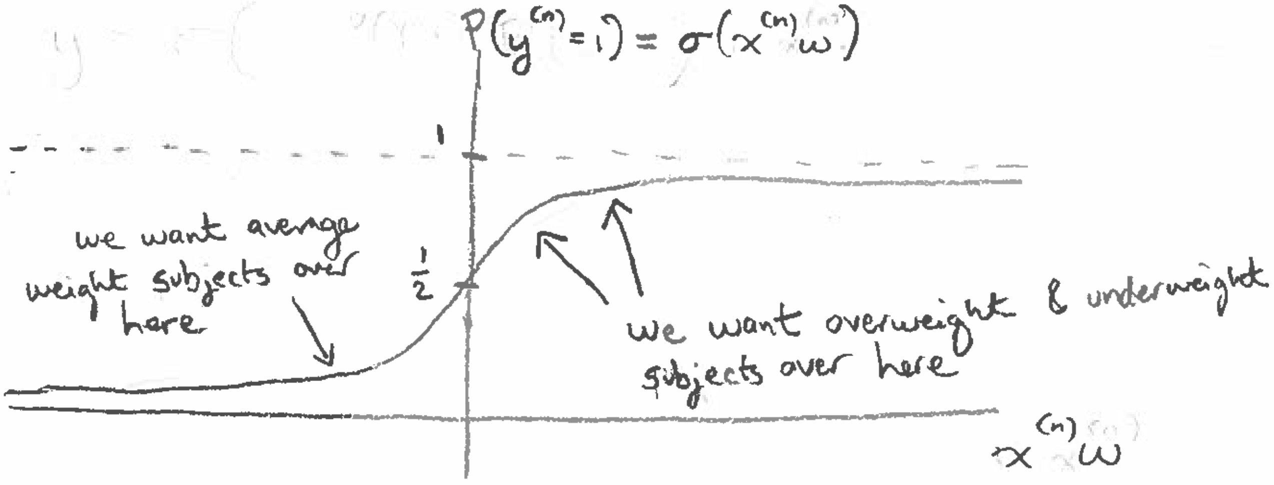
FEATURE DISCOVERY NETWORKS

① Consider a disease that affects people who are either overweight or underweight. We collect data:

X (evidence vars)			Y (response var)
X_1 (offset)	X_2 (height)	X_3 (mass)	(disease)
1	6.6	120	1
1	6.0	200	0
1	5.5	120	0
1	5.0	250	1
1	6.4	260	0
1	7.0	150	1

② Let's suppose we try to explain this data using a logistic regression model, i.e. $P(y^{(n)}=1) = \sigma(x^{(n)}w)$



We want $P(y^{(n)}=1) = \sigma(x^{(n)}w)$ to be high for over/underweight

In other words, we want $x^{(n)}w > 0$ for over/underweight subjects

FEATURE DISCOVERY NETWORKS

③ So we want (subjects 4 and 6 in our dataset):

$$w_1 + 5w_2 + 250w_3 > 0$$

$$w_1 + 7w_2 + 150w_3 > 0$$

for some weight vector $\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$, in order to give a high disease probability to the (overweight) 5', 250lb subject and the (underweight) 7', 150 lb subject.

At the same time, we want (subject 2 in our dataset):

$$w_1 + 6w_2 + 200w_3 < 0$$

in order to give a low disease probability to the 6', 200lb subject.

④ But is that even possible? It implies:

$$w_1 + 6w_2 + 200w_3 < w_1 + 7w_2 + 150w_3$$

$$\Rightarrow \quad w_2 > 50w_3$$

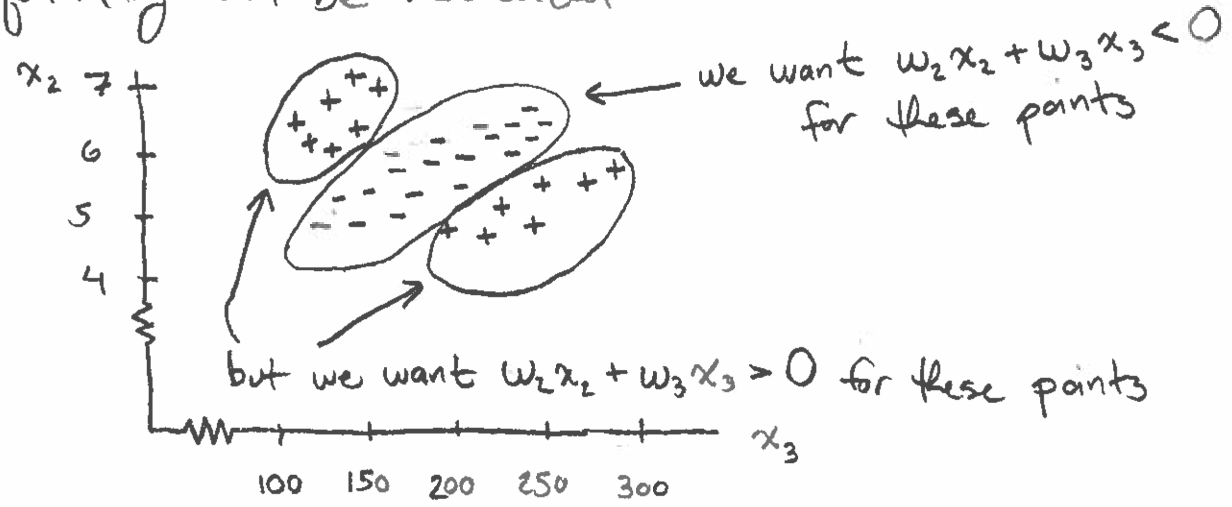
and:

$$w_1 + 6w_2 + 200w_3 < w_1 + 5w_2 + 250w_3$$

$$\Rightarrow \quad w_2 < 50w_3$$

However, w_2 cannot be both less than and greater than $50w_3$. So there's a contradiction — no such weight vector exists.

⑤ This quandary can be visualized:



we want $w_2x_2 + w_3x_3 < 0$ for these points

but we want $w_2x_2 + w_3x_3 > 0$ for these points

This is impossible.

⑥ What can we do? Rather than use the provided evidence variables (height and mass), can we make new evidence variables from the existing data?

Well, we could create a variable that indicates how underweight a person is:

$$\dot{x}_1 = \begin{cases} 40x_2 - x_3 - 120 & \text{if } 40x_2 - x_3 - 120 > 0 \\ 0 & \text{otherwise} \end{cases}$$

If a subject is not underweight, then $\dot{x}_1 = 0$. Otherwise $\dot{x}_1$ is some positive value (the higher it is, the more underweight). For instance, for the 7', 150 lb subject:

$$\dot{x}_1 = 40 \cdot 7 - 150 - 120 = 10$$

FEATURE DISCOVERY NETWORKS

⑦ Similarly, we can create a variable that indicates how overweight a subject is:

$$\dot{x}_2 = \begin{cases} x_3 - 45x_2 & \text{if } x_3 - 45x_2 > 0 \\ 0 & \text{otherwise} \end{cases}$$

For the 5', 250lb subject:

$$\dot{x}_2 = 250 - 200 = 50$$

⑧ If we convert our data into these new evidence variables, it looks like this:

$\dot{X}$ (new evidence variables)			Y (response var)
$\dot{X}_1$	$\dot{X}_2$	$\dot{X}_3$	
(underweight)	(overweight)	(offset)	(disease)
24	0	1	1
0	0	1	0
0	0	1	0
0	25	1	1
0	0	1	0
10	0	1	1

FEATURE DISCOVERY NETWORKS

(9) Now we can explain the data with logistic regression. Using weight vector $\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$, we get:

$\dot{X}$ (new evidence variables)			$\dot{X}w$	Y (response var)
$\dot{x}_1$ (underweight)	$\dot{x}_2$ (over weight)	$\dot{x}_3$ (offset)		(disease)
24	0	1	23	1
0	0	1	−1	0
0	0	1	−1	0
0	25	1	24	1
0	0	1	−1	0
10	0	1	9	1

So $\dot{x}w > 0$ (i.e. $P(Y=1) > \frac{1}{2}$) only for the subjects who got the disease.

Feature Discovery Networks

(10) But if we take our massaging of the evidence variables into consideration, the entire process was a bit more complicated than just logistic regression.

First, we took the original evidence variables and weighted them:

$$\begin{bmatrix} x_1^{(n)} & x_2^{(n)} & x_3^{(n)} \end{bmatrix} \begin{bmatrix} -120 \\ 40 \\ -1 \end{bmatrix} = \pm 120 x_1^{(n)} + 40 x_2^{(n)} - x_3^{(n)}$$

$$= 40 x_2^{(n)} - x_3^{(n)} - 120$$

$$\begin{bmatrix} x_1^{(n)} & x_2^{(n)} & x_3^{(n)} \end{bmatrix} \begin{bmatrix} 0 \\ -45 \\ 1 \end{bmatrix} = -45 x_2^{(n)} + x_3^{(n)}$$

also:

$$\begin{bmatrix} x_1^{(n)} & x_2^{(n)} & x_3^{(n)} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = x_1^{(n)} = 1$$

$$= x_3^{(n)} - 45 x_2^{(n)}$$

In other words, we right-multiplied each evidence vector $x^{(n)}$ by a weight matrix $\dot{W}$:

$$\begin{bmatrix} x_1^{(n)} & x_2^{(n)} & x_3^{(n)} \end{bmatrix} \begin{bmatrix} -120 & 0 & 1 \\ 40 & -45 & 0 \\ -1 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 40 x_2^{(n)} - x_3^{(n)} - 120 & x_3^{(n)} - 45 x_2^{(n)} & 1 \end{bmatrix}$$

(11) Then we applied the following function to each element of the resulting vector:

$$a(z) = \begin{cases} 0 & \text{if } z < 0 \\ z & \text{otherwise} \end{cases}$$

This particular function is known by the punchy name Rectified Linear Unit (ReLU).

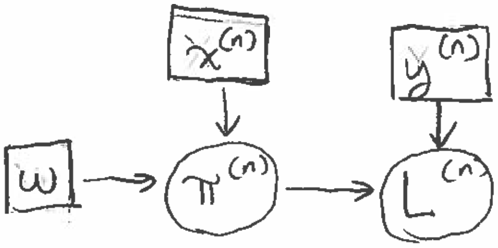
(12) This created a new set of evidence variables

$$\dot{x}^{(n)} = \begin{bmatrix} \dot{x}_1^{(n)} \\ \dot{x}_2^{(n)} \\ \dot{x}_3^{(n)} \end{bmatrix}^T = \begin{bmatrix} a(40x_2^{(n)} - x_3^{(n)} - 120) \\ a(x_3^{(n)} - 45x_2^{(n)}) \\ a(1) \end{bmatrix}^T = \begin{bmatrix} a(40x_2^{(n)} - x_3^{(n)} - 120) \\ a(x_3^{(n)} - 45x_2^{(n)}) \\ 1 \end{bmatrix}^T$$

that we used for standard logistic regression.

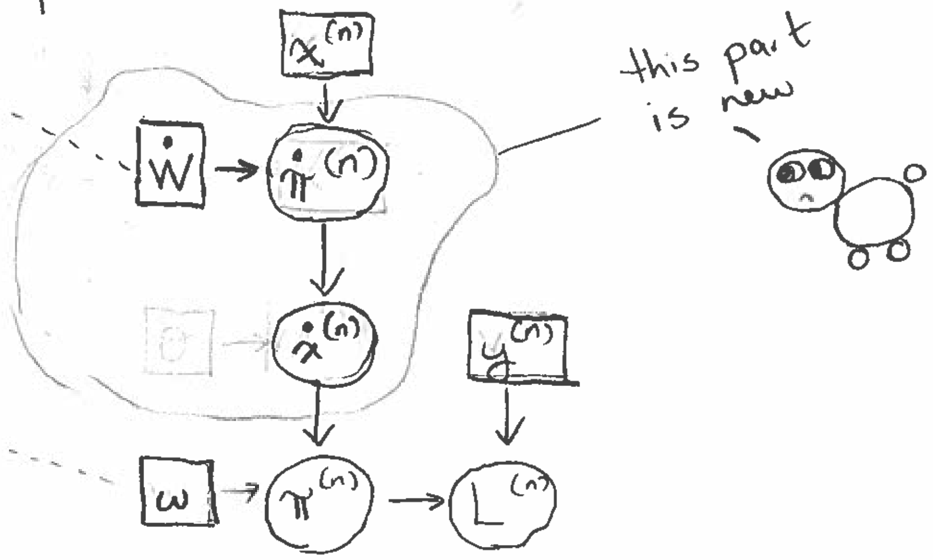
FEATURE DISCOVERY NETWORKS

⑬ While the causal diagram for logistic regression looked like:



This new process looks like:

$$\dot{W} = \begin{bmatrix} -120 & 0 & 1 \\ 40 & -45 & 0 \\ -1 & 1 & 0 \end{bmatrix}$$



$$w = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$$

where:

$$\dot{\pi}^{(n)} = x^{(n)} \dot{W}$$

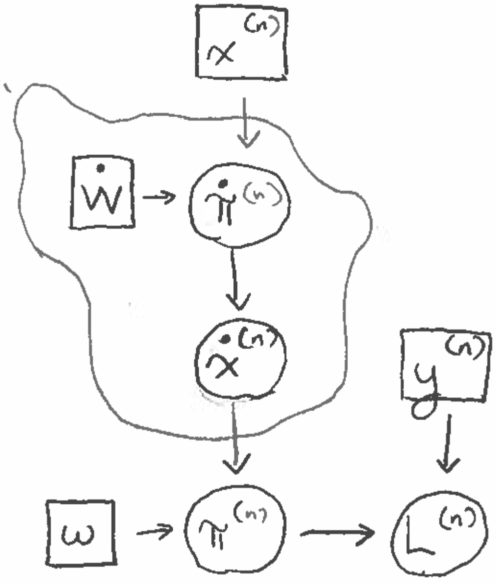
$$\dot{x}^{(n)} = \begin{cases} 0 & \text{if } \dot{\pi}^{(n)} \leq 0 \\ \dot{\pi}^{(n)} & \text{otherwise} \end{cases}$$

$$\pi^{(n)} = \dot{x}^{(n)} w$$

$$L^{(n)} = (1-y^{(n)})\pi^{(n)} + \log(1+e^{-\pi^{(n)}})$$

Feature Discovery Networks

(14) Observe that if this part is hand-engineered, then it's often called feature engineering.



For many years, this was the way machine learning went. Humans (through trial and error) painstakingly crafted features (evidence variables) for which models like logistic regression could learn good classifiers.

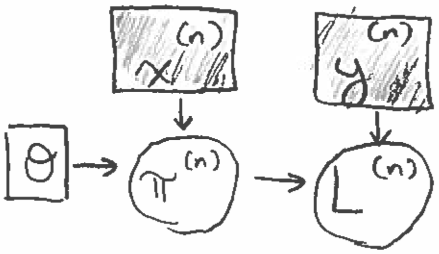
(15) Consider Project 3 (Logistics). Simple features like "percentage of black pixels" and "percentage of white pixels" aren't enough to train a good logistic regression classifier. So we used our intuition and imagination to derive better evidence variables from the raw image data, like "percentage of contrasting pixels separated by a single pixel".

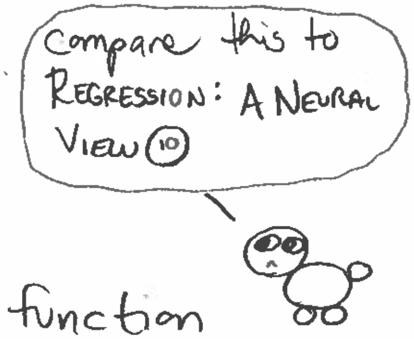
But what if we could learn the "feature engineering" step as well?

FEATURE DISCOVERY NETWORKS

⑯ Let's let $\theta = \{w_i \in w\} \cup \{\dot{w}_{ij} \in \dot{W}\}$.

If we redraw the causal diagram over $x^{(n)}, \theta, \pi^{(n)}, L^{(n)}, y^{(n)}$, we get:





Our structural equation for the loss function doesn't change:

$$L^{(n)} = (1-y^{(n)})\pi^{(n)} + \log(1+e^{-\pi^{(n)}})$$

Though now our structural equation for $\pi^{(n)}$ is some complicated non-linear function f of its parents

$$\pi^{(n)} = f(\theta, x^{(n)})$$

⑰ But our goal is still the same, right? We want to set weights θ to minimize loss $L^{(n)}$:

$$\text{compute } \hat{\theta} = \underset{\theta}{\text{argmin}}\ L^{(n)}(\theta \mid x^{(n)}, y^{(n)})$$

this is a function of $\theta, x^{(n)}, y^{(n)}$, but $x^{(n)}$ and $y^{(n)}$ are observed

FEATURE DISCOVERY NETWORKS

18 So all we really need is a way to compute $\frac{\partial}{\partial \theta} L^{(n)}$, and then we just use gradient descent.

Then we'd have a system that discovers its own features from raw data.

How hard could that be?