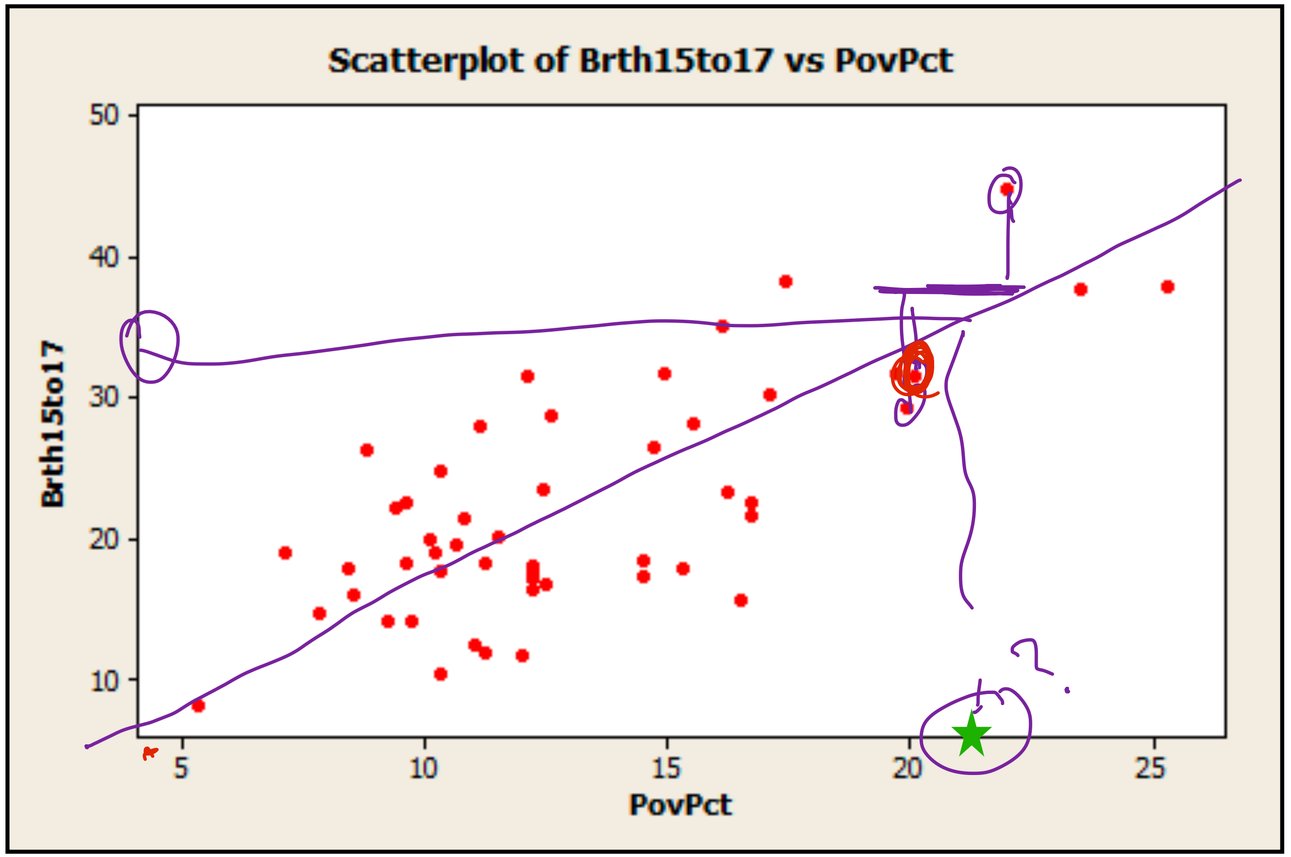
Introduction to Robot Intelligence
[Spring 2023]

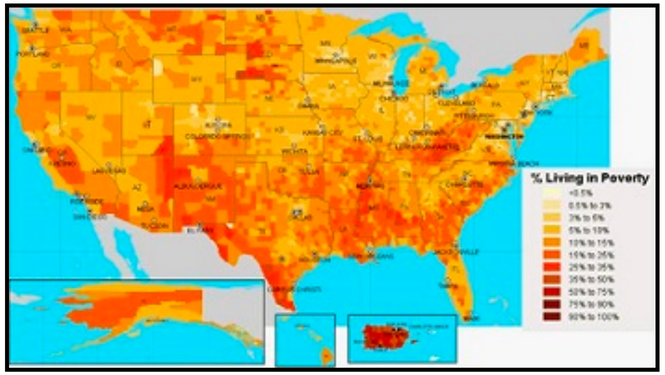
Supervised Learning with SGD

February 7, 2023

Lerrel Pinto

Framing





https://online.stat.psu.edu/stat462/node/101/

Topics for today

- What is supervised learning?
- Introduction to Linear Regression
- Gradient Descent for supervised learning

Supervised Learning

Label: Cat

Label: Dog

Supervised Learning

Label: Cat

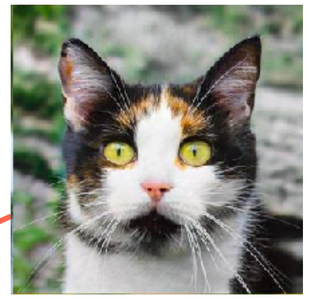
Label: Dog

Machine Learning Model

Supervised Learning

Label: Cat

Label: Dog



Machine Learning Model

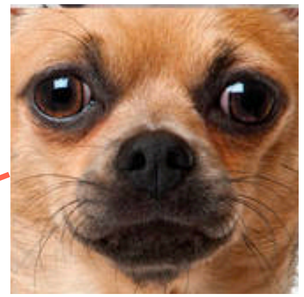
Cat

Supervised Learning

Label: Cat

Label: Dog

Machine Learning Model



Dog

Cat

Examples of Supervised Learning

- Face detection. (Input: Image of face; Output: Name of person)
- Medical diagnosis. (Input: Test results; Output: Risk for heart disease)
- Video tagging. (Input: Uploaded video; Output: Caption for the video)
- Poverty Estimation. (Input: Night light; Output: Income level)
- Robotics. (Input: Sensor data; Output: Joint velocities)

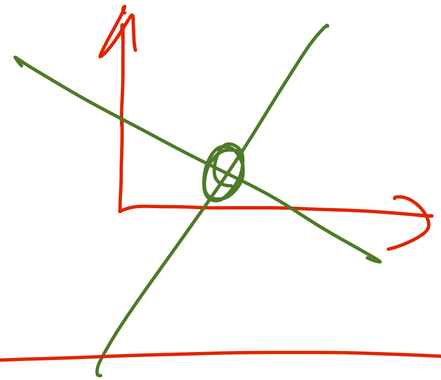
Math Recap – Solving Linear Equations

$5 + 8\omega_1 = 2$ | $\omega_1 = \frac{2-5}{8} = \frac{-3}{8}$

$5\omega_1 + 8\omega_2 = 10$

$3\omega_1 + 9\omega_2 = 11$

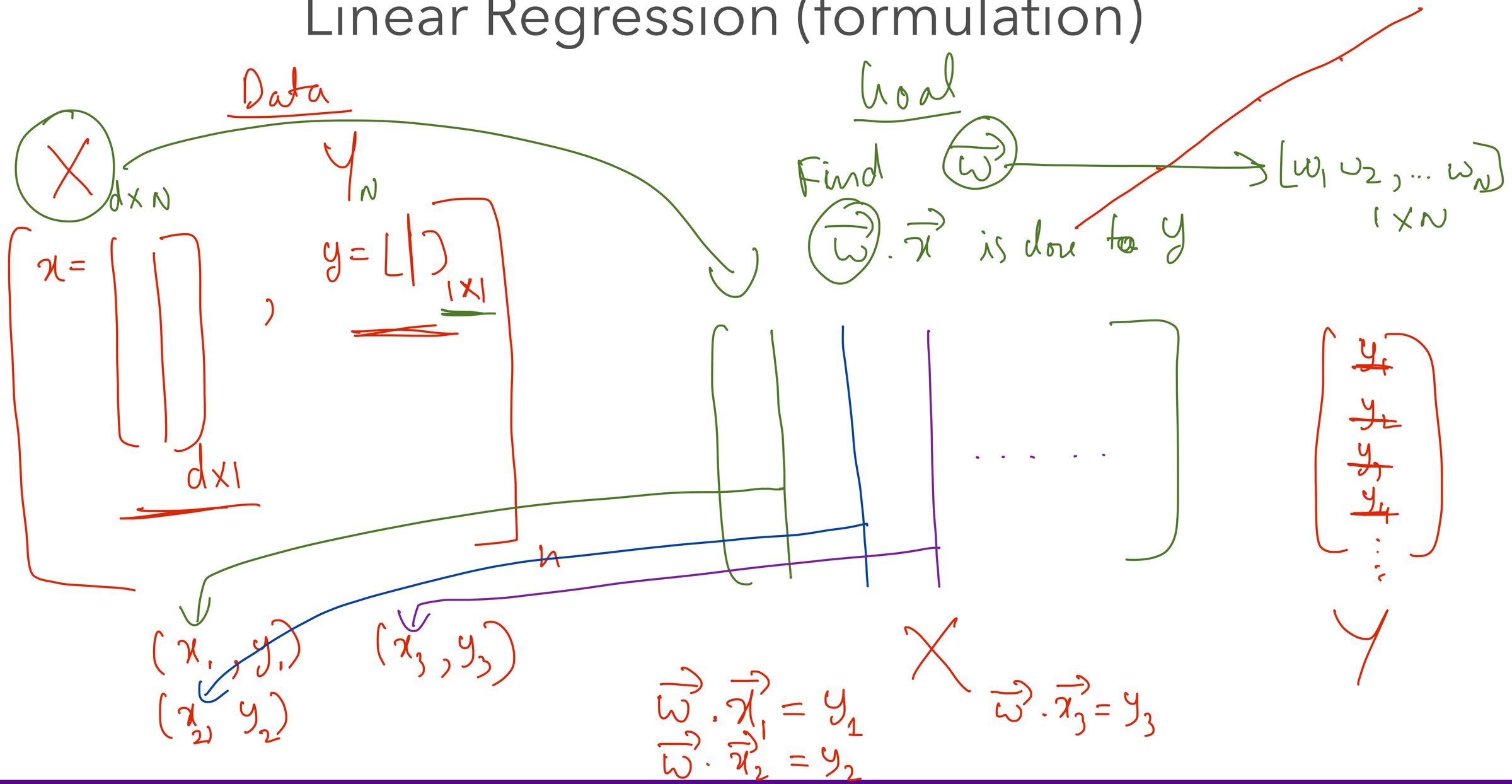
$\omega_1 = \frac{10 - 8\omega_2}{5}$



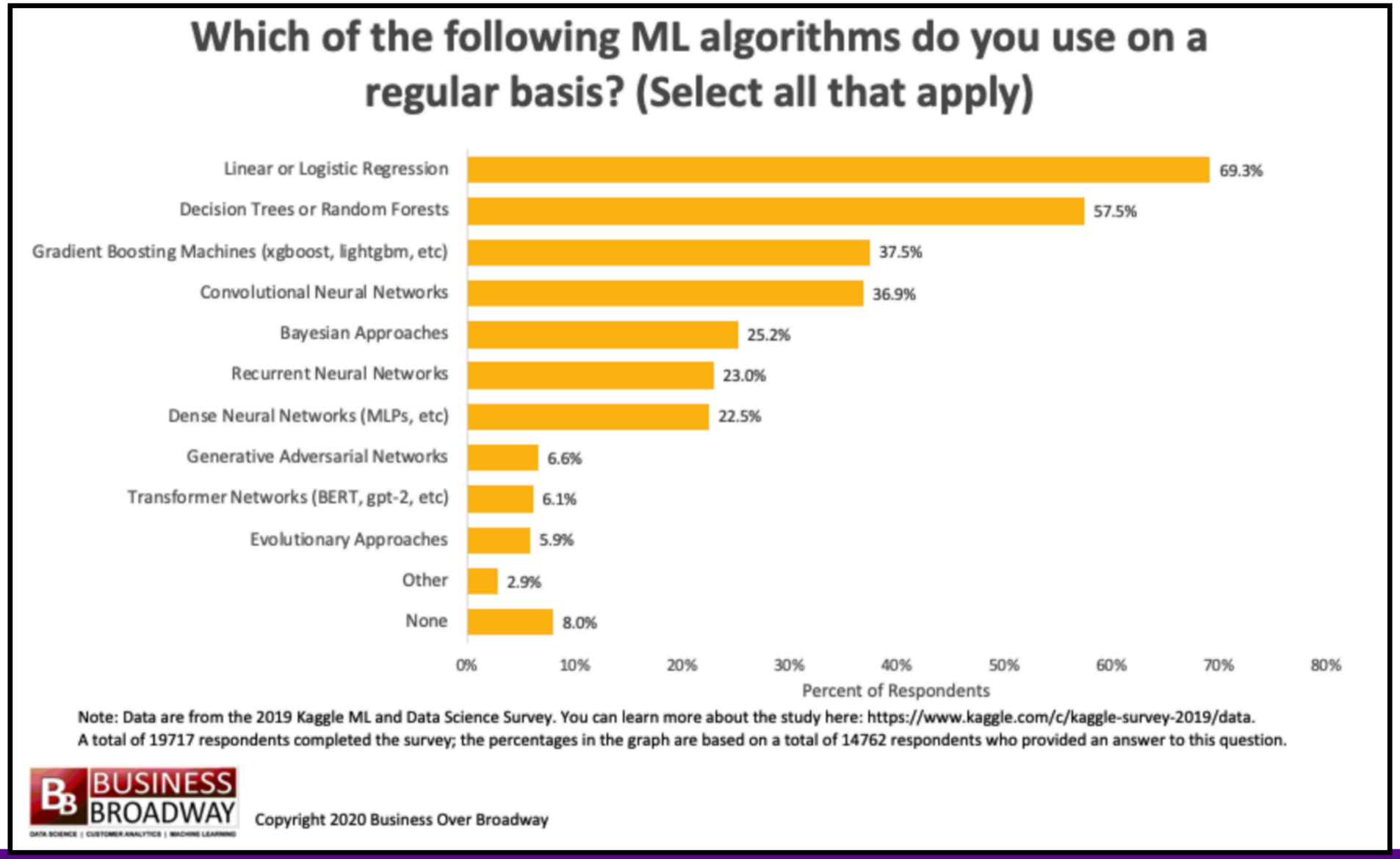
$$a_{11}\omega_1 + a_{12}\omega_2 + \dots a_{1m}\omega_m = b_1$$
$$a_{21}\omega_1 + a_{22}\omega_2 + \dots a_{2m}\omega_m = b_2$$
$$\vdots$$
$$a_{n1}\omega_1 + a_{n2}\omega_2 + \dots a_{nm}\omega_m = b_n$$

n equation
m variable

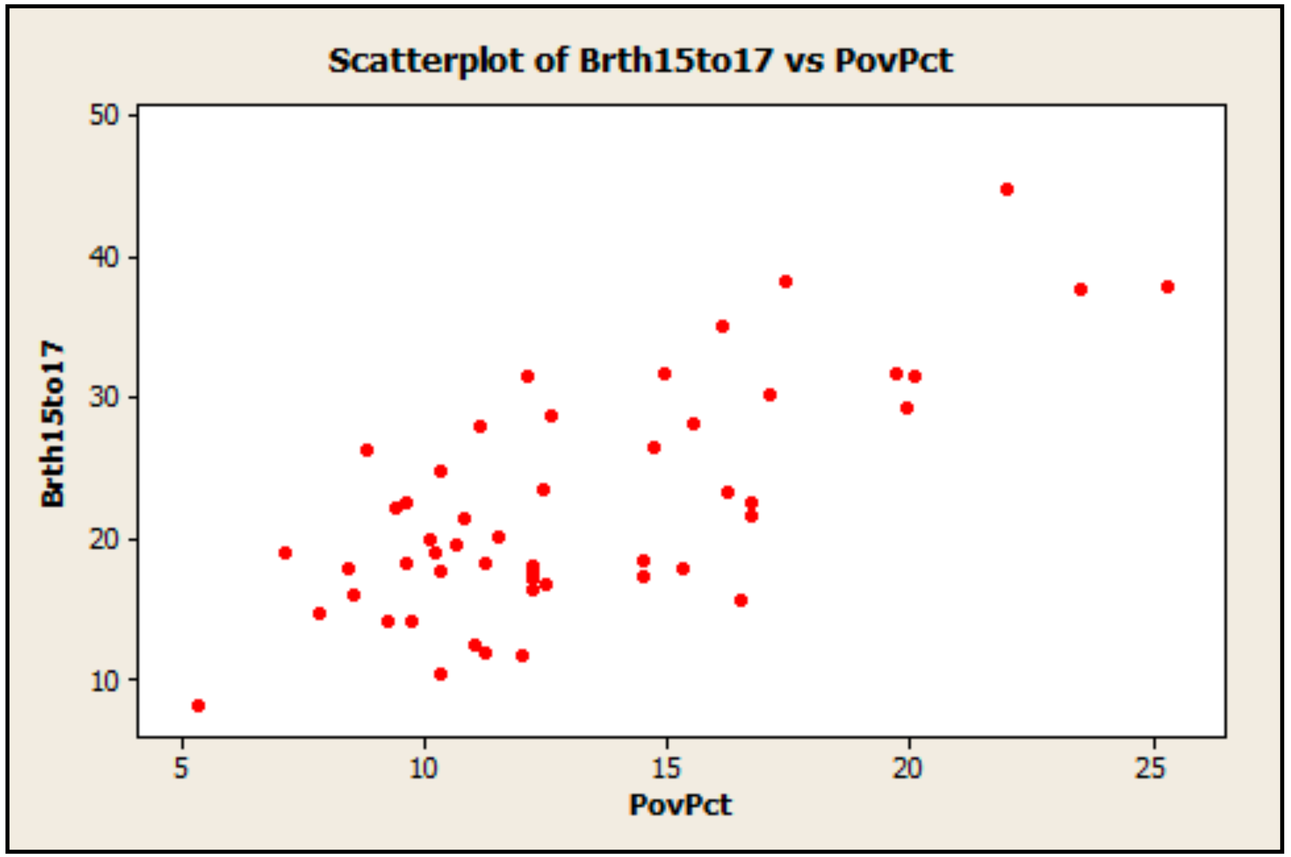
Linear Regression (formulation)

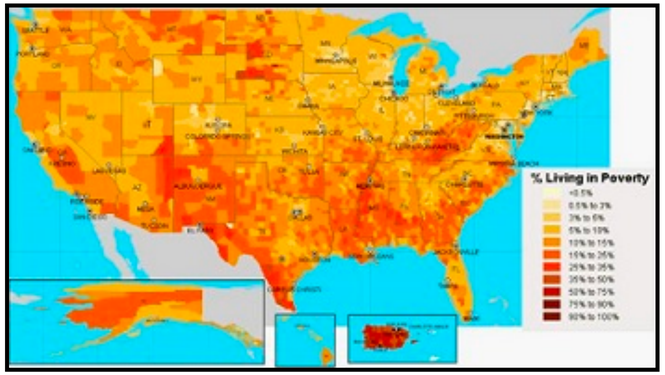


Linear Regression (why?)



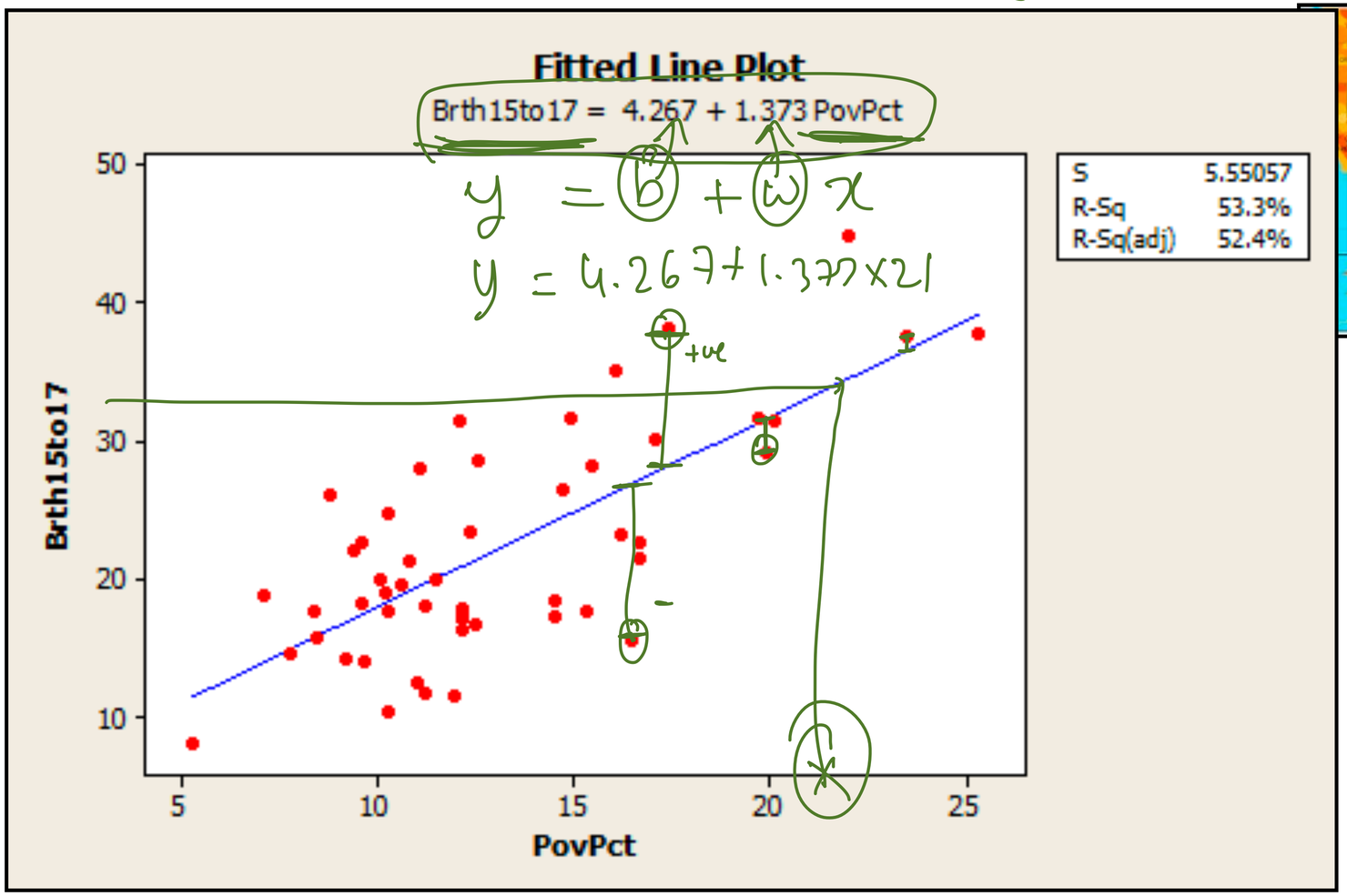
Linear Regression (example)

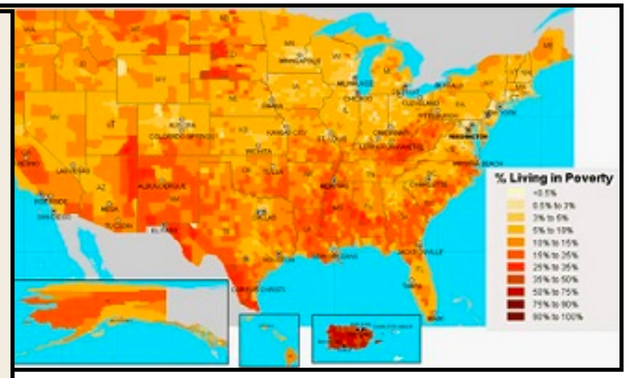




https://online.stat.psu.edu/stat462/node/101/

Linear Regression (example)





https://online.stat.psu.edu/stat462/node/101/

Linear Regression (objective)

$y = (ax+b)^2 \qquad x = -\frac{b}{2a}$

$\frac{dy}{dx} = 2(ax+b)a = 0$

$[e_1 e_2 \dots e_n] \times \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix} = e_1^2 + e_2^2 + \dots e_n^2$

Given

X $d \times n$ $\qquad$ y n

$\vec{w}$

$$\begin{bmatrix} x_1 \cdot w - y_1 \\ x_2 \cdot w - y_2 \\ \vdots \\ x_n \cdot w - y_n \end{bmatrix} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

minimize $\sum_{i=1}^{n} e_i^2$

$$= \sum_{i=1}^{n} (x_i^T w - y_i)^2$$

$$= \|X^T w - y\|^2$$

$$\begin{bmatrix} — x_1 — \\ — x_2 — \\ \vdots \\ — x_n — \end{bmatrix} \begin{bmatrix} w \\ 1 \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Eg. 1 $\quad \vec{w} \cdot \vec{x_1} \sim y_1 \rightarrow w^T x_1 \sim y_1$

$\rightarrow x_1^T w \sim y_1$

$e_1^2 = (x_1^T w - y_1)^2$

Eg. 2 $\quad e_2^2 = (x_2^T w - y_2)^2$

$\vdots$

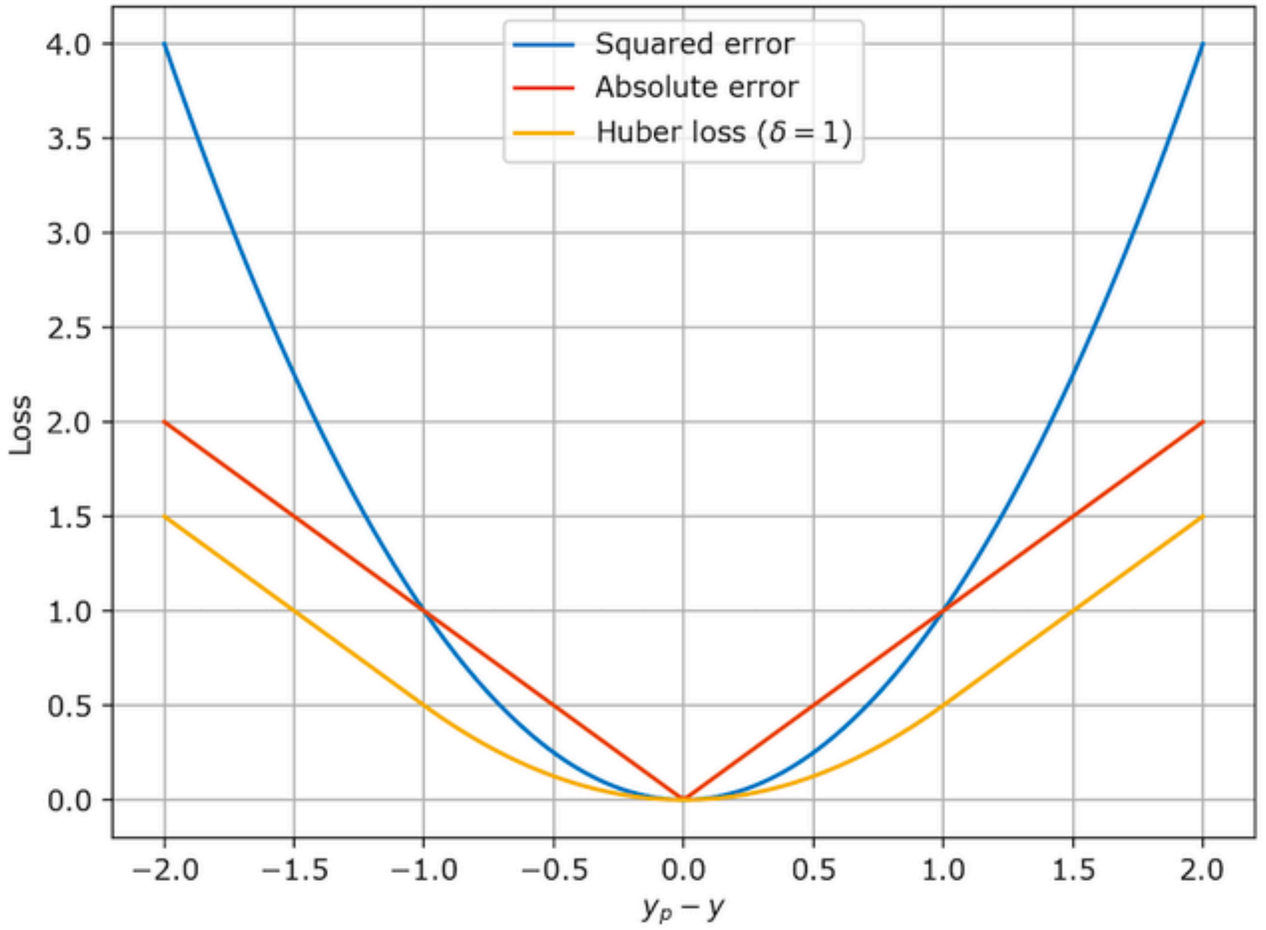
Eg. n $\quad e_n^2 = (x_n^T w - y_n)^2$

minimize $\sum_{i=1}^{n} e_i^2$

$|e_i|^2$

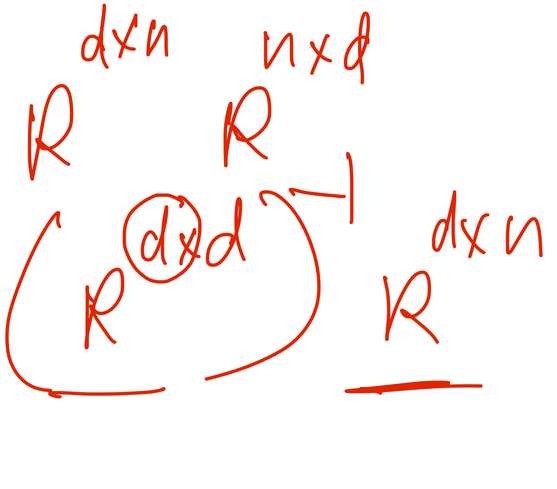
$\|e_i\|$

Linear Regression (error function)



Linear Regression

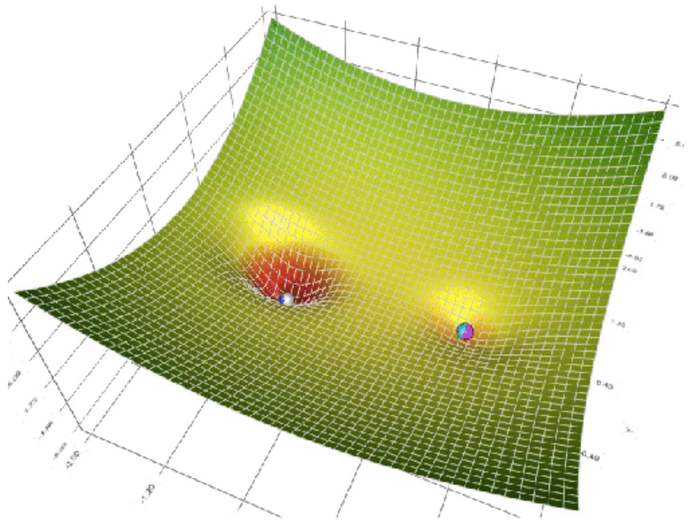
- Input data: $X \in \mathbb{R}^{d \times n}$, $Y \in \mathbb{R}^{n}$, where $(\vec{x} \in \mathbb{R}^{d}, y \in \mathbb{R}^{1})$ corresponds to a data point.
  - $n \rightarrow$ # of data points, $d \rightarrow$ # of features / input dim.
- Goal: to find $\vec{w} \in \mathbb{R}^{d}$ such that $< \vec{w}, \vec{x} > = y$
  - Minimize $\|X^{\mathsf{T}}\vec{w} - Y\|^{2}$
  - Solution: $\vec{w} = (XX^{\mathsf{T}})^{-1}XY$
    - Easy way to remember $\vec{w} = (X^{\mathsf{T}})^{+}Y$



Recap: Gradient Descent

Gradient Descent Algorithm with multiple params

- Given: cost / loss/ objective function $f(\vec{\theta})$. Where $\vec{\theta} \in \mathbb{R}^d$.
- Goal: find $\vec{\theta}^*$ such that $f(\vec{\theta}^*) = \min_{\vec{\theta}} f(\vec{\theta})$.
- Gradient descent solution:
  - Start from initial guess $\vec{\theta}^0$ and learning rate α
  - Update $\vec{\theta}^{i+1} \leftarrow \vec{\theta}^i - \alpha \nabla f(\vec{\theta})$
  - Repeat until change in θ is small, or maximum number of steps reached.



Linear Regression with Gradient Descent

Linear Regression with Gradient Descent

- Input data: $X \in \mathbb{R}^{d \times n}$, $Y \in \mathbb{R}^{n}$, where $(\vec{x} \in \mathbb{R}^{d}, y \in \mathbb{R}^{1})$ corresponds to a data point.
    - $n \rightarrow$ # of data points, $d \rightarrow$ # of features / input dim.
- Goal: to find $\vec{w} \in \mathbb{R}^{d}$ such that $< \vec{w}, \vec{x} > = y$
    - Minimize $\|X^{\mathsf{T}}\vec{w} - Y\|^{2}$
- Analytic Solution: $\vec{w} = (XX^{\mathsf{T}})^{-1}XY$

Linear Regression with Gradient Descent

- Input data: $X \in \mathbb{R}^{d \times n}$, $Y \in \mathbb{R}^{n}$, where $(\vec{x} \in \mathbb{R}^{d}, y \in \mathbb{R}^{1})$ corresponds to a data point.
    - $n \rightarrow$ # of data points, $d \rightarrow$ # of features / input dim.
- Goal: to find $\vec{w} \in \mathbb{R}^{d}$ such that $< \vec{w}, \vec{x} > = y$
    - Minimize $\|X^{\mathsf{T}}\vec{w} - Y\|^{2}$
- Loss function $f(\vec{w}) = \|X^{\mathsf{T}}\vec{w} - Y\|^{2} = (X^{\mathsf{T}}\vec{w} - Y)^{\mathsf{T}}(X^{\mathsf{T}}\vec{w} - Y)$

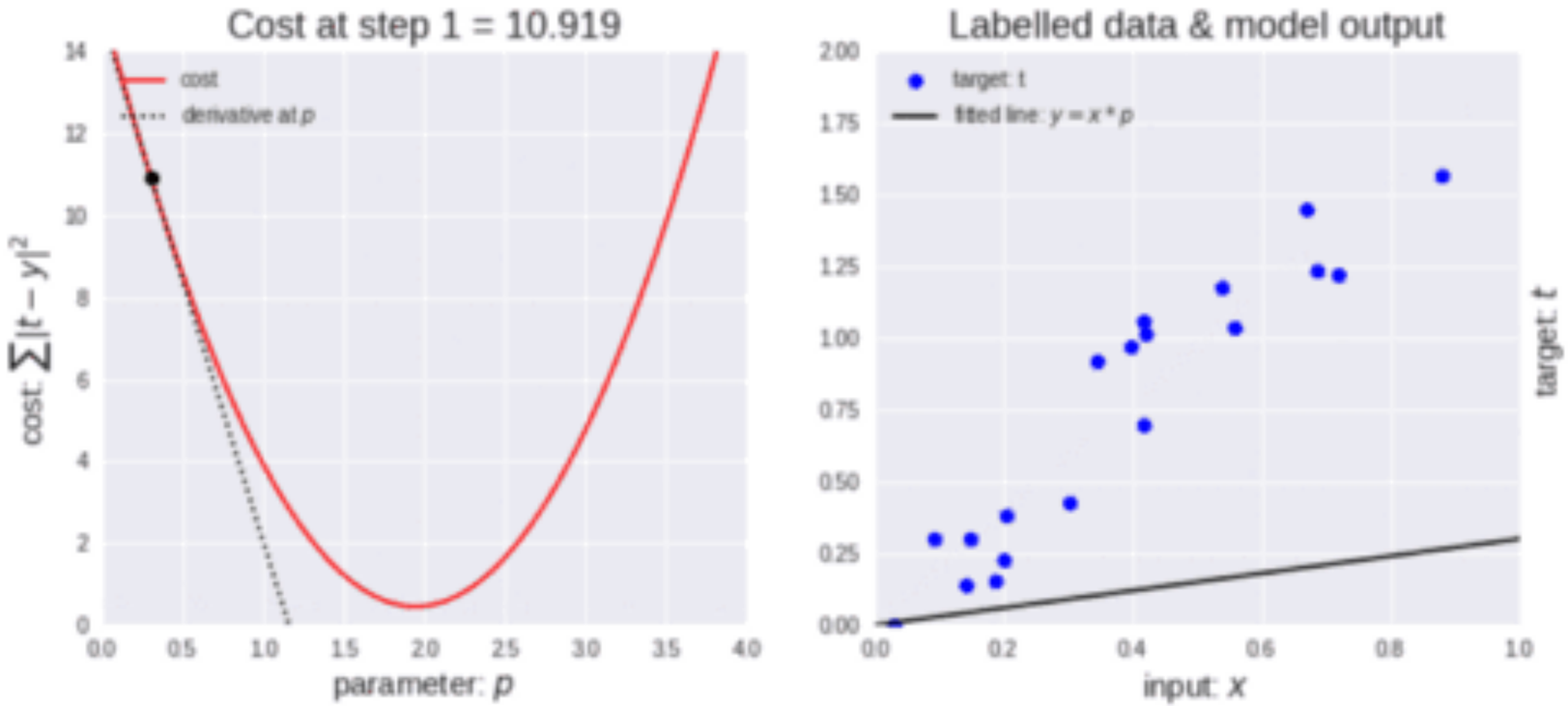
Linear Regression with Gradient Descent

- Loss function $f(\overrightarrow{w}) = \|X^\intercal \overrightarrow{w} - Y\|^2 = (X^\intercal \overrightarrow{w} - Y)^\intercal (X^\intercal \overrightarrow{w} - Y)$

Linear Regression with Gradient Descent

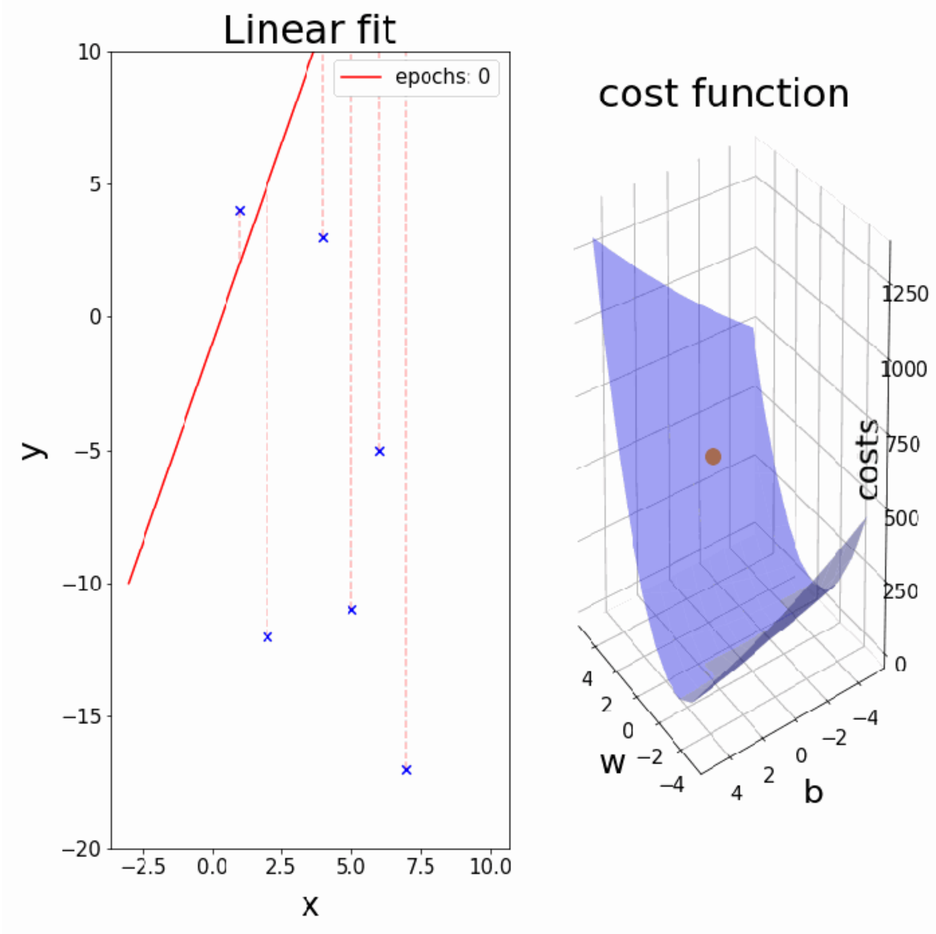
- Loss function $f(\vec{w}) = \|X^\mathsf{T}\vec{w} - Y\|^2 = (X^\mathsf{T}\vec{w} - Y)^\mathsf{T}(X^\mathsf{T}\vec{w} - Y)$
  - Set $f(\vec{w}) = \sum_{i=1}^{n} (x^i . \vec{w} - y^i)^2$
  - Compute $\frac{\partial f}{\partial w_k} = \sum_{i=1}^{n} 2(\vec{w}^T x^i - y^i) x_k^i$
  - Compute $\nabla f = [\frac{\partial f}{\partial w_1}, \cdots, \frac{\partial f}{\partial w_k}, \cdots, \frac{\partial f}{\partial w_d}]$
  - Use gradient descent

Linear Regression with Gradient Descent



Credits: https://www.kdnuggets.com/

Linear Regression with Gradient Descent



Credits: Tobias Roeschl

Gradient Descent Algorithm for Machine Learning

Gradient Descent Algorithm for Machine Learning

- Given: cost / loss/ objective function $f(\vec{\theta}, D)$. Where $\vec{\theta} \in \mathbb{R}^d$.

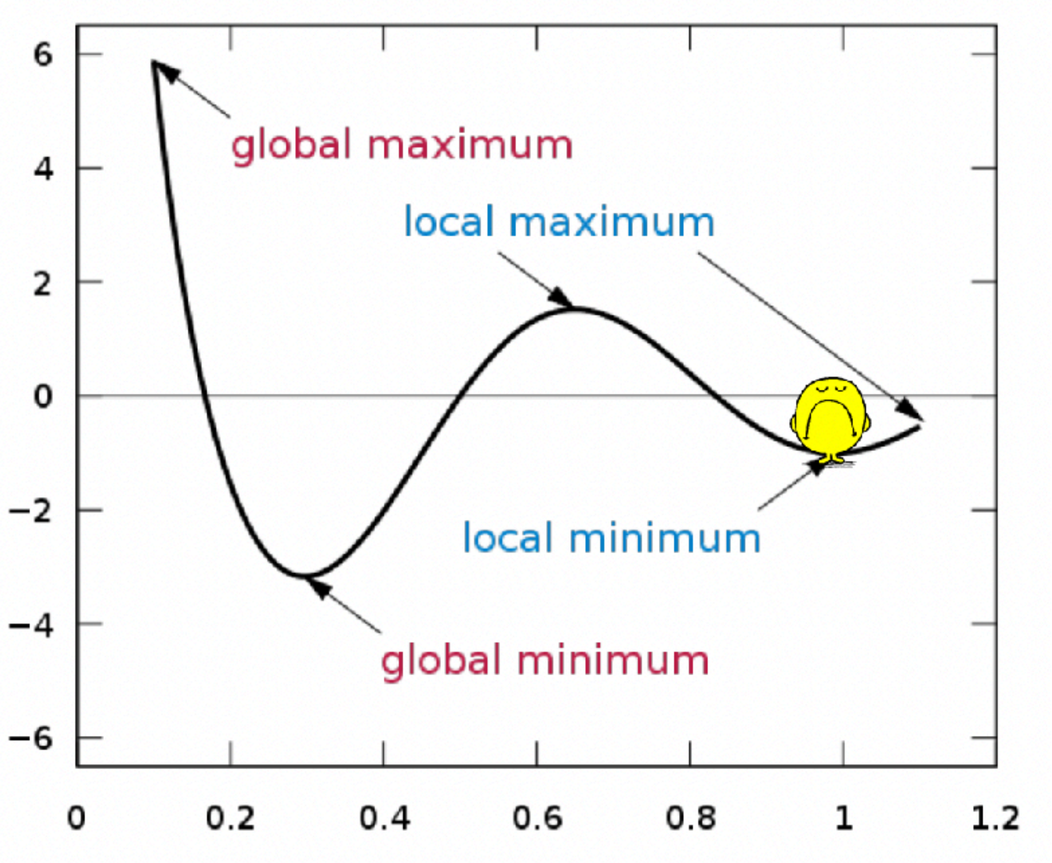
Gradient Descent Algorithm for Machine Learning

- Given: cost / loss/ objective function $f(\vec{\theta}, D)$. Where $\vec{\theta} \in \mathbb{R}^d$.
- Goal: find $\vec{\theta}*$ such that $f(\vec{\theta}*, D) = \min_{\vec{\theta}} f(\vec{\theta}, D)$.

Gradient Descent Algorithm for Machine Learning

- Given: cost / loss/ objective function $f(\vec{\theta}, D)$. Where $\vec{\theta} \in \mathbb{R}^d$.
- Goal: find $\vec{\theta}*$ such that $f(\vec{\theta}*, D) = \min_{\vec{\theta}} f(\vec{\theta}, D)$.
- Gradient descent solution:
  - Start from initial guess $\vec{\theta}^0$ and learning rate α
  - Update $\vec{\theta}^{i+1} \leftarrow \vec{\theta}^i - \alpha \nabla f(\vec{\theta}, D)$
  - Repeat until change in θ is small, or maximum number of steps reached.

Key issue: Local minima



Credits: Michail Michailidis & Patrick Maiden

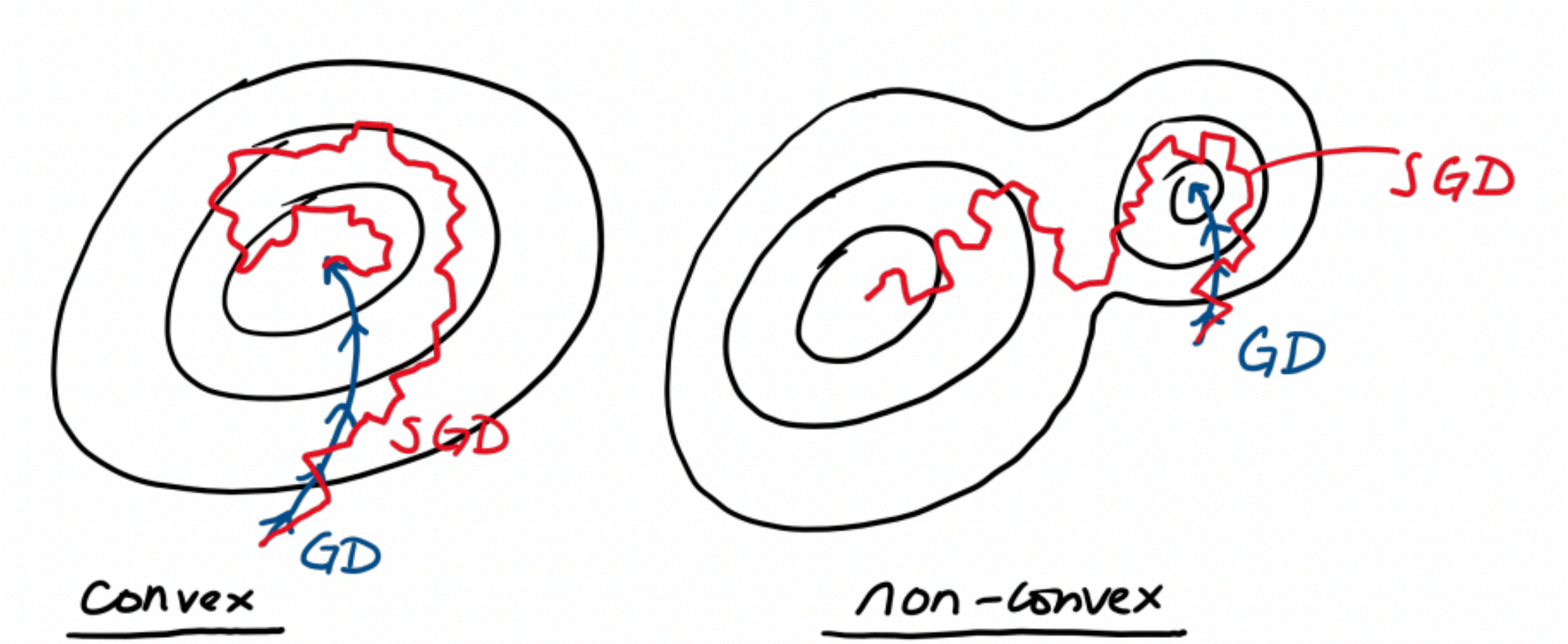
One solution: Stochastic Gradient Descent (SGD)

- Gradient descent solution:
  - Start from initial guess $\overrightarrow{\theta}^0$ and learning rate α
  - Update $\overrightarrow{\theta}^{i+1} \leftarrow \overrightarrow{\theta}^i - \alpha \nabla f(\overrightarrow{\theta}, D)$

One solution: Stochastic Gradient Descent (SGD)

- Gradient descent solution:
  - Start from initial guess $\overrightarrow{\theta}^0$ and learning rate α
  - Update $\overrightarrow{\theta}^{i+1} \leftarrow \overrightarrow{\theta}^i - \alpha \nabla f(\overrightarrow{\theta}, D)$
- SGD:
  - Sample single or multiple datapoints $d \sim D$
  - Update $\overrightarrow{\theta}^{i+1} \leftarrow \overrightarrow{\theta}^i - \alpha \nabla f(\overrightarrow{\theta}, d)$

One solution: Stochastic Gradient Descent (SGD)



Credits: Stanley Chan

Additional Reading

- https://www.mit.edu/~6.s085/notes/lecture3.pdf
- https://www.coconino.edu/resources/files/pdfs/academics/sabbatical-reports/kate-kozak/chapter_10.pdf
- Interactive tutorial: https://uclaacm.github.io/gradient-descent-visualiser/
- Book chapter: https://www.cs.utah.edu/~jeffp/IDABook/T6-GD.pdf
- SGD + variants: https://ruder.io/optimizing-gradient-descent

Questions?