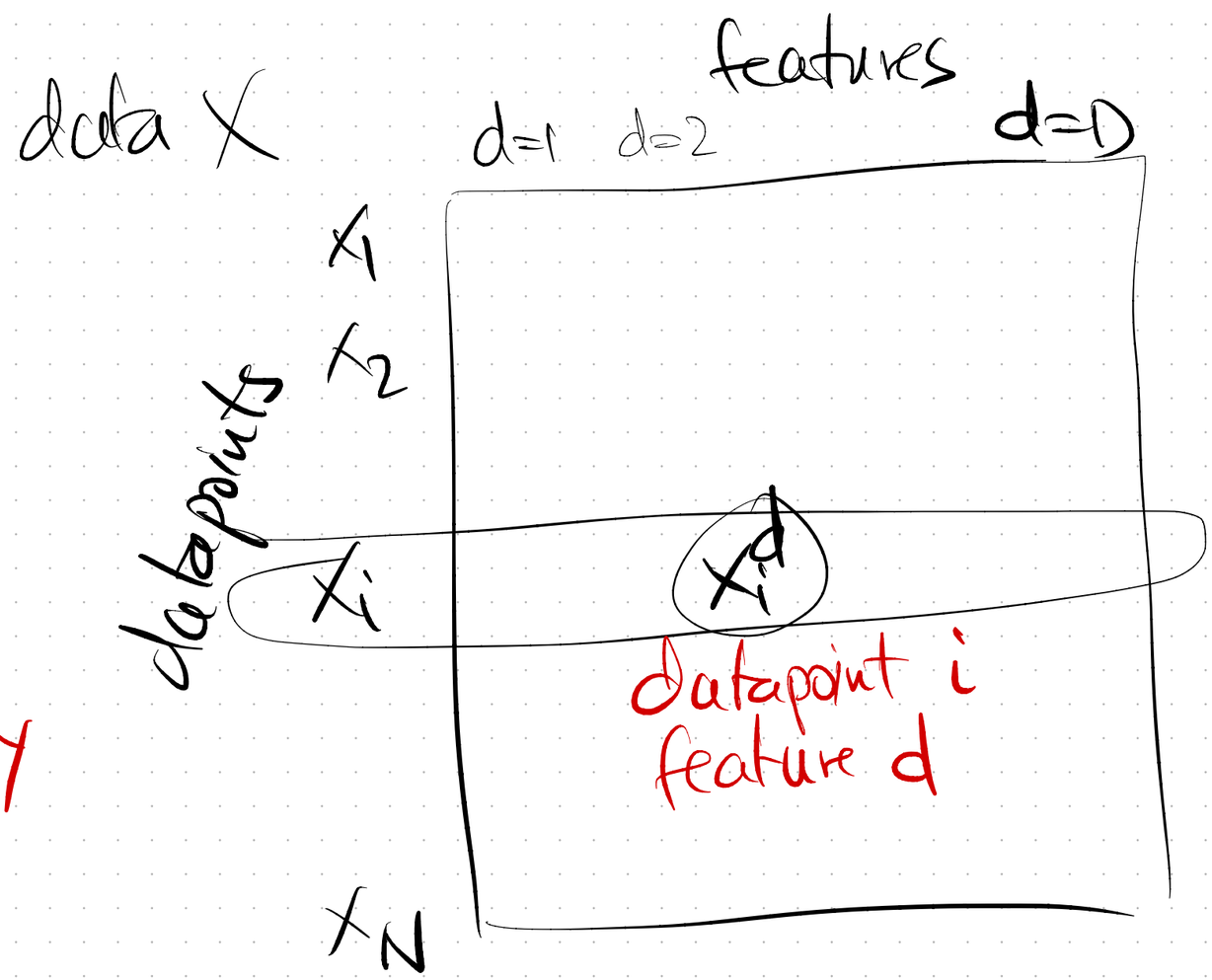
Lecture 5|21-23

- Gradient Descent
- Linear Regression with GD
- Logistic Regression
- HW2A (released later today) due 6|3
- ROC curve ≃ AUC geometrically
- Perceptron w/GD or geometry



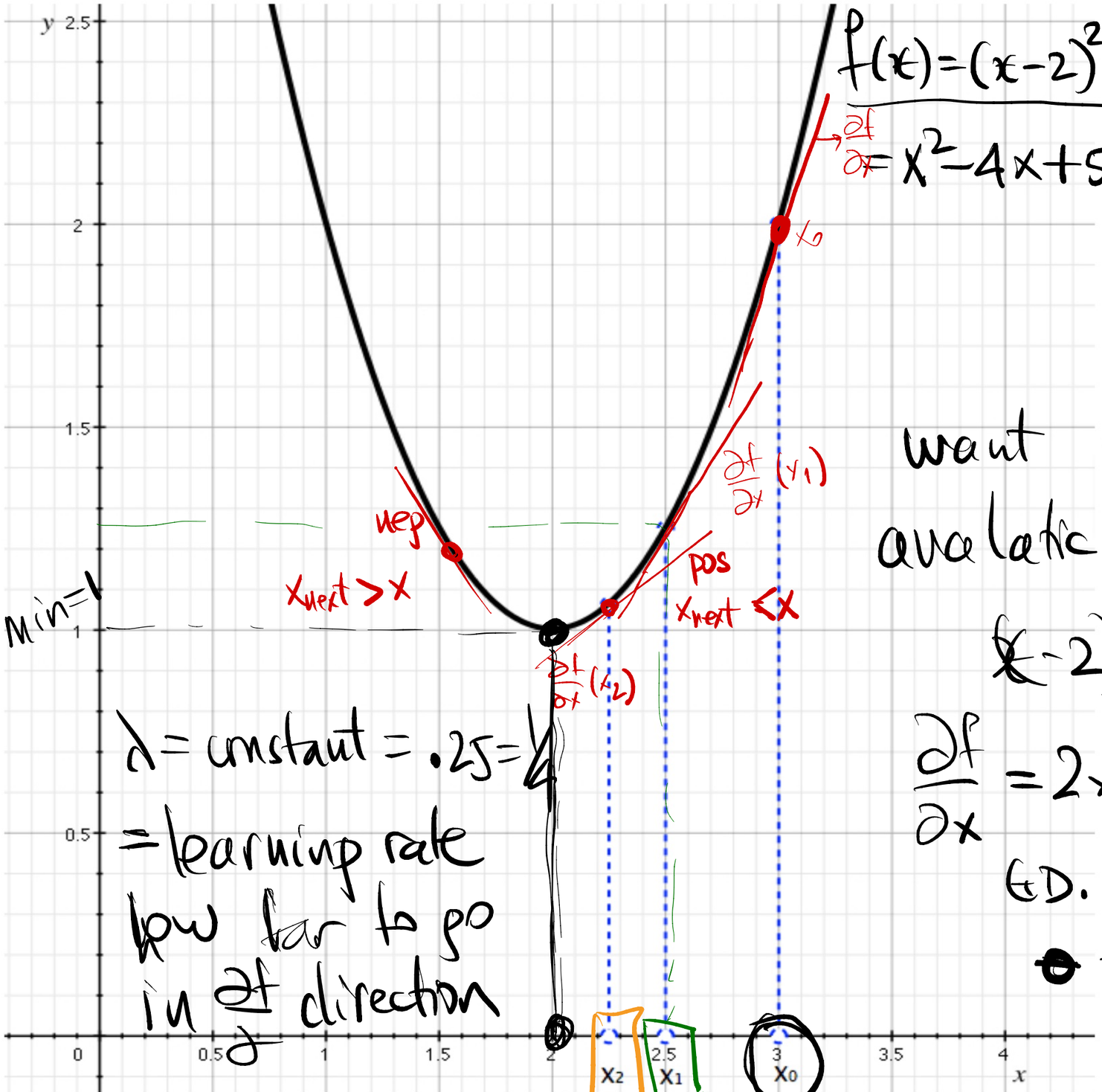


Figure 1: Gradient descent iterations

GD: iteratively find min (or max) of $f(x)$ by moving x in direction of gradient = differential.

want $\min_x f(x)$ $\operatorname{argmin} = x_{min}$

analytic solution $x_{min}=2$

$(x-2)^2+1 \geq 1$ equality if $x_{min}=2$

$\frac{\partial f}{\partial x}=2x-4$ gradient. Start at $x_0=3$

GD. iterations:

- $x_1 = x_0 - \lambda \frac{\partial f}{\partial x}(x_0)$

$= 3 - \frac{1}{4}(2\cdot 3-4) = 2.5$

- $x_2 = x_1 - \lambda \frac{\partial f}{\partial x}(x_1) = 2.5 - \frac{1}{4}(3 \times 2.5 - 4) = 2.25$

- ⋮

- i want $(x_1, x_2, x_3 \ldots) \rightarrow x_{min} = 2$

- What if I jump to far on the other side? $x_m < 2$

  $\frac{\partial f}{\partial x}(x_m) = 2 \cdot x_m - 4 < 0$ negative

  update $x_{m+1} = x_m - \frac{\partial f}{\partial x}(x_m)$ increasing $x_{m+1} > x_m$

  its moving x_{next} in right direction

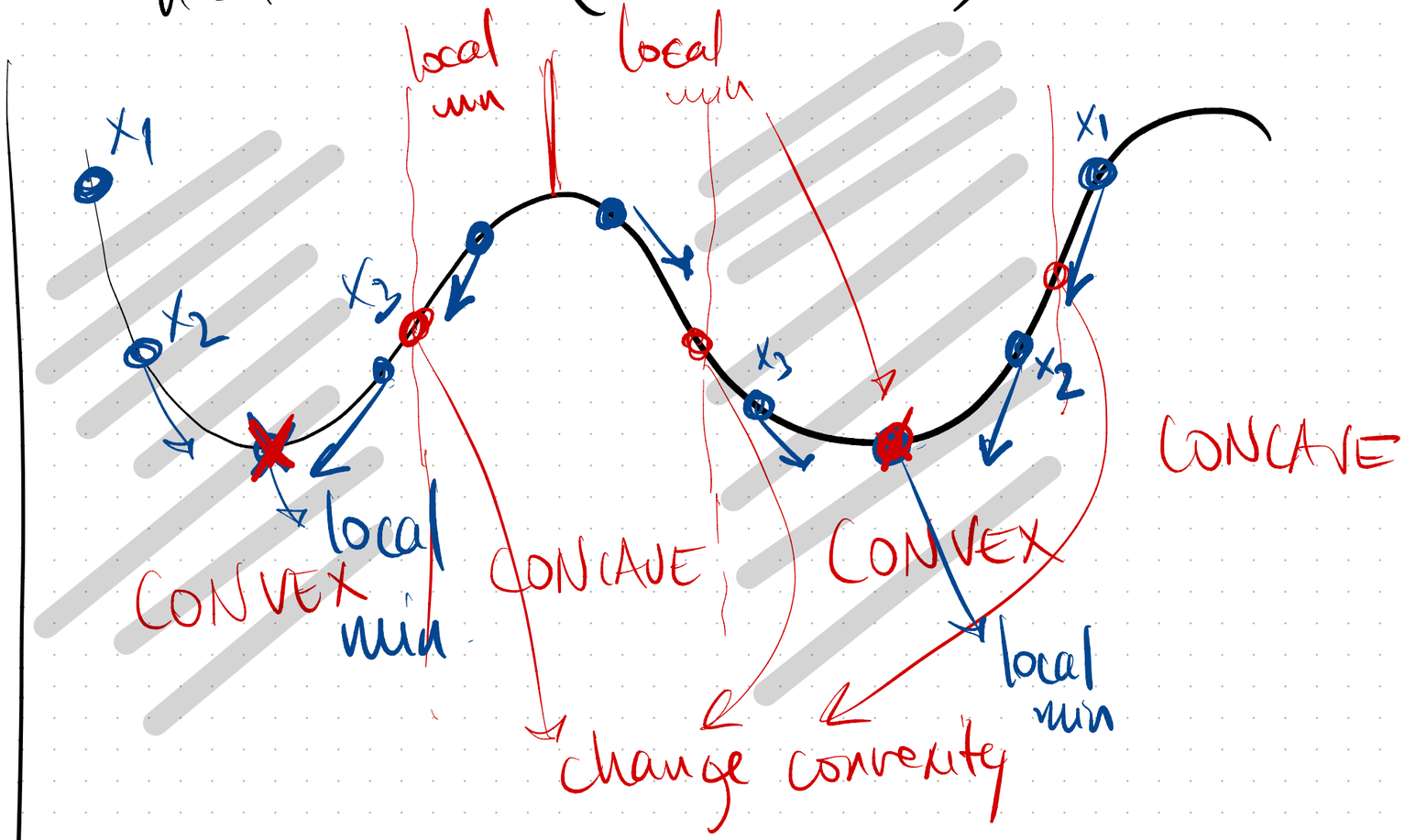
- what if $x_m = x_{min}$?

  $\frac{\partial f}{\partial x}(x_{min}) = 0 \Rightarrow$ update $x_{m+1} = x_m - \frac{\partial f}{\partial x}(x_m)$

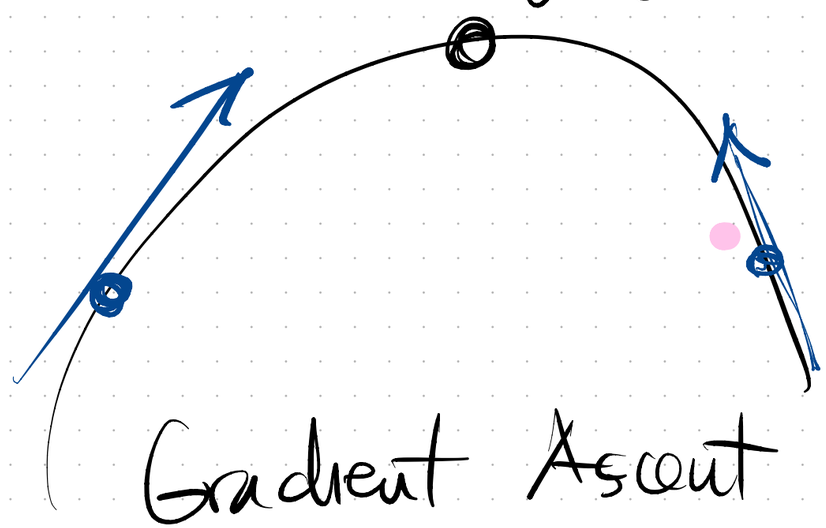
  $= x_m$ (no change)

  GD stays here.

Non convex (all the time) BUMPY



CONCAVE Region Max



Gradient Ascent

$$x_{new} = x + \lambda \frac{\partial f}{\partial x}(x)$$

λ = trial and error.

- too low : many steps to converge
- too high : overshoot (other side) DIVERGE

Regression w/ GD . Error = Obj = $J(w) = \frac{1}{2}\sum_{i=1}^{N}(x_i w - y_i)^2$

$$= \frac{1}{2}\sum_{i=1}^{N}\left(\sum_{d=1}^{D} x_i^d \cdot w^d - y_i\right)^2$$

$$\frac{\partial J}{\partial w^d} = \frac{1}{2}\cdot 2(xw-y)\frac{\partial(xw-y)}{\partial w^d} = (xw-y)\frac{\partial\left(\sum_d x^d w^d - y\right)}{\partial w^d} = (xw-y)\,x^d$$

x^d

w comp d

GD update RULE (iterative)

$$w^d_{new} = w^d - \lambda\left[(xw-y)x^d\right]$$

for all dimd one datapoint x_i

$$w_{new} = w - \lambda\left[(x_i w - y)x_i\right]$$

w_{new}: vector; w: vector

- **batch**: keep w = fixed (from prev iteration)

update $w_{new} = w - \lambda\left[(x_i w - y)x_i\right]$ for all $i = 1:N$

more stable / batch size $\uparrow = N$

$$\Rightarrow w_{new} = w_{(prev)} - \lambda\sum_{i=1}^{N}\left[(x_i w - y_i)x_i\right]$$

sum of GD updates for all datapoints

- <u>stochastic</u> ! change w (current) with every datap update

faster convergence "batch of size 1"

for i = 1 : N

$$w_{new} = w - \lambda\left[(x_i w - y_i)x_i\right]$$

$w = w_{new}$ → before moving to next datapoint x_{i+1}

- practice: batch of size $T = 20$.

batch { keep w fixed for T datapoints (w_prev)
update w = ...
move to next batch with updated w

TERMINATION
- $\frac{\partial J}{\partial w}$ close to zero
- $|w_{new} - w|$ very low
- J-oBJ good enough (not for regression) ⟹ performance (acc) good enough

Reg + L_2 regularization $J(w) = \frac{1}{2}\sum_{i=1}^{N}(x_i w - y_i)^2 + \frac{1}{2}L_2\|w\|^2$ "Ridge"

L_2 penalty

L_2 = how much penalty.

$$\frac{\partial J}{\partial w^d} = \cdots (xw - y)x^d + L_2 w^d$$

GD update $w^d_{new} = w^d - \lambda\left[(x_i w - y_i)x_i^d + L_2 w^d\right]$

for datap x_i

- high L_2 penalty $\Rightarrow$ $\|w\|$ small (constrained)
  $J(w)$ not nec. optimal
- low L_2 penalty $\Rightarrow$ $J(w)$ minimize (optimal / close to)
  $|w|$ large possibly.

L1-regularization $J(w) = \frac{1}{2}\sum_{i=1}^{N}(x_i w - y_i)^2 + L_1 \cdot |w|$ "Lasso"

$L_1 \cdot \sum_{d=1}^{D} |w^d|$

both $L_1 + L_2$: $J(w) = \frac{1}{2}\sum_{i=1}^{N}(x_i w - y_i)^2 + L_1|w| + \frac{1}{2}L_2\|w\|^2$ "Elastic Net"

derivation L_1-reg

$\frac{\partial J}{\partial w} = ?$ difficult

sparse w

some $w^d \simeq 0$

spread w over features

but try L_1-reg scikit-learn

call regression

Logistic Regression : deal with classification classes $Y \rightleftarrows \begin{matrix}\text{yes}\\ \text{no}\end{matrix}$

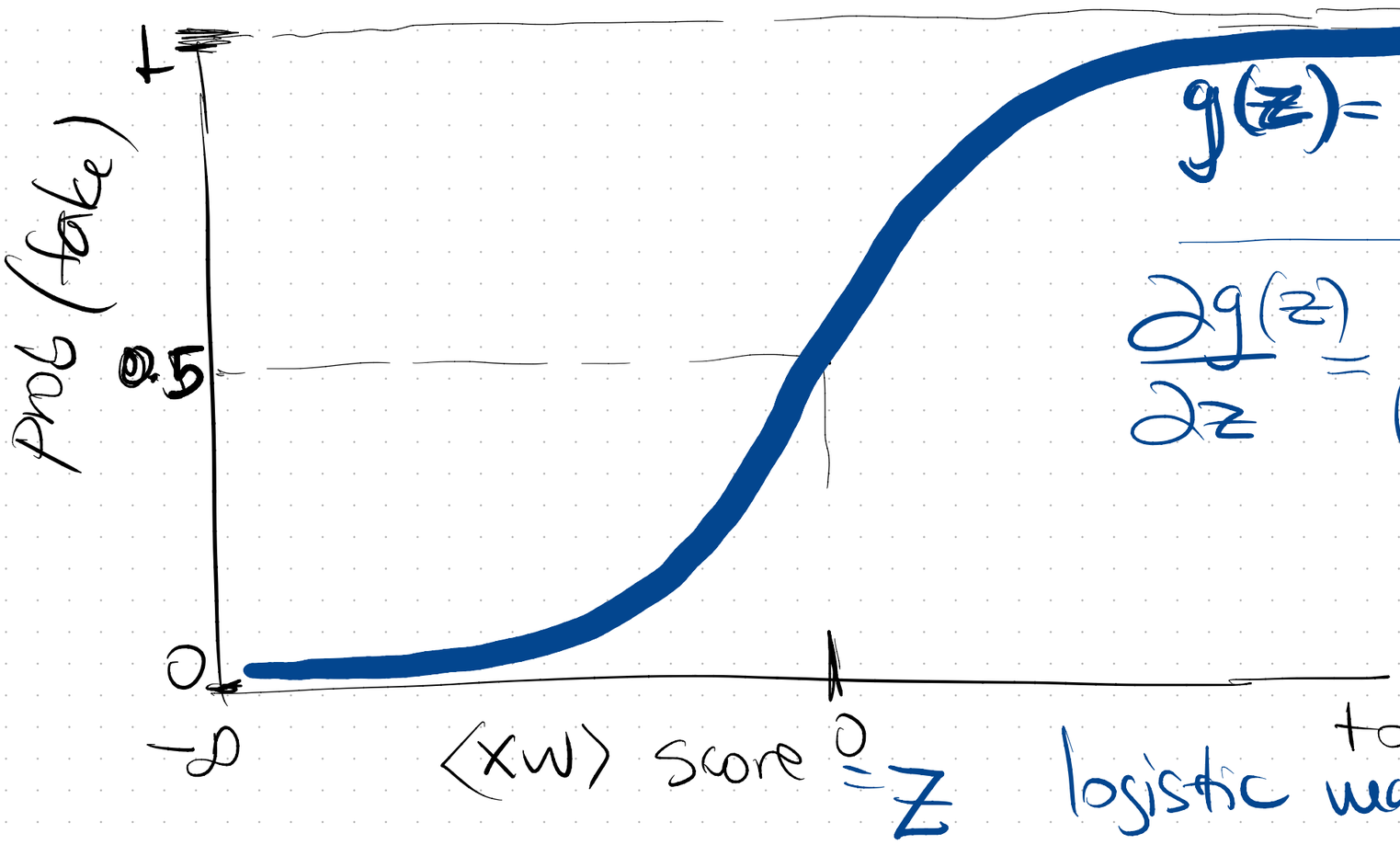
ex. Spambase dataset

Lin. Regression $h(x) = xw \triangleq Y$ = quantity label (ex. house price)

Idea • xw = linear regression score

• $h(x) = \boxed{\text{map}}$ score xw → probability of Y=yes } logistic map "g"

Z = score = xw



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

$$\frac{\partial g(z)}{\partial z} = \frac{1}{(1+e^{-z})^2} \cdot e^{-z} =$$

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

$$= g(z)\left(1 - g(z)\right)$$

- key property: overshooting does not penalize ORM

$Y = \{0, 1$

Lin Regression

want $XW \simeq Y$

if $XW \ll 0$: error $(XW-0)^2$

if $XW \gg 1$ error $(XW-1)^2$

Logistic Regression

want $\frac{1}{1+e^{-XW}} \simeq Y$

: XW very small $\to -\infty$

$\frac{1}{1+e^{\infty}} \simeq 0$

XW very large $\to +\infty$

$\frac{1}{1+e^{-\infty}} \simeq 1$

Logistic Regression uses log-likelihood OBJECTIVE (not sq error)

WANT $h(x_i) = g(x_i w) = \frac{1}{1+e^{-x_i w}} \simeq \text{prob}(Y_i = 1)$ 2 classes

$Y_i = \begin{cases} 1 \text{ yes} \\ 0 \text{ No} \end{cases}$

$P(y_i=1) \simeq h(x_i)$
$P(y_i=0) \simeq 1 - h(x_i)$
$\Rightarrow P(Y|x_i) = h(x_i)^{y_i} \cdot (1-h(x_i))^{1-y_i}$

y_i filters correct pred; $1-y_i$ filters

y_i 0/1 fillers

$h(x_i)^{y_i} \cdot (1-h(x_i))^{1-y_i} = \begin{cases} h(x_i) \text{ if } y_i = 1 \\ 1-h(x_i) \text{ if } y_i = 0 \end{cases}$

if $y_i = 1$ (yes) $\Rightarrow$ want $h(x_i)$ = high ; $1-h(x_i)$ low

$y_0 = 0$ (no) $\Rightarrow$ want $h(x_i)$ = low ; $1-h(x_i)$ high

log likelihood "LL"

$LL(w) = \log \left[\prod_{i=1}^{N} P(y_i|x_i) \right]$

product of prob of correct prediction

datapoints indep. of each other.

want MAX LL(w)

$$= \log\left(\prod_{i=1}^{N} h(x_i)^{y_i}\,(1-h(x_i))^{1-y_i}\right)$$

$$= \sum_{i=1}^{N} \log\left(h(x_i)^{y_i}\cdot(1-h(x_i))^{1-y_i}\right)$$

$$= \sum_{i=1}^{N}\left(\log\left(h(x_i)^{y_i}\right) + \log\left(1-h(x_i)\right)^{1-y_i}\right)$$

$$= \sum_{i=1}^{N}\left[y_i \log\left(h(x_i)\right) + (1-y_i)\log(1-h(x_i))\right]$$

gradient

- $\dfrac{\partial LL}{\partial w} =$
- GD update $w_{new} = w + \lambda \dfrac{\partial LL}{\partial w}(w)$ ("ascent")