

Applied Machine Learning

Linear Regression

Isabeau Prémont-Schwarz



COMP 551 (Fall 2024)

In-person office hours in **Trottier 3104**. See <https://edstem.org/us/courses/63321/discussion/5247282>

The following office hours by Santosh and Hilal will be specifically to answer questions about math and notation which you may have.

@Santosh

10 Sep: 4 PM - 5 PM

11 Sep: 4 PM - 5 PM

13 Sep: 4 PM - 5 PM

@Hilal

10 Sep: 10 AM - 11 AM

12 Sep: 10 AM - 11 AM

17 Sep: 10 AM - 11 AM

Click for Office Hours Schedule

Navigation

- [Home](#)
- [READ THIS!](#)
- [Textbooks](#)
- [Key Dates](#)
- [Lectures](#)
- [Office Hours](#)
- [Evaluation](#)
- [Academic Integrity](#)
- [Online Study Material](#)
- [FAQ](#)



COMP 551: Applied Machine Learning - Fall 2024



Contact

Email: comp551-f24.cs@mcgill.ca

Class Times and Locations

Days: Mondays and Wednesdays

Time: 14:35-15:55

Dates: Aug. 28 to Dec. 4th

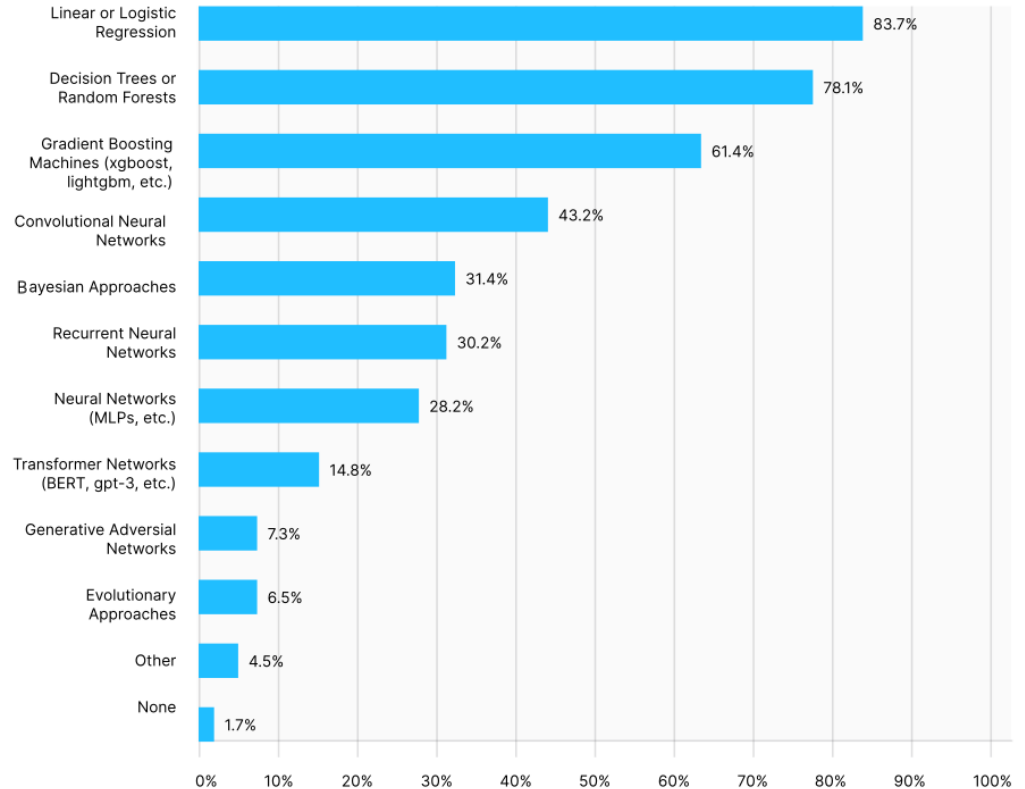
Location: McConnell 204

Class	Office Hours				
Tutorials	(TR: Trottier; MC: McConnell)				
Instructor OH					
TA OH	Monday	Tuesday	Wednesday	Thursday	Friday
8:00 - 8:30					
8:30 - 9:00					
9:00 - 9:30				Abdelrahman 9-10	
9:30 - 10:00	Huiliang Zhang 9-11			https://mcgill.zoom.us/j/87077992621	
10:00 - 10:30		Charlotte Volk 10-11			
10:30 - 11:00	https://mcgill.zoom.us/j/86993218741	TR 3104			
11:00 - 11:30	Jean-François Tremblay 11-12				
11:30 - 12:00	TR 3104				
12:00 - 12:30					
12:30 - 13:00				Elham Daneshmand 12:30-13:30	
13:00 - 13:30				https://mcgill.zoom.us/j/88103746351	
13:30 - 14:00					
14:00 - 14:30					
14:30 - 15:00	Lecture 14:30-16:00		Lecture 14:30-16:00		
15:00 - 15:30					
15:30 - 16:00	MC 204 (Overflow: Tr1080)		MC 204 (Overflow: Tr1080)		
16:00 - 16:30	Instructor OH*		Instructor OH*	Shubham Vashisth 16:30-17:30	
16:30 - 17:00				TR 3104	
17:00 - 17:30					
17:30 - 18:00		Ali Saheb Pasand 17:30-18:30		Ali Saheb Pasand 17:30-18:30	
18:00 - 18:30		https://mcgill.zoom.us/j/83365065305?pwd=8Zats1MR6jAZujr0Ob6H85qfG		https://mcgill.zoom.us/j/83365065305?pwd=8Zats1MR6jAZujr0Ob6H85qfG	
18:30 - 19:00					
19:00 - 19:30					

* Instructor OH is held by the instructor who gave the lecture. If it is Isabeau then it is in MC 205N, if Reihaneh then MC 232

Sometimes all you need is a linear regression ...

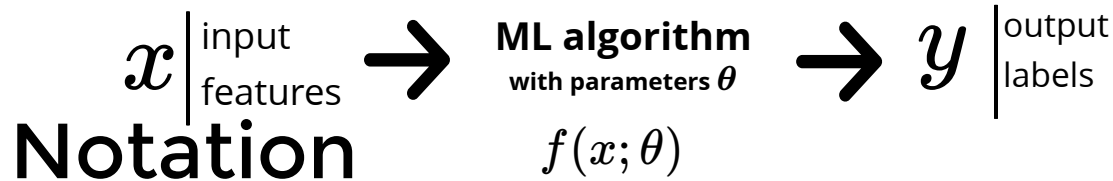
METHODS AND ALGORITHMS USAGE



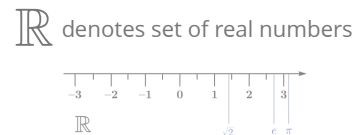
from 2020 Kaggle's survey on the state of Machine Learning and Data Science, you can read the full version [here](#)

Learning objectives

- linear model
- evaluation criteria
- how to find the best fit
- geometric interpretation
- maximum likelihood interpretation



each instance: $x \in \mathbb{R}^D$
 $y \in \mathbb{R}$



vectors are assumed to be **column vectors** $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}$ = a feature = $[x_1, x_2, \dots, x_D]^T$

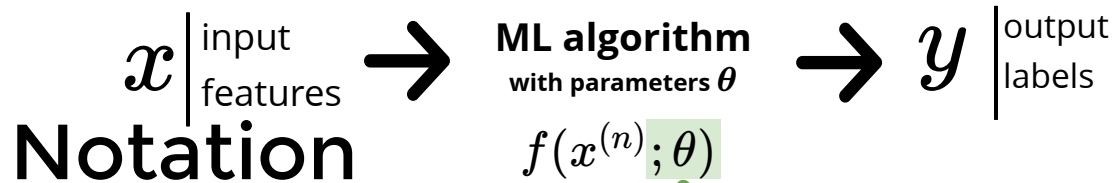
example

<tumorsize, texture, perimeter> = <18.2, 27.6, 117.5> \rightarrow growth = +2

$$x = [18.2, 27.6, 117.5]^T$$

$$y = 2$$

$$x = [x_1, x_2, x_3]^T$$



each instance: $x^{(n)} \in \mathbb{R}^D$
 $y^{(n)} \in \mathbb{R}$

instance number

$$\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^N$$

we assume **N** instances in the dataset $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^N$

each instance has **D** features indexed by **d**

for example, $x_d^{(n)} \in \mathbb{R}$ is the feature **d** of instance **n**

Notation

design matrix: concatenate all instances $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^N$
 each row is a datapoint, each column is a feature

$$X = \begin{bmatrix} x^{(1)\top} \\ x^{(2)\top} \\ \vdots \\ x^{(N)\top} \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_D^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_D^{(N)} \end{bmatrix} \in \mathbb{R}^{N \times D}$$

one instance
one feature

$$Y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix} \in \mathbb{R}^{N \times 1}$$

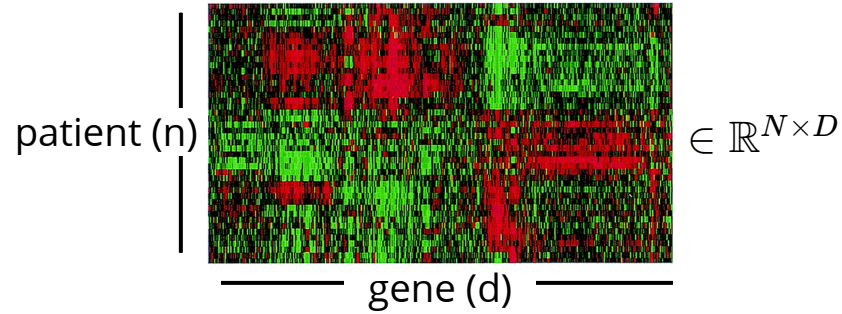
Example:

instances: 5 documents
 features: 7 words

	it	is	puppy	cat	pen	a	this
it is a puppy	1	1	1	0	0	1	0
it is a kitten	1	1	0	0	0	1	0
it is a cat	1	1	0	1	0	1	0
that is a dog and this is a pen	0	1	0	0	1	1	1
it is a matrix	1	1	0	0	0	1	0

Example:

Micro array data (X), contains gene expression levels
 labels (y) can be {cancer/no cancer classification} label for each patient, or how fast it is growing (regression)



Regression: examples

Age-estimating.

input: face
output: age

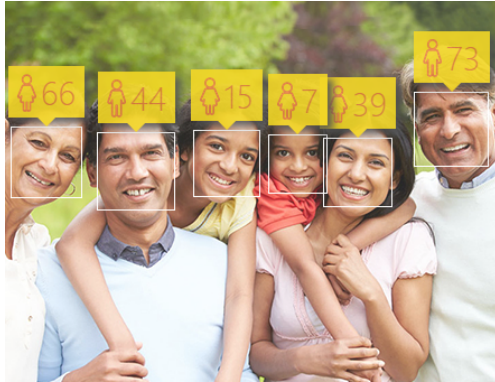


image from Microsoft
age estimator [here](#)

Protein folding.

input: sequences
output: 3D structure

instead of is it cancer? yes, no
How fast is it growing? 1.5

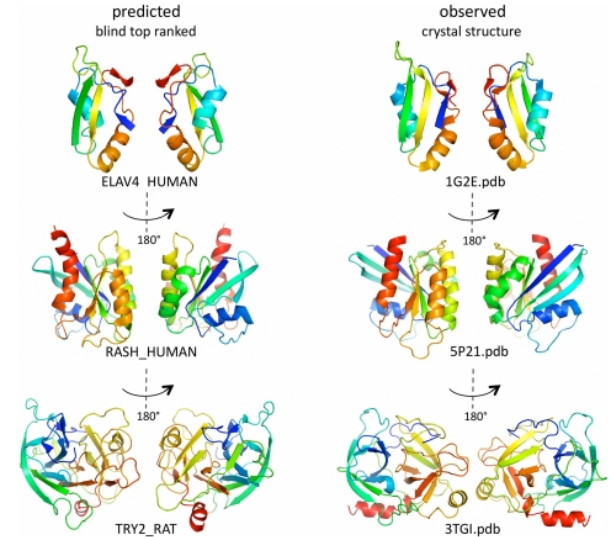


Image from Marks et al. [link](#)

Colourization.

input: gray scale image
output: colour image

Image from Zhang et al. [link](#)



Origin of Regression

Method of least squares was invented by **Legendre** and **Gauss** (1800's)

Gauss used it to predict the future location of Ceres (largest asteroid in the asteroid belt)



ocean navigation
image from wiki history of navigation



Gauss
used it

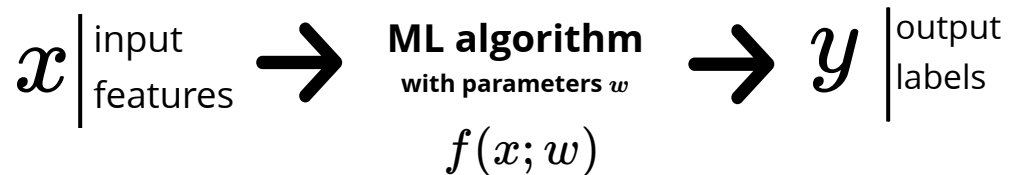


Legendre
published it



Pearson
named it regression

Linear model of regression



$$f_w(x) = w_0 + w_1 x_1 + \dots + w_D x_D$$

model parameters or weights (we also called them θ before)

$[w_0, w_1, \dots, w_D]$

bias or intercept

assuming a scalar output $f_w : \mathbb{R}^D \rightarrow \mathbb{R}$

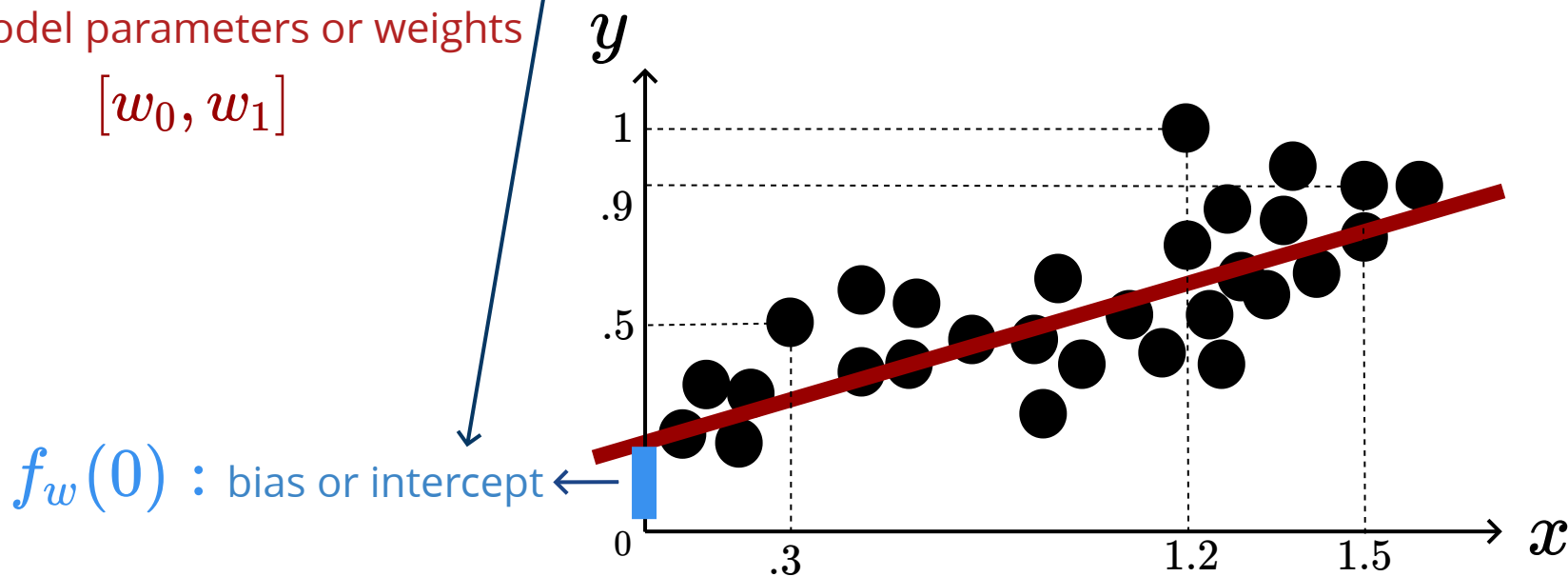
will generalize to a vector later

Linear model of regression: example $D = 1$

$$f_w(x) = w_0 + w_1 x_1$$

model parameters or weights
 $[w_0, w_1]$

forms a line in 1 dimension



$$X = \begin{bmatrix} .3 \\ 1.2 \\ 1.5 \\ \vdots \\ \vdots \\ \vdots \end{bmatrix}, Y = \begin{bmatrix} .5 \\ .9 \\ 1 \\ \vdots \\ \vdots \\ \vdots \end{bmatrix}$$

Linear model of regression

$$f_w(x) = w_0 + w_1 x_1 + \dots + w_D x_D$$

model parameters or weights

bias or intercept

simplification

concatenate a 1 to x $\longrightarrow x = [1, x_1, \dots, x_D]^\top$

$$f_w(x) = w^\top x$$

$$w = [w_0, w_1, \dots, w_D]^\top$$

Linear regression: Objective

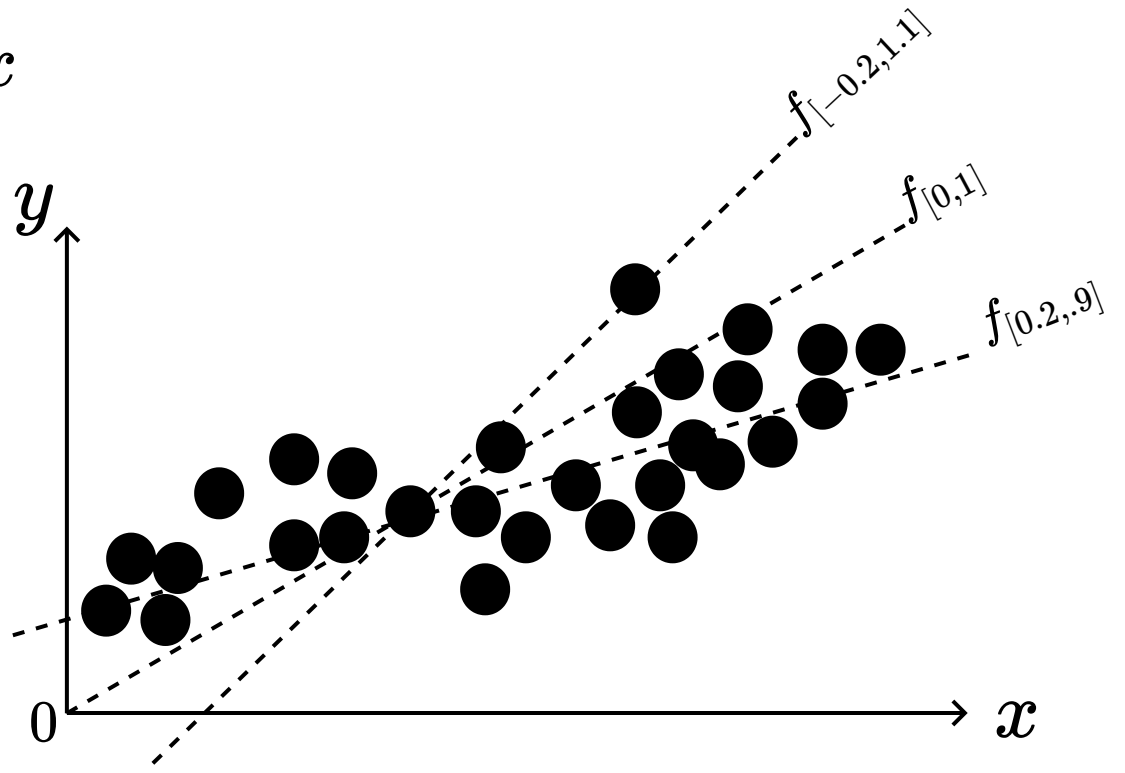
objective: find parameters to fit the data

model: $f_w(x) = w^\top x$

example $D = 1$

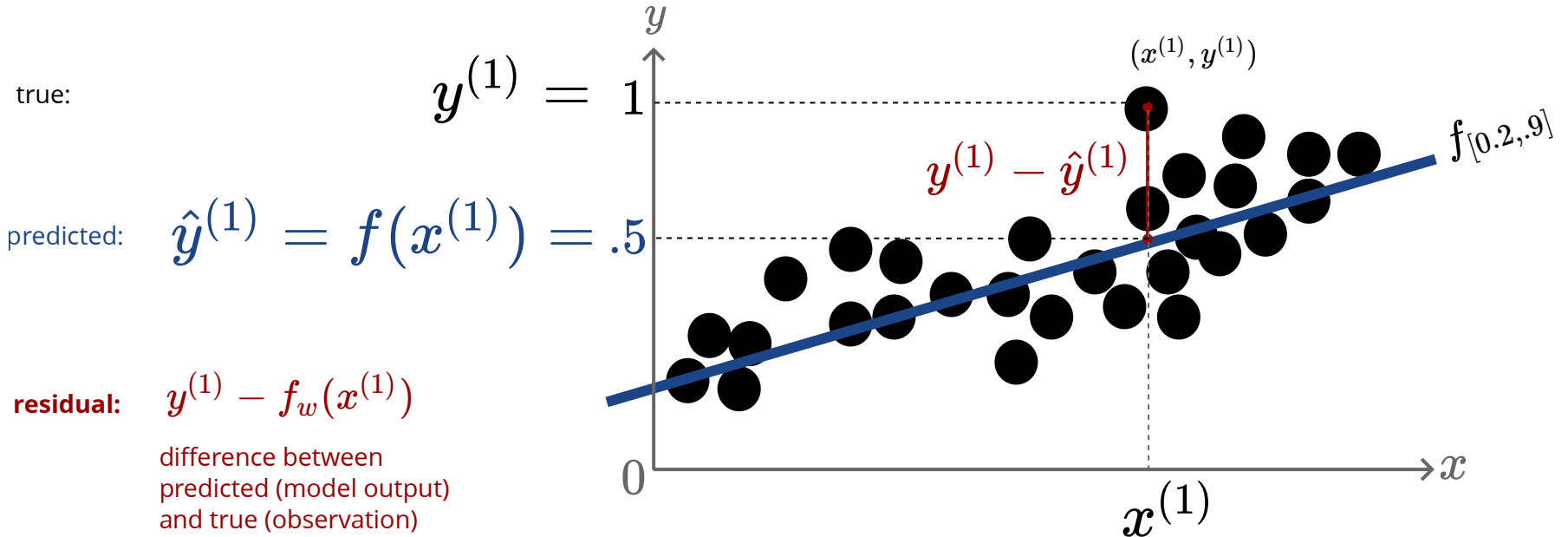
$w = [w_0, w_1]$

Which line is better?



Linear regression: Objective

objective: find parameters to fit the data



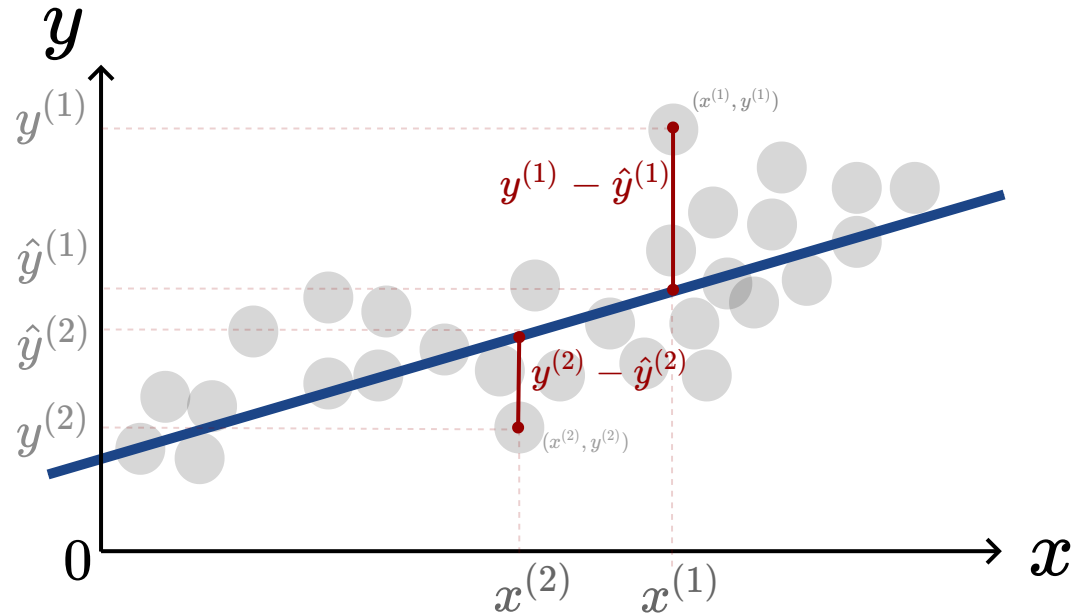
Linear regression: Objective

objective: find parameters to fit the data

how to consider all observations? sum all residuals?

square error **loss**
(a.k.a. **L2** loss)

$$L(y, \hat{y}) \triangleq (y - \hat{y})^2$$



Linear regression: **cost function**

objective: find parameters to **fit the data**

$$f_w(x^{(n)}) \approx y^{(n)} \quad x^{(n)}, y^{(n)} \quad \forall n$$

minimize a measure of difference between $\hat{y}^{(n)} = f_w(x^{(n)})$ and $y^{(n)}$

square error **loss** (a.k.a. **L2** loss) $L(y, \hat{y}) \triangleq \frac{1}{2}(y - \hat{y})^2$

for a single instance (a function of labels)

for future convenience

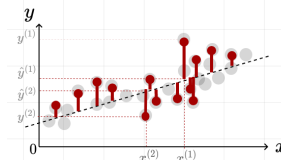
versus

for the whole dataset

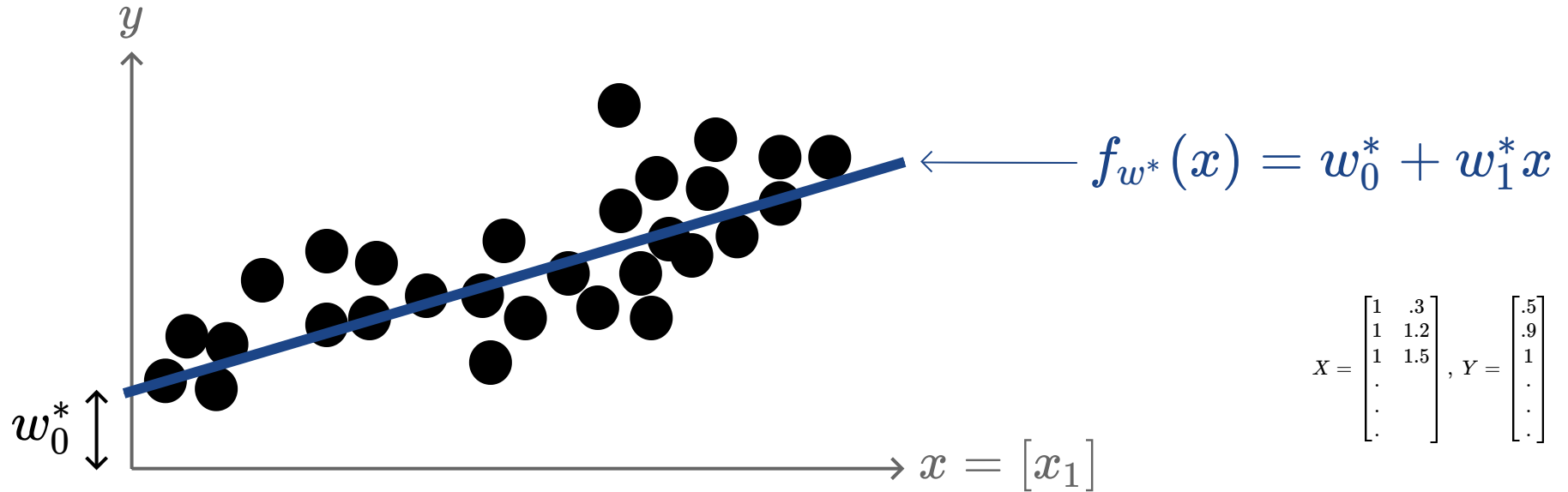
sum of squared errors **cost function**

$$J(w) = \frac{1}{2} \sum_{n=1}^N \left(y^{(n)} - w^\top x^{(n)} \right)^2$$

$$w^* = \arg \min_w J(w)$$

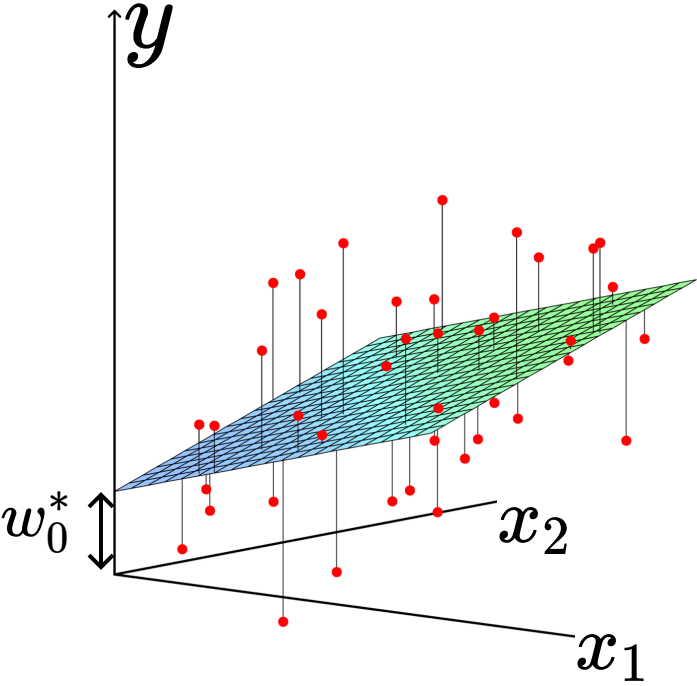


Example ($D = 1$) + bias ($D=2$)!



Linear Least Squares solution: $w^* = \arg \min_w \sum_n \frac{1}{2} \left(y^{(n)} - w^T x^{(n)} \right)^2$

Example (D=2) +bias (D=3)!



$$f_{w^*}(x) = w_0^* + w_1^*x_1 + w_2^*x_2$$

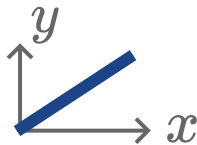
$$X = \begin{bmatrix} 1 & .3 & .5 \\ 1 & 1.2 & 1.6 \\ 1 & 1.5 & 1.2 \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \end{bmatrix}, Y = \begin{bmatrix} .5 \\ .9 \\ 1 \\ \vdots \\ \vdots \end{bmatrix}$$

Linear Least Squares

$$w^* = \arg \min_w \sum_n \left(y^{(n)} - w^T x^{(n)} \right)^2$$

Minimizing the cost

Simple case: $D = 1$ (no intercept)



model: $f_w(x) = wx$
both scalar

cost function $J(w) = \frac{1}{2} \sum_n (y^{(n)} - wx^{(n)})^2$

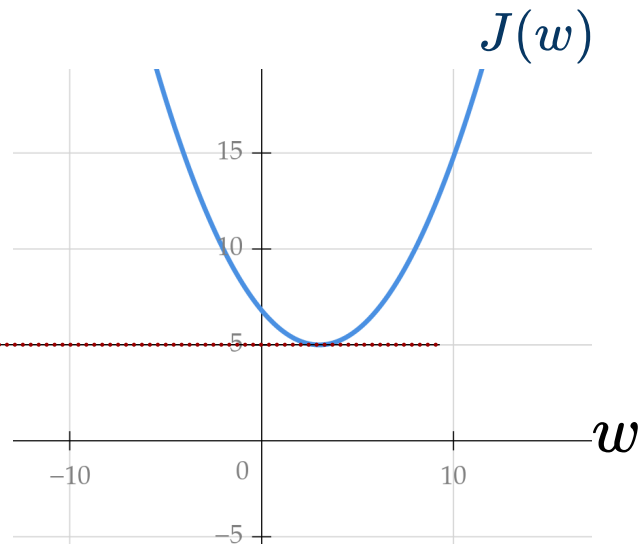
derivative: $\frac{dJ}{dw} = \sum_n x^{(n)} (wx^{(n)} - y^{(n)})$

set to 0: $0 = w \sum_n x^{(n)} x^{(n)} - \sum_n x^{(n)} y^{(n)}$

setting the derivative to zero $w^* = \frac{\sum_n x^{(n)} y^{(n)}}{\sum_n x^{(n)2}}$

global minimum because the cost function is smooth and *convex*

more on convexity later

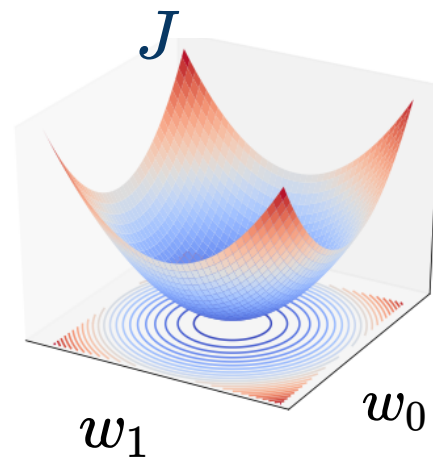


Minimizing the cost

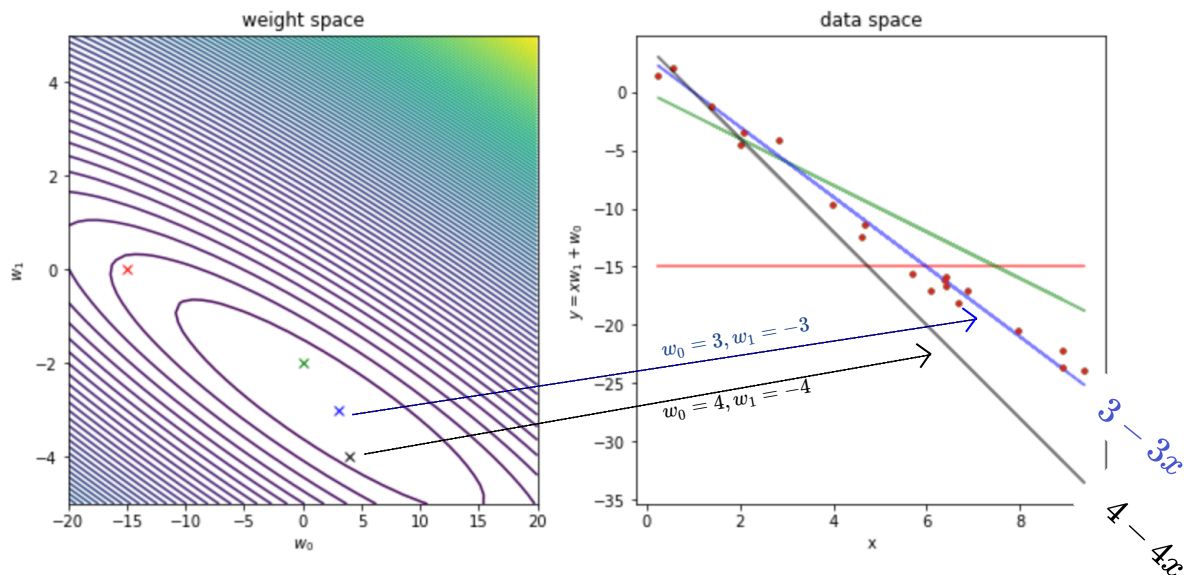
$D = 1$ (with intercept)

model: $f_w(x) = w_0 + w_1 x$

cost: a multivariate function $J(w_0, w_1)$



the cost function is a smooth function of w
find minimum by setting partial derivatives to zero



Minimizing the cost

for a multivariate function $J(w_0, w_1)$

partial derivatives instead of derivative

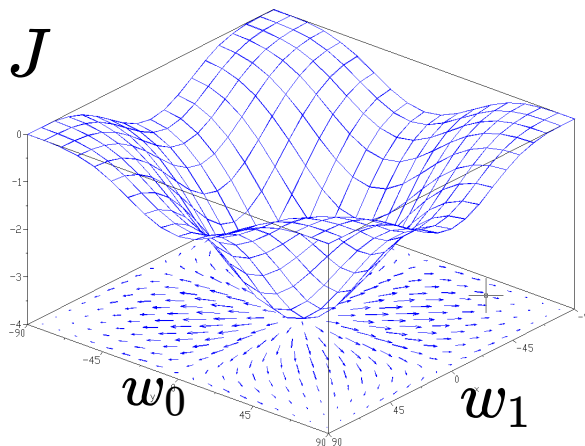
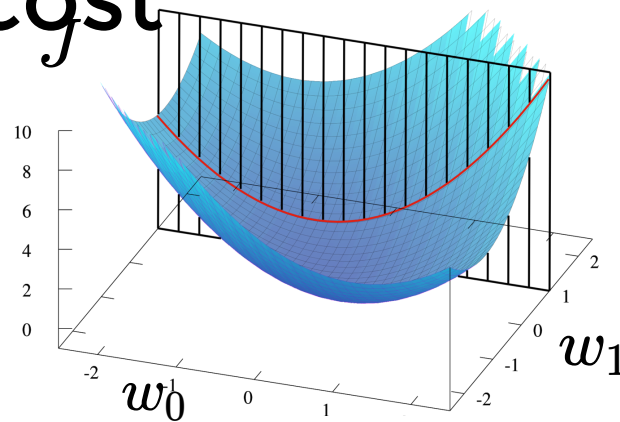
= derivative when other vars. are fixed

$$\frac{\partial}{\partial w_0} J(w_0, w_1) \triangleq \lim_{\epsilon \rightarrow 0} \frac{J(w_0 + \epsilon, w_1) - J(w_0, w_1)}{\epsilon}$$

critical point: all partial derivatives are zero

gradient: vector of all partial derivatives

$$\nabla J(w) = \left[\frac{\partial}{\partial w_1} J(w), \dots, \frac{\partial}{\partial w_D} J(w) \right]^\top$$



Minimizing the cost

for general case (any D)

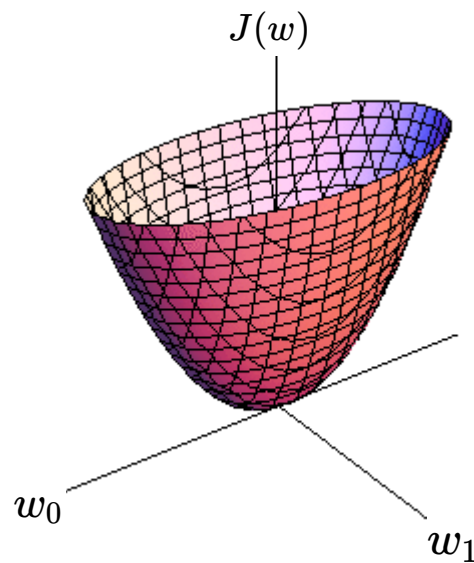
find the critical point by setting $\frac{\partial}{\partial w_d} J(w) = 0$

$$\frac{\partial}{\partial w_d} \sum_n \frac{1}{2} (y^{(n)} - f_w(x^{(n)}))^2 = 0$$

using **chain rule**: $\frac{\partial J}{\partial w_d} = \frac{dJ}{df_w} \frac{\partial f_w}{\partial w_d}$

we get $\sum_n (w^\top x^{(n)} - y^{(n)}) x_d^{(n)} = 0 \quad \forall d \in \{1, \dots, D\}$

D equations with D unknowns



cost is a smooth and convex function of w

we are ignoring the bias term here, with the bias term, it would be D+1 equations and D+1 unknown for d in [0,D]

Linear regression: Matrix form

instead of $\hat{y}^{(n)} \in \mathbb{R} = \mathbf{w}^\top \mathbf{x}^{(n)}$
 $1 \times D \quad D \times 1$

Note: D is in fact dimensions of the input +1 due to the simplification and adding the bias/intercept term

use **design matrix** to write $\hat{\mathbf{y}} = \mathbf{X} \mathbf{w}$
 $N \times 1 \quad N \times D \quad D \times 1$

$$\hat{y}^{(1)} = w_0 + x_1^{(1)} w_1 + x_2^{(1)} w_2 + \dots + x_D^{(1)} w_D$$
$$\hat{\mathbf{Y}} = \begin{bmatrix} \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \vdots \\ \hat{y}^{(N)} \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \dots & x_D^{(1)} \\ 1 & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & x_2^{(N)} & \dots & x_D^{(N)} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix}$$

Linear least squares

$$\arg \min_w \frac{1}{2} \| \mathbf{y} - \mathbf{X} \mathbf{w} \|_2^2 = \frac{1}{2} (\mathbf{y} - \mathbf{X} \mathbf{w})^\top (\mathbf{y} - \mathbf{X} \mathbf{w})$$

squared L2 norm of the **residual** vector

Minimizing the cost: **Matrix form**

Linear least squares

$$J(w) = \frac{1}{2} \|y - Xw\|^2 = \frac{1}{2} (y - Xw)^T (y - Xw)$$

$$y^T Xw = w^T X^T y$$

$$\frac{\partial J(w)}{\partial w} = \frac{\partial}{\partial w} [y^T y + w^T X^T Xw - 2y^T Xw]$$

$$\frac{\partial Xw}{\partial w} = X^T$$

Using matrix differentiation

$$\frac{\partial w^T Xw}{\partial w} = 2Xw$$

$$\frac{\partial J(w)}{\partial w} = 0 + X^T Xw - X^T y = X^T (Xw - y)$$

Minimizing the cost: Matrix form

Linear least squares

$$\hat{y} = Xw$$

$N \times 1$ $N \times D$ $D \times 1$

$$J(w) = \frac{1}{2} \|y - Xw\|^2 = \frac{1}{2} (y - Xw)^T (y - Xw)$$

$$\frac{\partial J(w)}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_i} [y^T y + w^T X^T X w - 2y^T X w]$$

Minimizing the cost: Matrix form

Linear least squares

$$\hat{y} = Xw$$

$N \times 1$ $N \times D$ $D \times 1$

$$J(w) = \frac{1}{2} \|y - Xw\|^2 = \frac{1}{2} (y - Xw)^T (y - Xw)$$

$$\frac{\partial J(w)}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_i} [y^T y + w^T X^T X w - 2y^T X w]$$

$$\frac{\partial J(w)}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_i} [\sum_j y_{j1} y_{j1} + \sum_{j,k,m} w_{j1} X_{kj} X_{km} w_{m1} - \sum_{j,k} 2y_{j1} X_{jk} w_{k1}]$$

Minimizing the cost: Matrix form

Linear least squares

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$$

$N \times 1$ $N \times D$ $D \times 1$

$$J(\mathbf{w}) = \frac{1}{2} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 = \frac{1}{2} (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

$$\frac{\partial J(\mathbf{w})}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_i} [\mathbf{y}^T \mathbf{y} + \mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w} - 2\mathbf{y}^T \mathbf{X} \mathbf{w}]$$

$$\frac{\partial J(\mathbf{w})}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_i} \left[\sum_j y_{j1} y_{j1} + \sum_{j,k,m} w_{j1} X_{kj} X_{km} w_{m1} - \sum_{j,k} 2y_{j1} X_{jk} w_{k1} \right]$$

✦✦ Einstein notation: implicit sum on repeated indices ✦✦ $(AB)_{ij} = A_{ik} B_{kj}$

$$\frac{\partial J(\mathbf{w})}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_{i1}} [y_{j1} y_{j1} + w_{j1} X_{kj} X_{km} w_{m1} - 2y_{j1} X_{jk} w_{k1}]$$

Minimizing the cost: Matrix form

Linear least squares

$$\hat{y} = Xw$$

$N \times 1$ $N \times D$ $D \times 1$

$$J(w) = \frac{1}{2} \|y - Xw\|^2 = \frac{1}{2} (y - Xw)^T (y - Xw)$$

$$\frac{\partial J(w)}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_i} [y^T y + w^T X^T X w - 2y^T X w]$$

$$\begin{aligned} \frac{\partial J(w)}{\partial w_i} &= \frac{1}{2} \frac{\partial}{\partial w_{i1}} \left[\sum_j y_{j1} y_{j1} + \sum_{j,k,m} w_{j1} X_{kj} X_{km} w_{m1} - \sum_{j,k} 2y_{j1} X_{jk} w_{k1} \right] \\ &= \frac{1}{2} \left[\sum_{j,k,m} \delta_{ij} X_{kj} X_{km} w_{m1} + \sum_{j,k,m} w_{j1} X_{kj} X_{km} \delta_{mi} - \sum_{j,k} 2y_{j1} X_{jk} \delta_{ki} \right] \end{aligned}$$

Minimizing the cost: Matrix form

Linear least squares

$$\hat{y} = Xw$$

$N \times 1$ $N \times D$ $D \times 1$

$$J(w) = \frac{1}{2} \|y - Xw\|^2 = \frac{1}{2} (y - Xw)^T (y - Xw)$$

$$(AB)_{ij} = \sum_k A_{ik} B_{kj}$$

$$\begin{aligned} \frac{\partial J(w)}{\partial w_i} &= \frac{1}{2} \frac{\partial}{\partial w_{i1}} \left[\sum_j y_{j1} y_{j1} + \sum_{j,k,m} w_{j1} X_{kj} X_{km} w_{m1} - \sum_{j,k} 2y_{j1} X_{jk} w_{k1} \right] \\ &= \frac{1}{2} \left[\sum_{j,k,m} \delta_{ij} X_{kj} X_{km} w_{m1} + \sum_{j,k,m} w_{j1} X_{kj} X_{km} \delta_{mi} - \sum_{j,k} 2y_{j1} X_{jk} \delta_{ki} \right] \\ &= \frac{1}{2} \left[\sum_{k,m} X_{ki} X_{km} w_{m1} + \sum_{j,k} w_{j1} X_{kj} X_{ki} - \sum_j 2y_{j1} X_{ji} \right] = \left(\frac{2}{2} X^T (Xw - y) \right)_i \end{aligned}$$

Minimizing the cost: Matrix form

Linear least squares

$$\hat{y} = Xw$$

$N \times 1$ $N \times D$ $D \times 1$

$$J(w) = \frac{1}{2} \|y - Xw\|^2 = \frac{1}{2} (y - Xw)^T (y - Xw)$$

$$\frac{\partial J(w)}{\partial w_i} = \frac{1}{2} \frac{\partial}{\partial w_i} [y^T y + w^T X^T X w - 2y^T X w]$$

$$(AB)_{ij} = \sum_k A_{ik} B_{kj}$$

$$= \frac{1}{2} [\sum_{j,k,m} \delta_{ij} X_{kj} X_{km} w_{m1} + \sum_{j,k,m} w_{j1} X_{kj} X_{km} \delta_{mi} - \sum_{j,k} 2y_{j1} X_{jk} \delta_{ki}]$$

$$= \frac{1}{2} [\sum_{k,m} X_{ki} X_{km} w_{m1} + \sum_{j,k} w_{j1} X_{kj} X_{ki} - \sum_j 2y_{j1} X_{ji}] = \left(\frac{\partial}{\partial w} J(w) \right)_i$$

$$0 = \frac{\partial J(w)}{\partial w} = X^T (Xw - y)$$

Minimizing the cost: Matrix form

Linear least squares

$$\hat{y} = Xw$$

$N \times 1$ $N \times D$ $D \times 1$

$$0 = \frac{\partial J(w)}{\partial w} = X^T (Xw - y)$$

$$0 = X^T Xw - X^T y \implies X^T Xw = X^T y$$
$$\implies w = (X^T X)^{-1} X^T y$$

Closed form solution

$$\overbrace{X^\top}^{D \times N} \overbrace{(y - Xw)}^{N \times 1} = \vec{0} \quad \text{matrix form (using the design matrix)}$$

$$X^\top X w = X^\top y \quad \text{system of } D \text{ linear equations (} Aw = b \text{)}$$

each row enforces one of D equations

$$w^* = \underbrace{(X^\top X)^{-1}}_{D \times D} \underbrace{X^\top}_{D \times N} \underbrace{y}_{N \times 1}$$

pseudo-inverse of X

closed form solution

similar to non-matrix form: optimal weights w^* satisfy

$$\sum_n (y^{(n)} - w^\top x^{(n)}) x_d^{(n)} = 0 \quad \forall d$$

D equations with D unknowns

Uniqueness of the solution

we can get a closed form solution!

$$w^* = (X^T X)^{-1} X^T y$$

unless $D \geq N$

or when the $X^T X$ **matrix** is not invertible

this matrix is not invertible when some of eigenvalues are zero!

that is, if features are completely correlated

... or more generally if features are **not linearly independent**

examples having a binary feature x_1 as well as its negation $x_2 = (1 - x_1)$

Time complexity

$$w^* = \overbrace{(X^T X)^{-1}}^{D \times D} \overbrace{X^T y}^{D \times N \quad N \times 1}$$

$\mathcal{O}(D^3)$ matrix inversion
 $\mathcal{O}(ND)$ D elements, each using N ops.
 $\mathcal{O}(D^2 N)$ $D \times D$ elements, each requiring N multiplications

total complexity for is $\mathcal{O}(ND^2 + D^3)$ which becomes $\mathcal{O}(ND^2)$ for $N > D$

in practice we don't directly use matrix inversion (unstable)

however, other more stable solutions (e.g., Gaussian elimination) have similar complexity

Multiple targets

instead of $\mathbf{y} \in \mathbb{R}^N$ we have $\mathbf{Y} \in \mathbb{R}^{N \times D'}$

a different weight vectors for each target

each column of \mathbf{Y} is associated with a column of \mathbf{W}

$$\hat{\mathbf{Y}} = \mathbf{X}\mathbf{W}$$

$N \times D'$ $N \times D$ $D \times D'$

$$\mathbf{W}^* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$$

$D \times D$ $D \times N$ $N \times D'$

$$\hat{\mathbf{Y}} = \begin{bmatrix} \hat{y}_1^{(1)} & \hat{y}_2^{(1)} \\ \hat{y}_1^{(2)} & \hat{y}_2^{(2)} \\ \vdots & \vdots \\ \hat{y}_1^{(N)} & \hat{y}_2^{(N)} \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_D^{(1)} \\ 1 & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & x_2^{(N)} & \cdots & x_D^{(N)} \end{bmatrix} \begin{bmatrix} w_{0,1} & w_{0,2} \\ w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \\ \vdots & \vdots \\ w_{D,1} & w_{D,2} \end{bmatrix}$$

$$\hat{y}_1^{(1)} = w_{0,1} + x_1^{(1)} w_{1,1} + x_2^{(1)} w_{2,1} + \cdots + x_D^{(1)} w_{D,1}$$

$$\hat{y}_2^{(1)} = w_{0,2} + x_1^{(1)} w_{1,2} + x_2^{(1)} w_{2,2} + \cdots + x_D^{(1)} w_{D,2}$$

Fitting non-linear data

so far we learned a linear function $f_w = \sum_d w_d x_d$
sometimes this may be too simplistic

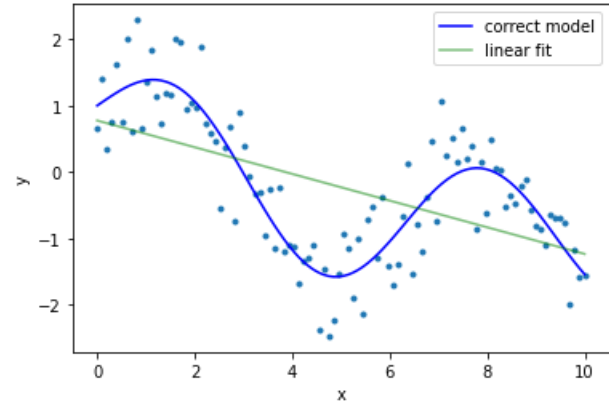
example

Synthetic data when we generated data
from a function

$$y^* = \sin(x) + \cos(\sqrt{x})$$

$$\mathcal{D} = \{(x^{(n)}, y^*(x^{(n)})) + \epsilon\}_{n=1}^N$$

small
noise



we see linear fit is not close to correct model that
the data is generated from, can we get a better fit?

idea

create new more useful features out of initial set of given features

e.g., $x_1^2, x_1 x_2, \log(x)$,

how about $x_1 + 2x_3$?

Nonlinear basis functions

so far we learned a linear function $f_w = \sum_d w_d x_d$

let's denote the set of all features by $\phi_d(x) \forall d$

the problem of linear regression doesn't change

$$f_w = \sum_d w_d \phi_d(x)$$

$\phi_d(x)$ is the new x

solution simply becomes $(\Phi^\top \Phi)w^* = \Phi^\top y$

replacing X with Φ

a (nonlinear) feature

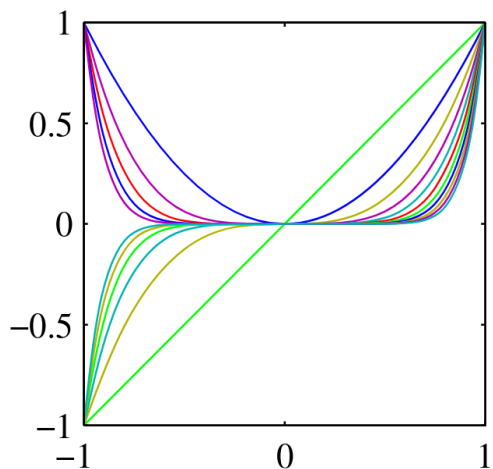
$$\Phi = \begin{bmatrix} \phi_1(x^{(1)}), & \phi_2(x^{(1)}), & \cdots, & \phi_D(x^{(1)}) \\ \phi_1(x^{(2)}), & \phi_2(x^{(2)}), & \cdots, & \phi_D(x^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x^{(N)}), & \phi_2(x^{(N)}), & \cdots, & \phi_D(x^{(N)}) \end{bmatrix}$$

one instance

Nonlinear basis functions

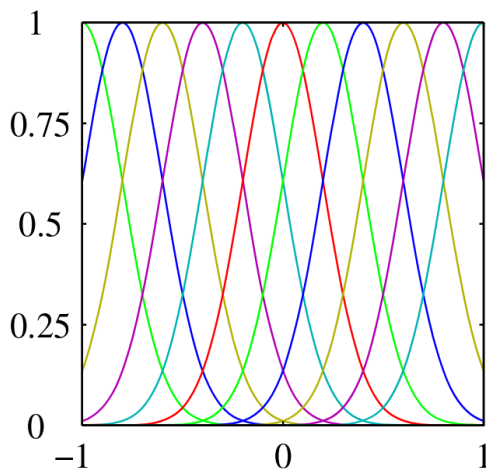
example

original input is scalar $x \in \mathbb{R}$



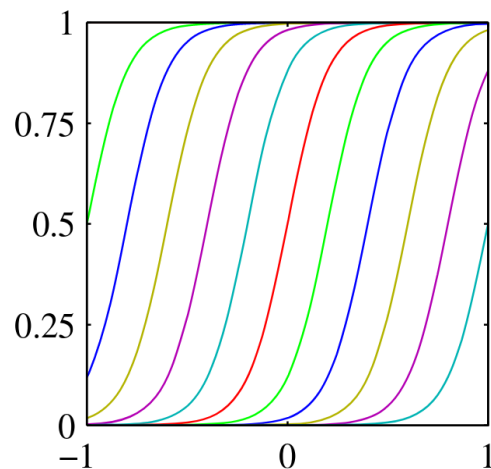
polynomial bases

$$\phi_k(x) = x^k$$



Gaussian bases

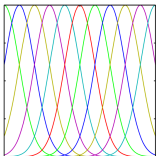
$$\phi_k(x) = e^{-\frac{(x-\mu_k)^2}{s^2}}$$



Sigmoid bases

$$\phi_k(x) = \frac{1}{1+e^{-\frac{x-\mu_k}{s}}}$$

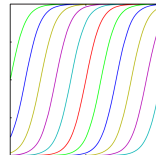
Linear regression with nonlinear bases: **example**



Gaussian bases

$$\phi_k(x) = e^{-\frac{(x-\mu_k)^2}{s^2}}$$

we are using a fixed standard deviation of $s=1$

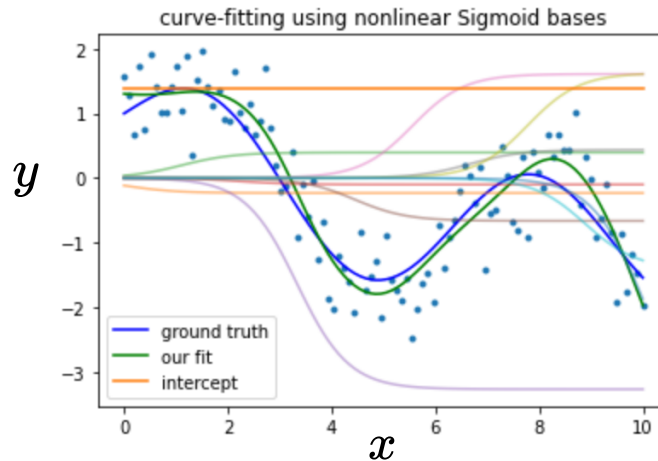
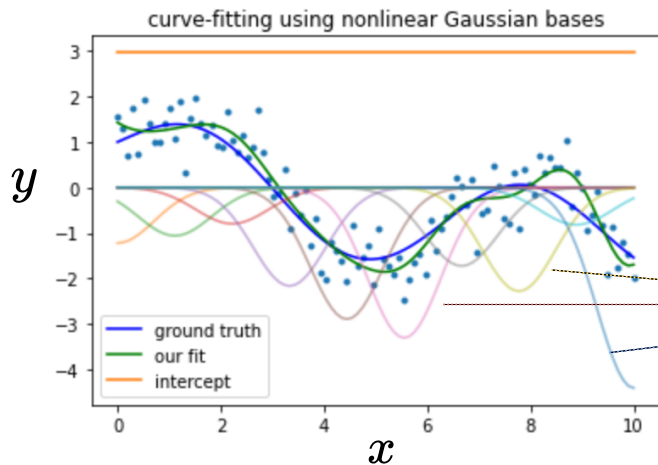


Sigmoid bases

$$\phi_k(x) = \frac{1}{1+e^{-\frac{x-\mu_k}{s}}}$$

we are using a fixed standard deviation of $s=1$

$$\hat{y}^{(n)} = w_0 + \sum_k w_k \phi_k(x)$$



Probabilistic interpretation

idea

given the dataset $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \dots, (x^{(N)}, y^{(N)})\}$

learn a probabilistic model $p(y|x; w)$

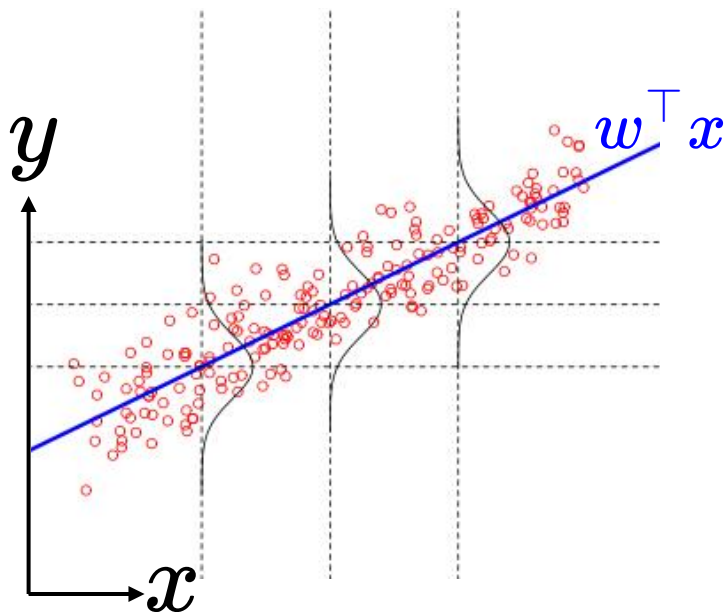


image from [here](#)

consider $p(y|x; w)$ with the following form

$$p_w(y | x) = \mathcal{N}(y | w^\top x, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-w^\top x)^2}{2\sigma^2}}$$

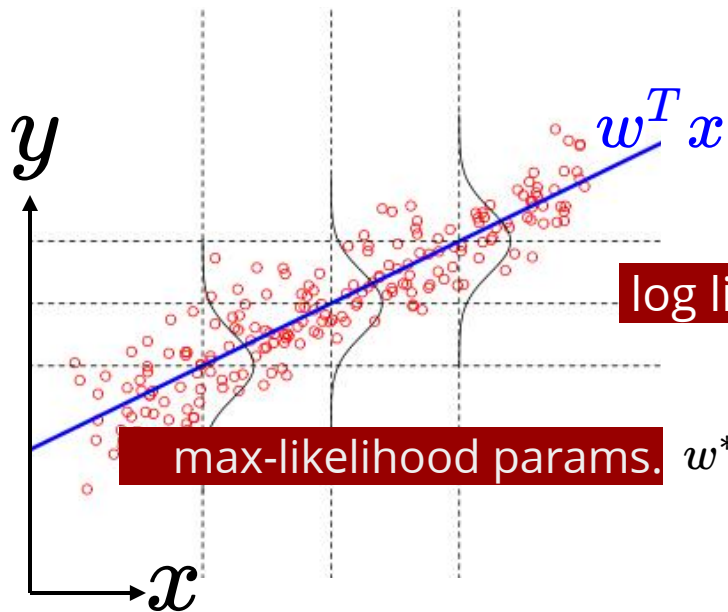
assume a fixed variance, say $\sigma^2 = 1$

Q: how to fit the model?

A: maximize the conditional likelihood!

Maximum likelihood & linear regression

cond. probability $p(y | x; w) = \mathcal{N}(y | w^\top x, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-w^\top x)^2}{2\sigma^2}}$



likelihood $L(w) = \prod_{n=1}^N p(y^{(n)} | x^{(n)}; w)$

log likelihood $\ell(w) = \sum_n -\frac{1}{2\sigma^2} (y^{(n)} - w^\top x^{(n)})^2 + \text{constants}$

max-likelihood params. $w^* = \arg \max_w \ell(w) = \arg \min_w \frac{1}{2} \sum_n (y^{(n)} - w^\top x^{(n)})^2$
linear least squares!

image from [here](#)

whenever we use square loss, we are assuming Gaussian noise!

Summary

linear regression:

- models targets as a **linear function of features**
- fit the model by **minimizing the sum of squared errors**
- has a **direct solution** with $\mathcal{O}(ND^2 + D^3)$ complexity
- probabilistic interpretation

we can build more expressive models:

- using any number of **non-linear features**