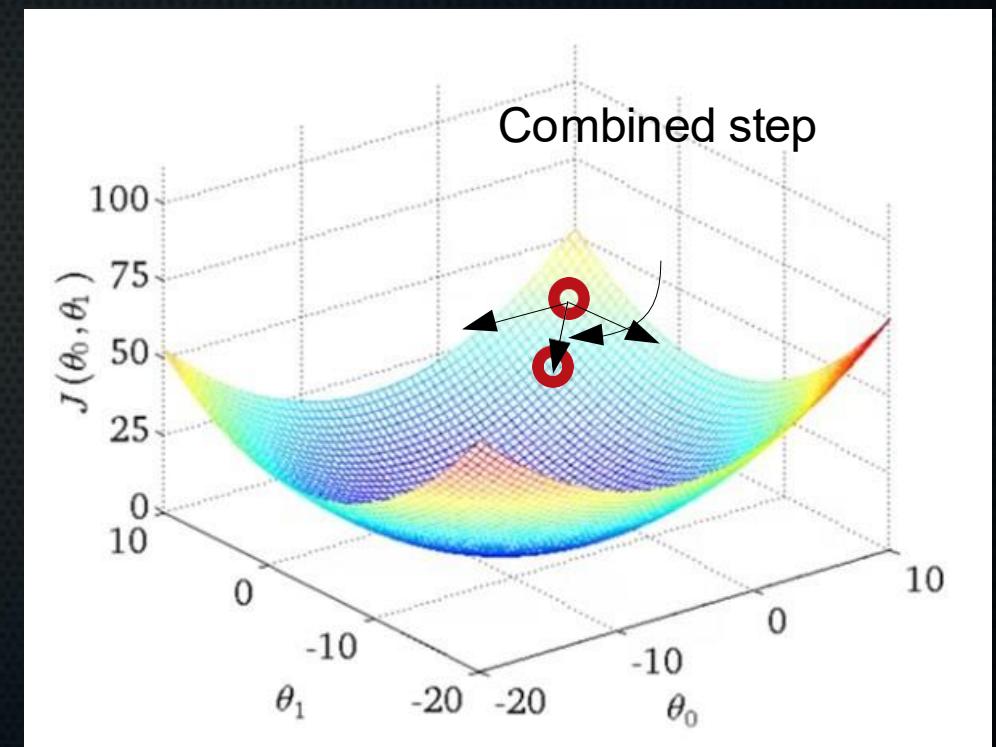
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional.
Want to fit intercept *and* slope.
- Use partial derivatives instead:



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

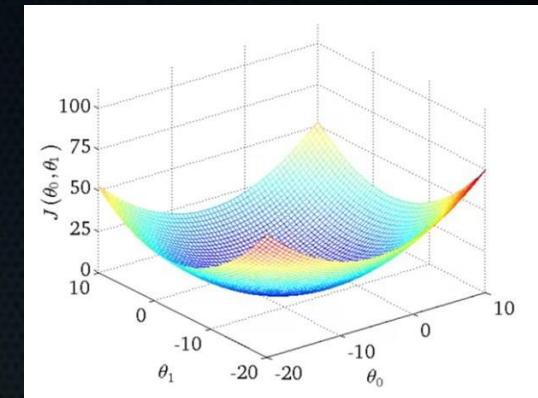
- Iteratively descend down the gradient of the cost function until convergence → optimal parameters!
- In reality: not one-dimensional.
Want to fit intercept *and* slope.
- Use partial derivatives instead:



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!

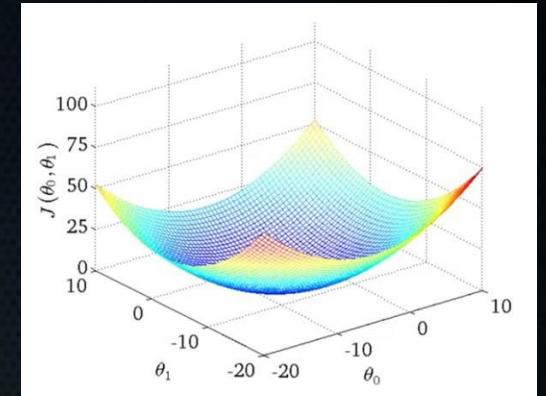


Source: Andrew Ng, Coursera

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

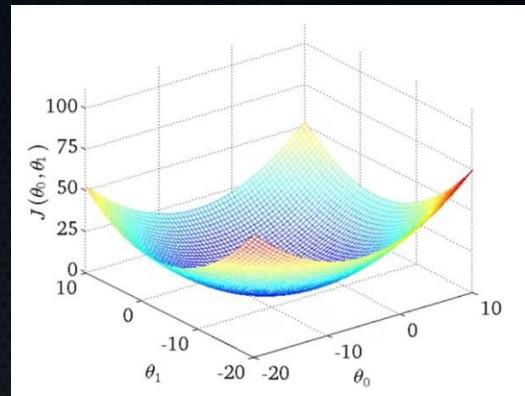
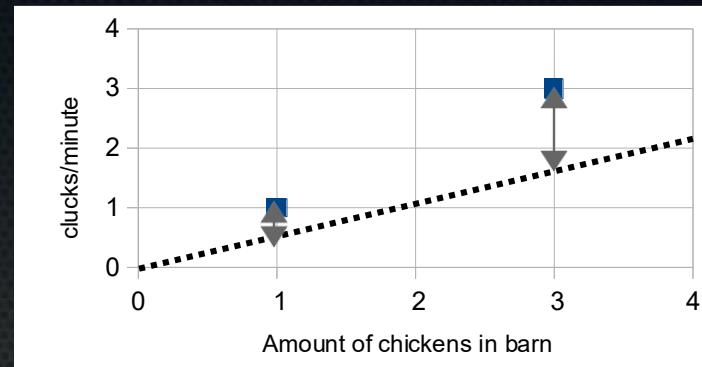
Gradient descent: partial derivatives

- 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

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

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

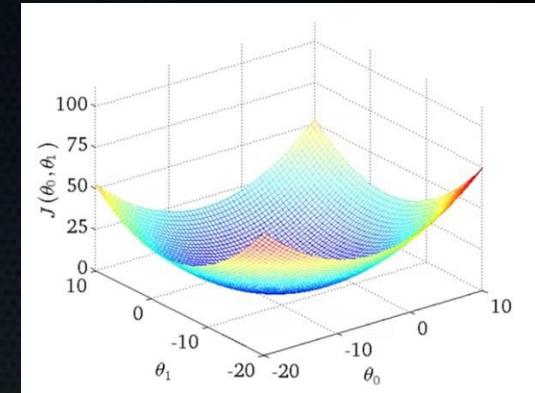
Gradient descent: partial derivatives

- Partial derivatives:

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

$$m=2$$

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



Source: Andrew Ng, Coursera

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

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

Gradient descent: partial derivatives

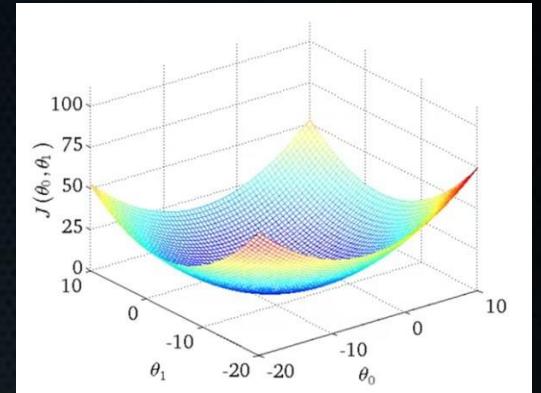
- Partial derivatives:

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

$$m=2$$

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

2x 2z



Source: Andrew Ng, Coursera

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

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

Gradient descent: partial derivatives

- Partial derivatives:

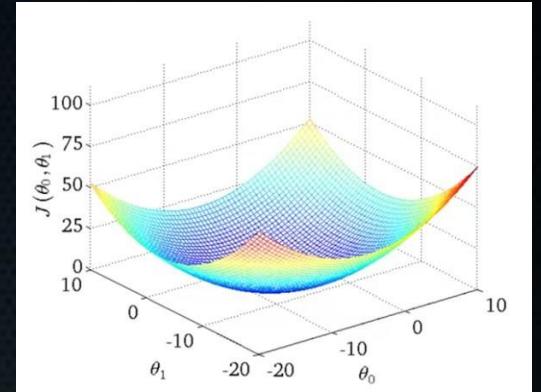
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (\theta_0 + \theta_1 \cdot x^{(i)} - y^{(i)})^2$$

$$m=2$$

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

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

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



Source: Andrew Ng, Coursera

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

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

Gradient descent: partial derivatives

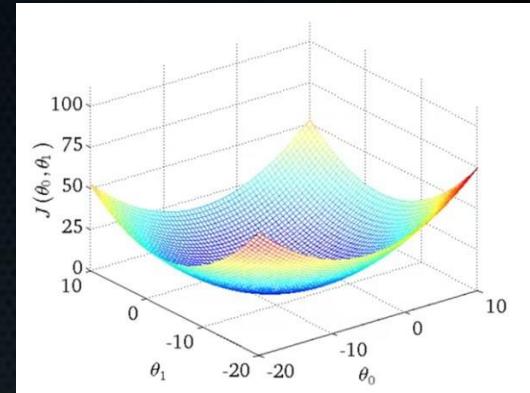
- Partial derivatives:

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

$$m=2$$

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

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



Source: Andrew Ng, Coursera

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

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

Gradient descent: partial derivatives

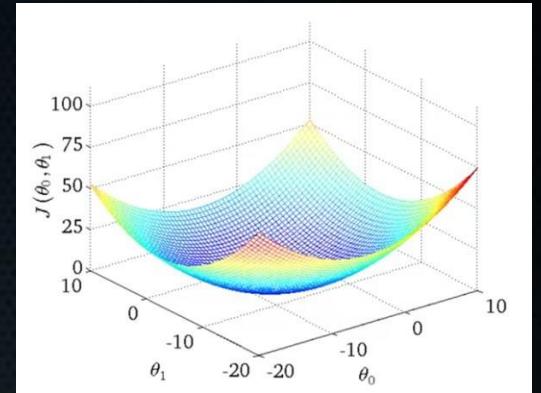
- Partial derivatives:

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

$$m=2$$

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

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



Source: Andrew Ng, Coursera

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

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

Gradient descent: partial derivatives

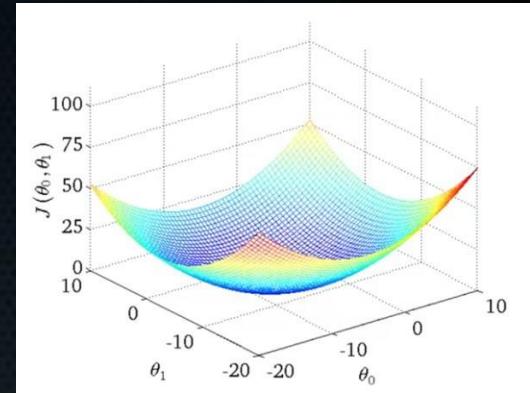
- Partial derivatives:

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

$$m=2$$

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

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



Source: Andrew Ng, Coursera

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

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

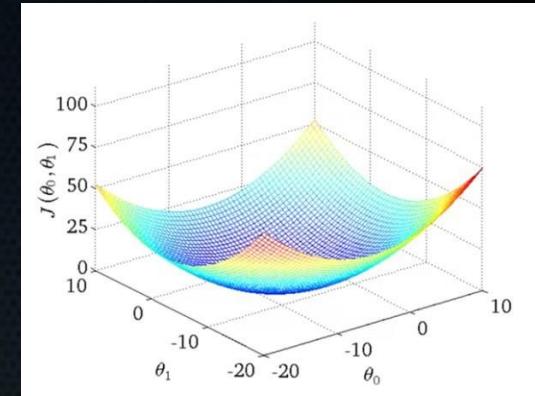
Gradient descent: partial derivatives

- Partial derivatives:

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

$$m=2$$

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



Source: Andrew Ng, Coursera

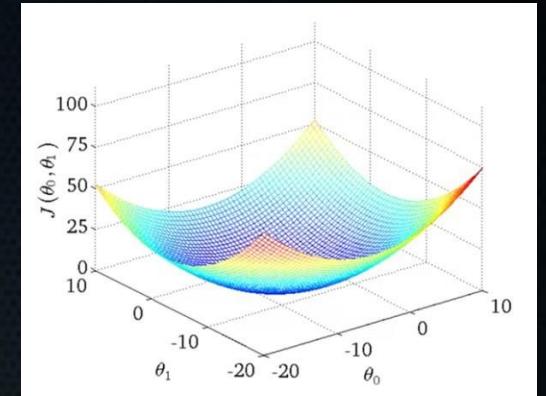
Gradient descent: partial derivatives

- Partial derivatives:

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

$$m=2$$

$$\begin{aligned}\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) &= \frac{1}{2 \cdot 2} \left(\underbrace{2(\theta_0 + \theta_1 \cdot x^{(1)} - y^{(1)}) * 1}_{+2(\theta_0 + \theta_1 \cdot x^{(2)} - y^{(2)}) * 1} \right) \\ &\quad + 2(\theta_0 + \theta_1 \cdot x^{(2)} - y^{(2)}) * 1\end{aligned}$$



Source: Andrew Ng, Coursera

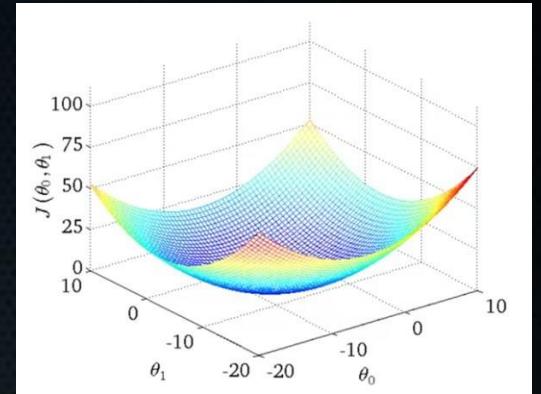
Gradient descent: partial derivatives

- 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

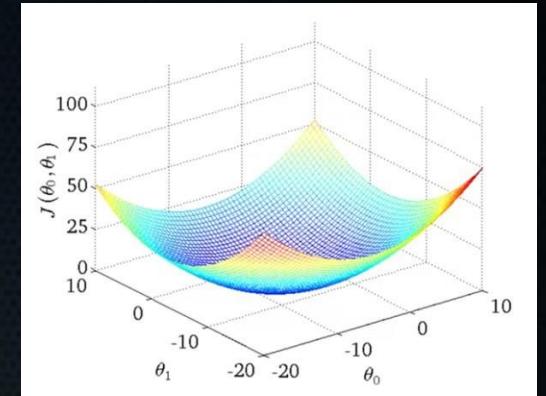
Gradient descent: partial derivatives

- Partial derivatives:

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

$$m=2$$

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



Source: Andrew Ng, Coursera

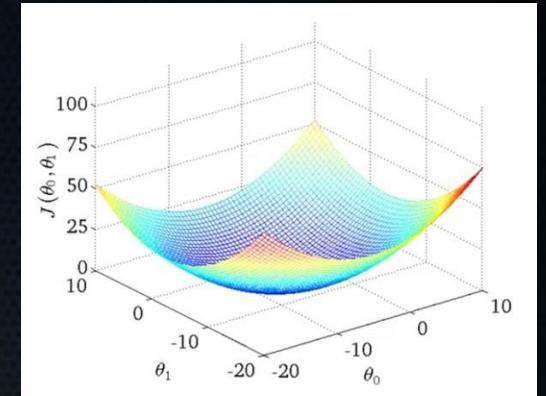
Gradient descent: partial derivatives

- 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

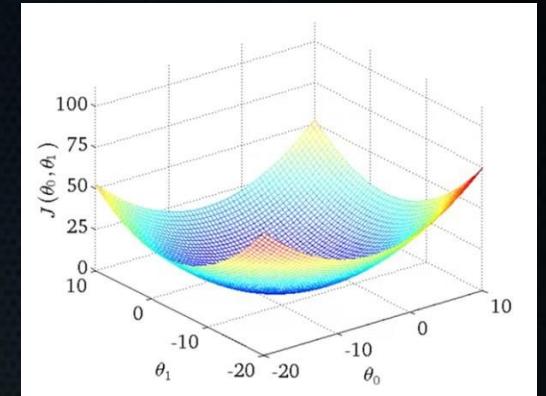
- Partial derivatives:

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

$$m=2$$

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

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



Source: Andrew Ng, Coursera

Gradient descent: partial derivatives

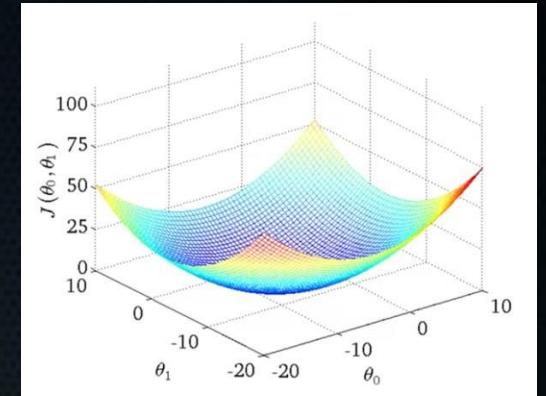
- Partial derivatives:

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

$$m=2$$

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

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



Source: Andrew Ng, Coursera

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

Gradient descent: partial derivatives

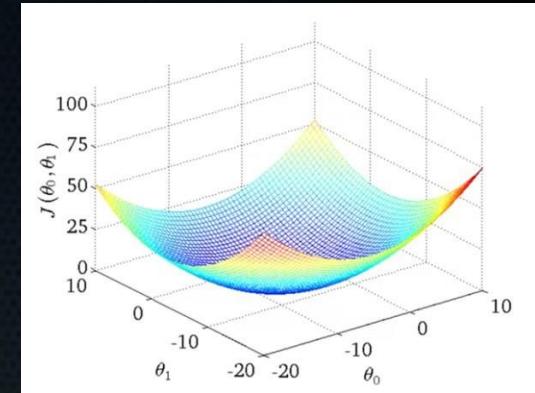
- Partial derivatives:

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

$$m=2$$

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

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



Source: Andrew Ng, Coursera

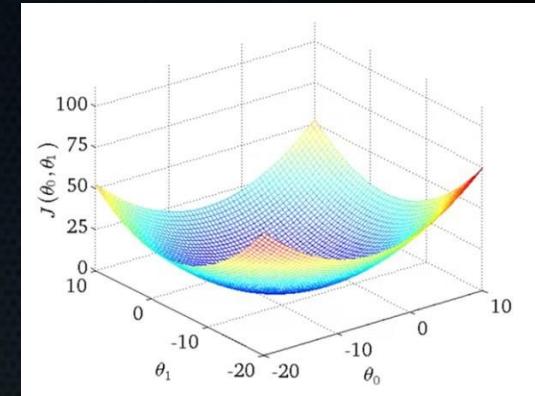
$$f(g(x))$$

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

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

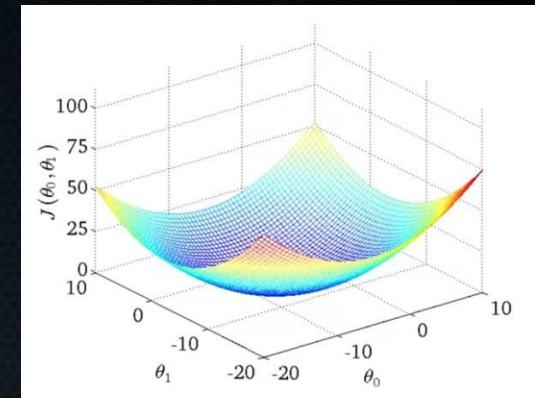
- Cost function:

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

- 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

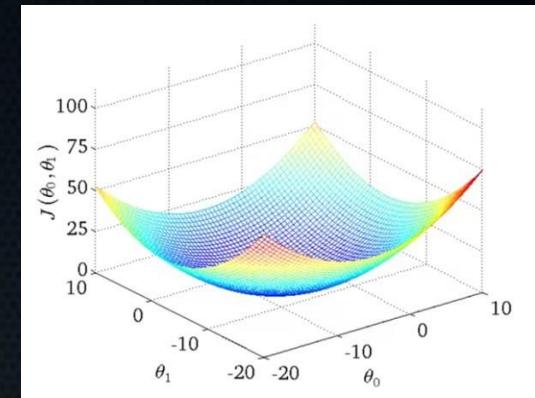
- Cost function:

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

- Gradient descent update:

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

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

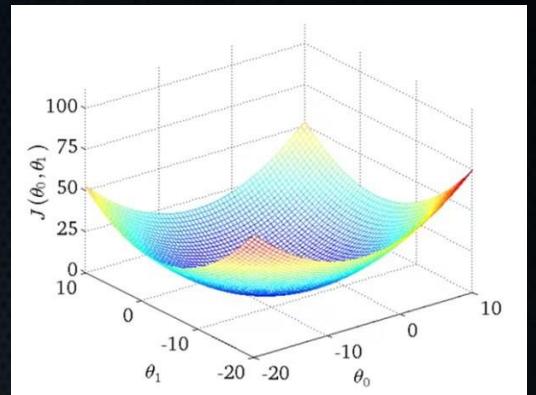


Source: Andrew Ng, Coursera

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

What about α ?

- Learning rate, so-called hyperparameter.

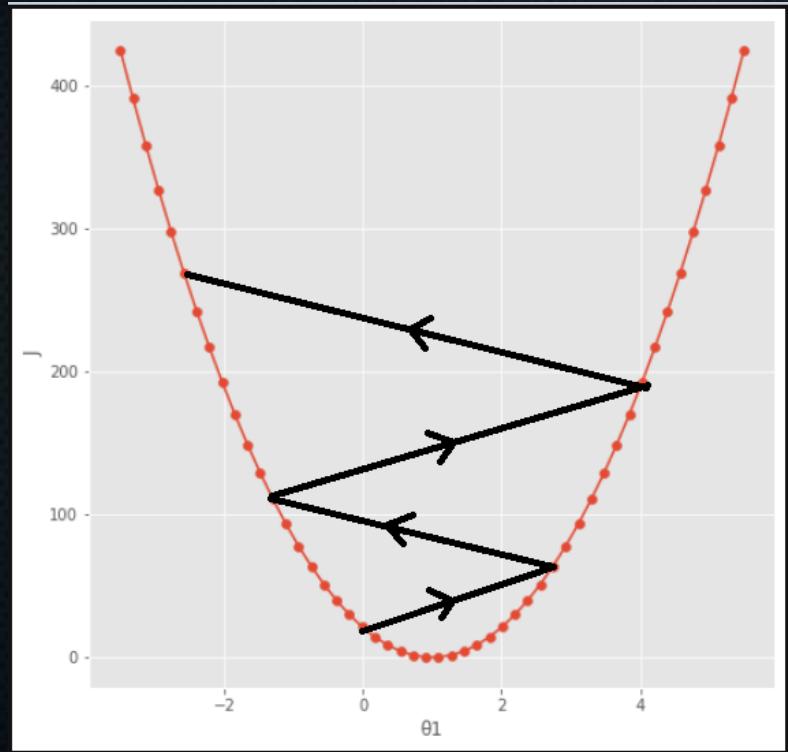


Source: Andrew Ng, Coursera

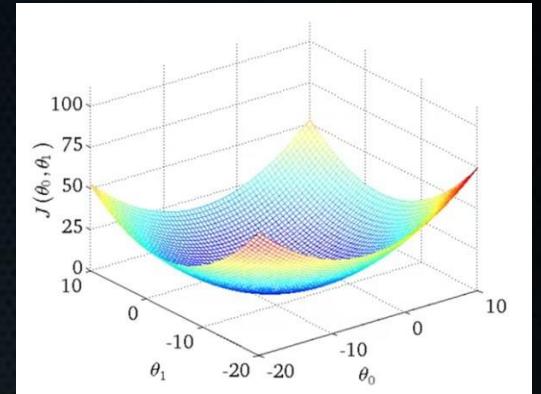
$$\theta_0 = \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \theta_0 - \frac{\alpha}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})$$

What about α ?

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



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

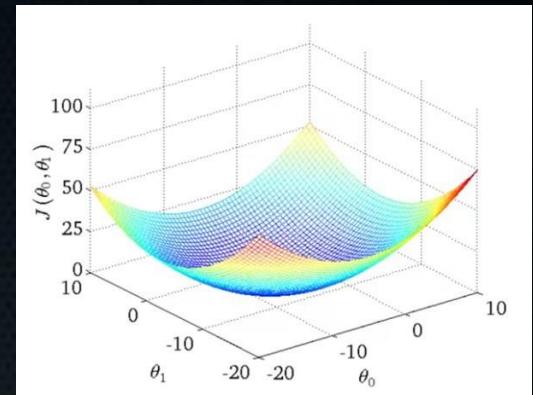
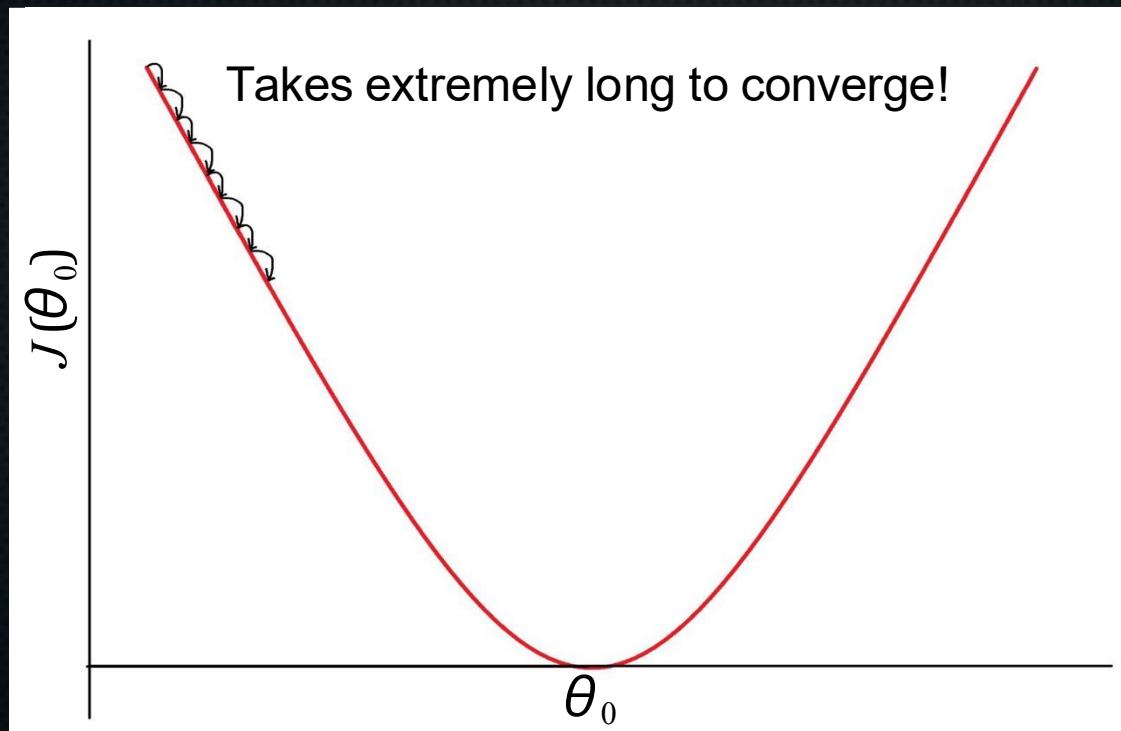


Source: Andrew Ng, Coursera

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

What about α ?

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

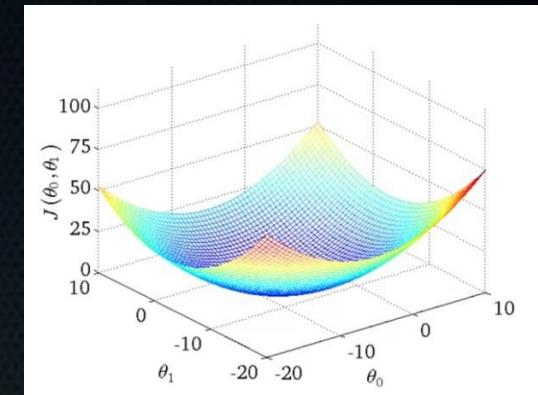


Source: Andrew Ng, Coursera

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

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.