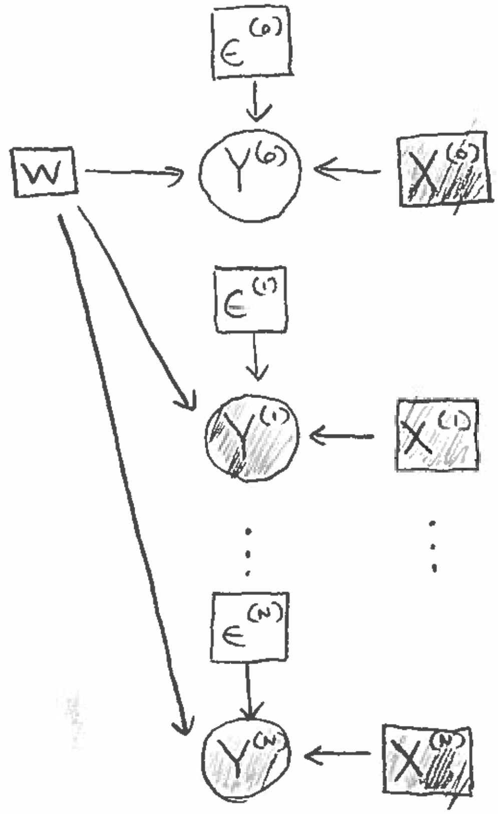
Regression: A Neural View

① By now, we've seen several instances of the regression model:



② For each of these, we've figured out how to compute a point estimate of the weight vector by minimizing a loss function:

$$\text{compute } \hat{w} = \underset{w}{\operatorname{argmax}}\ P(w) \prod_{n=1}^{N} P(y^{(n)} \mid w, x^{(n)})$$

$$= \underset{w}{\operatorname{argmin}}\ L(w)$$

where:

$$L(w) = \sum_{n=1}^{N} (y^{(n)} - w^T x^{(n)})^2 \quad \text{for ordinary linear regression}$$

$$L(w) = \sum_{n=1}^{N} (y^{(n)} - w^T x^{(n)})^2 + \frac{\sigma^2}{\tau^2} w^T w \quad \text{for ridge regression}$$

$$L(w) = \sum_{n=1}^{N} (1 - y^{(n)}) w^T x^{(n)} + \log(1 + e^{-w^T x^{(n)}}) \quad \text{for logistic regression}$$

Regression: A Neural View

③ Notice that all of these loss functions can be expressed:

$$L(w) = \sum_{n=1}^{N} L^{(n)}(w)$$

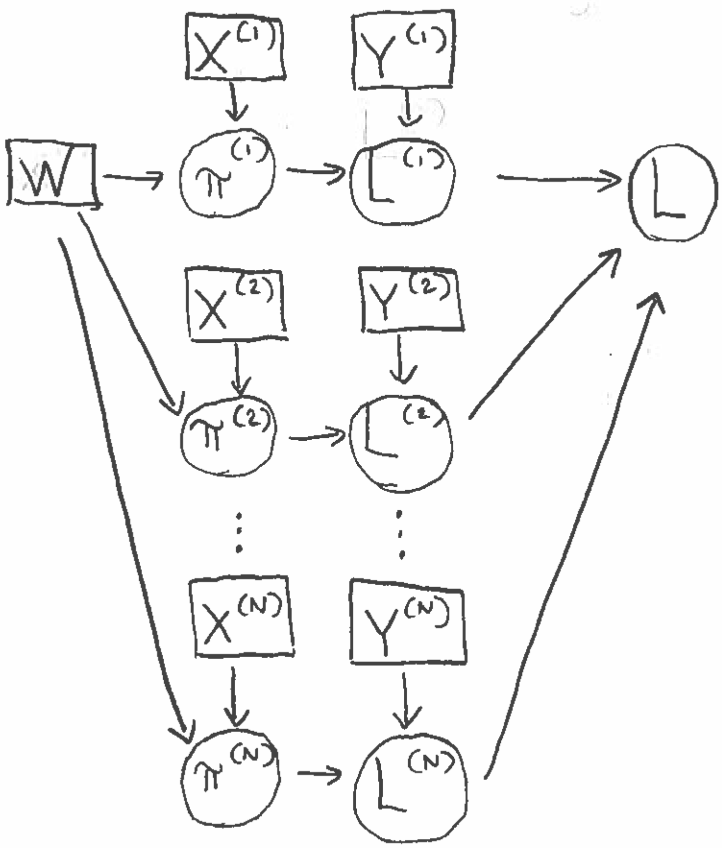
e.g.

$$L^{(n)}(w) = (y^{(n)} - w^T x^{(n)})^2 \quad \text{for ordinary linear regression}$$

$$L^{(n)}(w) = (1-y^{(n)})w^T x^{(n)} + \log(1+e^{-w^T x^{(n)}}) \quad \text{for logistic regression}$$

---

④ We can depict point estimation as a causal diagram:



where: $\pi^{(n)} \leftarrow w^T x^{(n)}$

$$L \leftarrow \sum_{n=1}^{N} L^{(n)}$$

$$L^{(n)} \leftarrow \begin{cases} (y^{(n)} - \pi^{(n)})^2 & \text{for ordinary linear regression} \\ (1-y^{(n)})\pi^{(n)} + \log(1+e^{-\pi^{(n)}}) & \text{for logistic regression} \end{cases}$$

REGRESSION: A NEURAL VIEW

(5) The variables $L^{(1)}, \ldots, L^{(N)}$ separate W from L in the causal diagram, so we can apply the Chain Rule of Partial Derivatives to compute $\frac{\partial L}{\partial w}$:

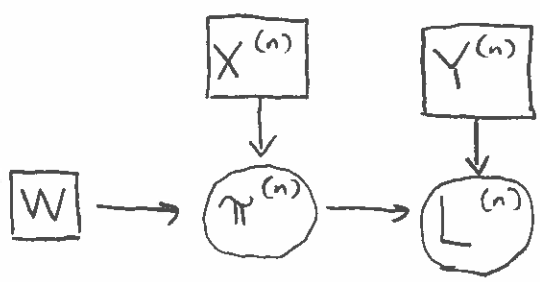
$$\frac{\partial L}{\partial w} = \sum_{n=1}^{N} \frac{\partial L}{\partial L^{(n)}} \cdot \frac{\partial L^{(n)}}{\partial w}$$

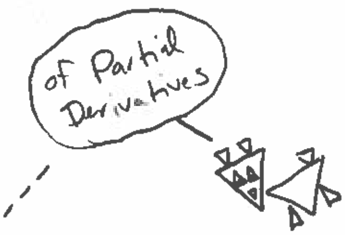
$$= \sum_{n=1}^{N} \frac{\partial \sum_{n=1}^{N} L^{(n)}}{\partial L^{(n)}} \cdot \frac{\partial L^{(n)}}{\partial w}$$

$$= \sum_{n=1}^{N} \frac{\partial L^{(n)}}{\partial w} \qquad \left[\text{b/c } \frac{\partial \sum_{n=1}^{N} L^{(n)}}{\partial L^{(n)}} = 1 \right]$$

---

(6) So the main computational task is to compute $\frac{\partial L^{(n)}}{\partial w}$ for any arbitrary n:





We can continue to use the Chain Rule to do this, since $\pi^{(n)}$ separates W from $L^{(n)}$.

$$\frac{\partial L^{(n)}}{\partial w} = \frac{\partial}{\partial \pi^{(n)}} L^{(n)} \cdot \frac{\partial}{\partial w} \pi^{(n)}$$

Regression: A Neural View

⑦ Simplifying, we get:

$$\frac{\partial}{\partial w} L^{(n)} = \left(\frac{\partial}{\partial \pi^{(n)}} L^{(n)}\right) \cdot \frac{\partial w^T x^{(n)}}{\partial w}$$

$$= \left(\frac{\partial}{\partial \pi^{(n)}} L^{(n)}\right) \cdot \begin{bmatrix} \frac{\partial}{\partial w_1} w_1 x_1^{(n)} + \dots + w_D x_D^{(n)} \\ \vdots \\ \frac{\partial}{\partial w_D} w_1 x_1^{(n)} + \dots + w_D x_D^{(n)} \end{bmatrix}$$

$$= \left(\frac{\partial}{\partial \pi^{(n)}} L^{(n)}\right) \cdot \begin{bmatrix} x_1^{(n)} \\ \vdots \\ x_D^{(n)} \end{bmatrix}$$

$$= \left(\frac{\partial}{\partial \pi^{(n)}} L^{(n)}\right) \cdot x^{(n)}$$

---

⑧ Finally, we need to compute $\frac{\partial}{\partial \pi^{(n)}} L^{(n)}$, which is the only thing that depends on which version of regression we're using (each has its own loss function). For ordinary linear regression:

$$\frac{\partial L^{(n)}}{\partial \pi^{(n)}} = \frac{\partial}{\partial \pi^{(n)}} (y^{(n)} - \pi^{(n)})^2$$

$$= 2(y^{(n)} - \pi^{(n)}) \cdot (-1)$$

$$= -2(y^{(n)} - \pi^{(n)})$$

9 Putting it all together:

$$\frac{\partial L}{\partial w} = \sum_{n=1}^{N} \frac{\partial L^{(n)}}{\partial w}$$

$$= \sum_{n=1}^{N} \frac{\partial}{\partial \pi^{(n)}} L^{(n)} \frac{\partial}{\partial w} \pi^{(n)}$$

$$= \sum_{n=1}^{N} -2\left(y^{(n)} - \pi^{(n)}\right) x^{(n)}$$

$$= -2 \sum_{n=1}^{N}\left(y^{(n)} x^{(n)} - \pi^{(n)} x^{(n)}\right)$$

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

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

[b/c $y^{(n)}$ and $w^T x^{(n)}$ are scalars]

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

[dot product commutes]

$$= -2\left(X^T y - X^T X w\right)$$

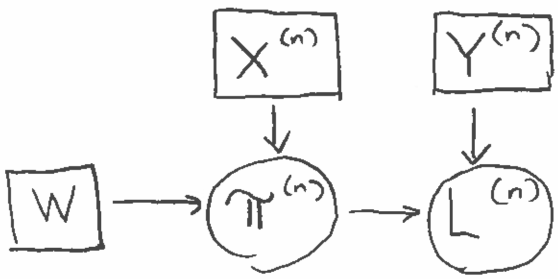
[replacing sum w matrix multiplication]

$$= -2X^T y + 2X^T X w$$

which is our gradient from LINEAR REGRESSION: MLE 10.

REGRESSION: A NEURAL VIEW

(10) This technique of computing $\frac{\partial L}{\partial w}$ by breaking it down repeatedly into simpler derivatives using the Chain Rule is a simple instance of a technique called <u>backpropagation</u>. The subdiagram



is a simple instance of a <u>neural network</u>.