Lecture - 10 - fT2qJM

fT2qJM

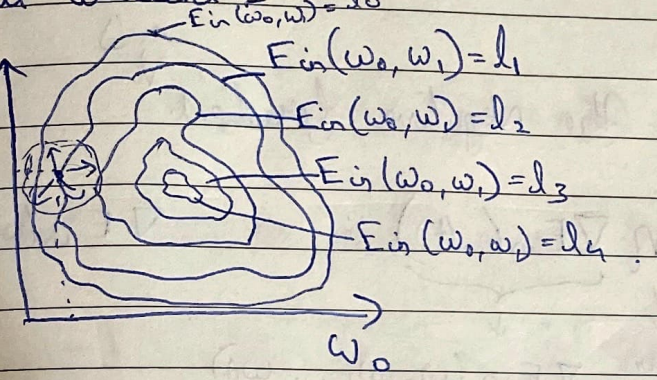
26th September.

Minimize $E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{y_n w^T x_n}\right)$

* Method of Maximum likelihood: Minimise Cross Entropy Error.
- No Analytic Solution, cannot $(\nabla w) = 0$.

To maximise the probability i.e to minimise the cross entropy error, take derivative.

* Hill descent:



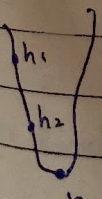
Contour map.

$l_0 > l_1 > l_2 > l_3 > l_4$

To find the required point, ~~the~~ take the gradient i.e take the derivative till it becomes 0.

$\Delta E_{in}(w) = E_{in}(w_{new}) - E_{in}(w)$ (Not a formal defn).

When choosing the direction where our new (w_0, w_1) will be based on the step size (radius)



here the derivative at point $h_1 \neq 0$
at point $h_3 = 0$.

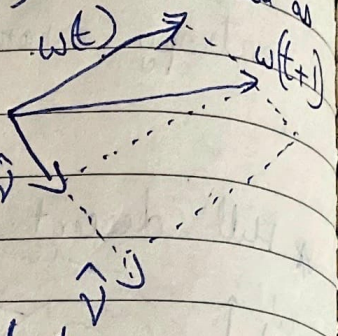
Take a ...

★ How to Roll down the hill?

- At any ~~step~~ time step t, suppose we are at weights $w(t)$
  - take a step of size η in direction $\hat{v}$

$$w(t+1) = w(t) + \eta\hat{v}$$

Greedify : Pick $\hat{v}$ so $E_{in}(w(t+1))$ is as small as possible.



$\hat{v}$ = ~~&~~ Greedy direction.

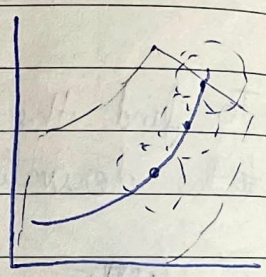
Rolling down ≡ Iterating the Negative Gradient.

$$w(t+1) = w(t) - \eta \nabla E_{in}(w(t))$$

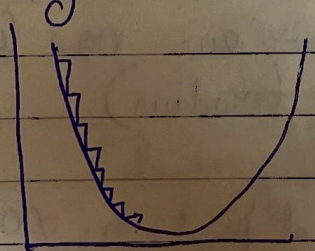
$d+1 \times 1$ vector; scalar value; $\nabla E_{in}(w_0, w_1, \ldots w_d)$ $d+1 \times 1$ vector.

$$\nabla E_{in}(w(t)) = \begin{bmatrix} \frac{\partial E_{in}(w)}{\partial w_0} \\ \vdots \\ \frac{\partial E_{in}(w)}{\partial w_d} \end{bmatrix}$$

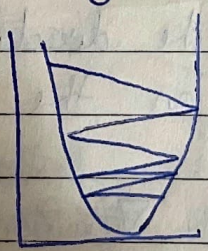
$\hat{v}$ choose direction to reduce E_{in}.
Problem is with ~~the~~ choosing the right value η.



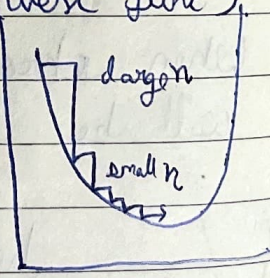
★ Picking the "learning Rate" η (convex func')



η too small



η too large



Variable η

↓

How to drop to small η when the ball jumps to

* Rolling down.

$$E_{in}(w(t+1)) = E_{in}(w(t) + \eta\hat{v})$$

[Taylor approximation] $\approx E_{in}(w(t)) + \eta\hat{v}^T \nabla E_{in}(w(t)) + \frac{\eta^2}{2!}\hat{v}^T\nabla^2 \ldots$

$$\approx E_{in}(w(t)) + \eta\hat{v}^T \nabla E_{in}(w(t)) + O(\eta^2)$$

$$\Delta E_{in} = E_{in}(w(t+1)) - E_{in}(w(t))$$
$$\approx \eta\hat{v}^T \nabla E_{in}(w(t)) + O(\eta^2)$$
$$\leq 0. \qquad \text{(We want to reduce the in sample error)}$$

So $E_{in}(w(t+1)) < E_{in}(w(t))$

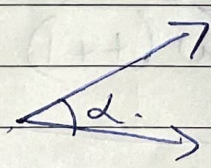
Fastest Way to Roll down. (Is the gradient)

$$-1 \leq \cos(\alpha) = \frac{\hat{v}^T \nabla E_{in}(w(t))}{\|\hat{v}\| \, \|\nabla E_{in}(w(t))\|} \leq 1$$

substitute $\|\hat{v}\| \, \|\nabla E_{in}(w(t))\|$ as c

$$-c \leq c.\cos(\alpha) = \hat{v}^T \nabla E_{in}(w(t)) \leq c.$$

for $\cos 180 = -1$.



So we should choose $\alpha = 180°$

★ Gradient Descent Algo :-

1. Initialize at step $t=0$ to $w(0)$.
2. For $t = 0, 1, 2, \cdots$ do.
3. Compute the gradient $g_t = \nabla E_{in}(w(t))$ → can be expensive
4. move in direction $v_t = -g_t$.
5. Update the weights $w(t+1) = w(t) + \eta v_t$.
6. Iterate until stopping condition is met.
7. End for.
8. Return final weights.

$$\text{Minimize } E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-y_n w^T x_n}\right)$$

By moving small steps $-\eta \nabla E_{in}(w)$.

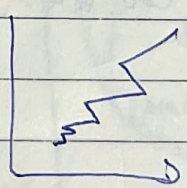
★ Stochastic Gradient Descent (SGD) :-

→ • Pick a data point (x_*, y_*) from D at random.

→ • Run an iteration of GD on error (w, x_*, y_*)

$$w(t+1) = w(t) - \eta \nabla \text{error}(w, x_*, y_*)$$

$$\boxed{w(t+1) = w(t) + y_* x_* \frac{\eta}{1 + e^{y_* w^T x_*}}}$$ SGD.



U

GD
10 steps

Not necessary to have ~~higher~~ lower E_{in}.
that may increase cost by overfitting

SGD
30 steps.