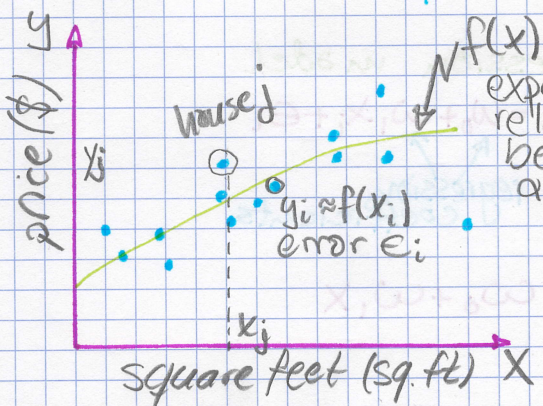
Simple Regression

Linear regression with one input

$(x_1 = \text{sq.ft}, y_1 = \$)$
$(x_2 = \text{sq.ft}, y_2 = \$)$
$(x_3 = \text{sq.ft}, y_3 = \$)$
$(x_4 = \text{sq.ft}, y_4 = \$)$
$(x_5 = \text{sq.ft}, y_5 = \$)$

Input — Output

- Input vs. Output
  - y is the quantity of interest
  - assume y can be predicted from x



Regression model:

$$y_i = f(x_i) + \epsilon_i$$

$$E[\epsilon_i] = 0$$

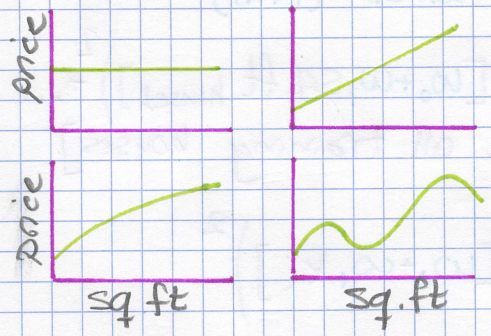
↑ expected value

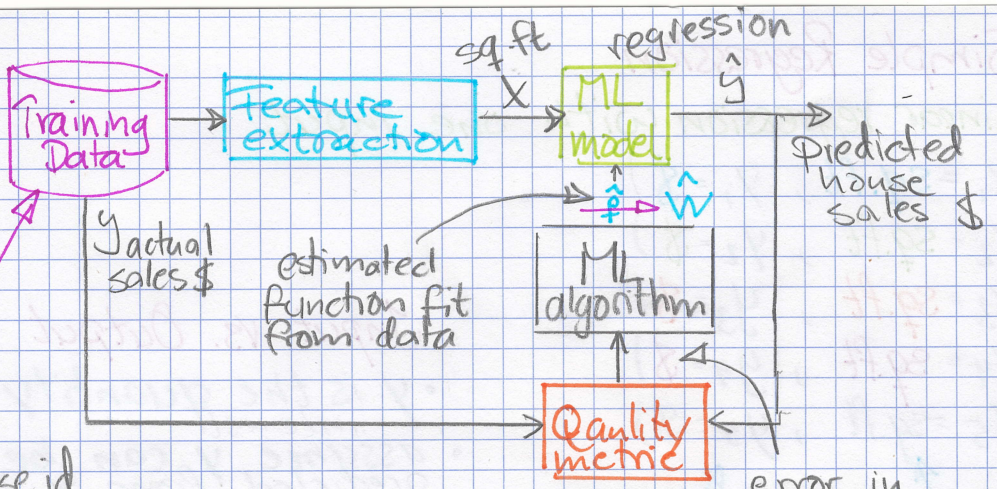
equally likely that error is + or −

⇓

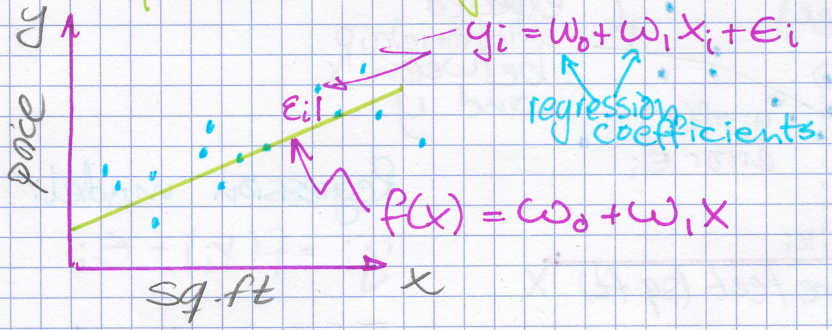
y_i is equally likely to be above or below $f(x_i)$

Which Model f(x)?





Simple Linear regression model



Cost of using a given line
Residual sum of squares (RSS)

$$RSS(w_0, w_1) = \left(\$_{house1} - [w_0 + w_1 \, sq.ft_{house1}]\right)^2 + \ldots$$
$$+ [\text{include all training houses}]$$

$$RSS(w_0, w_1) = \sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i]\right)^2$$

Regression model

$$y_i = w_0 + w_1 x_i + \varepsilon_i$$

w_0, w_1: parameters (unknown variables)

Estimated parameters

$\hat{w}_0, \hat{w}_1$

take actual values

Minimizing the cost

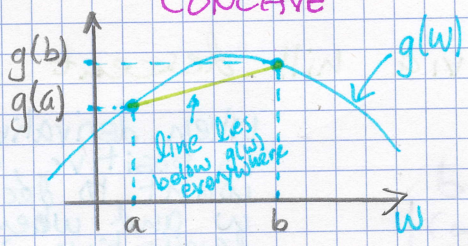
$$\min_{w_0, w_1} \sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i] \right)^2$$

$RSS(w_0, w_1)$ is a function of 2 variables $= g(w_0, w_1)$

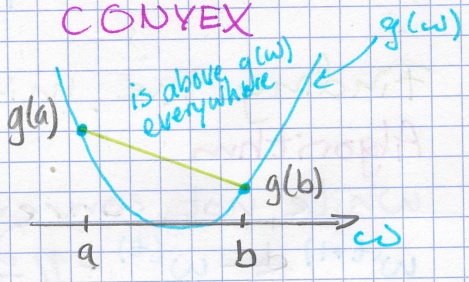
Finding maxima or minima analytically

Convex / concave functions

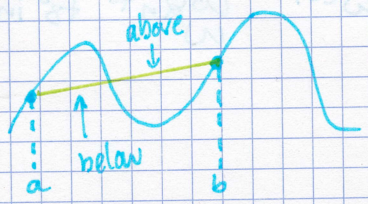
CONCAVE



CONVEX



NEITHER



Example

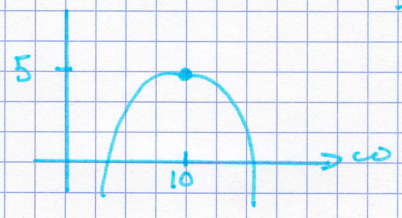
$g(w) = 5 - (w - 10)^2$

$\frac{\partial g(w)}{\partial w} = 0 - 2(w-10)^1 \cdot 1 = -2w + 20$

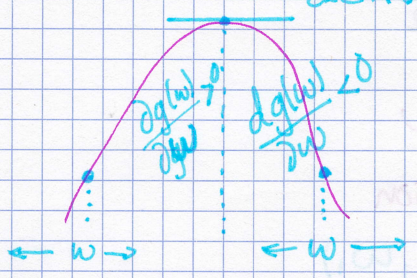
set derivate $= 0$:

$-2w + 20 = 0$

$w = 10$



Finding the max via hill climbing



How do we know whether to move w to right or left? (Inc. or dec. the value of w?)

while not converged

$w^{(t+1)} \leftarrow w^{(t)} + \eta \frac{\partial g(w)}{\partial w}$

$w^{(t+1)}$: iteration t; η: step size

Finding the min via hill descent

Algorithm

while not converged

$w^{(t+1)} \leftarrow w^{(t)} - \eta \frac{\partial g(w)}{\partial w}\Big|_{w^{(t)}}$

When derivative is positive, we want to decrease w and when derivative is negative, we want to increase w

Choosing the stepsize — fixed stepsize

η ← stepsize

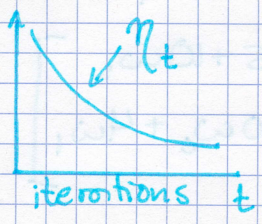
$\eta = 0.1$

Decreasing stepsize or stepsize schedule

Common choices:

$$\eta = \frac{\alpha}{t}$$

$$\eta = \frac{\alpha}{\sqrt{t}}$$



Convergence criteria

For convex functions, optimum occurs when

$$\frac{\partial g(w)}{\partial w} = 0$$

In practice, stop when

$$\left| \frac{\partial g(w)}{\partial w} \right| < \varepsilon$$

ε ← threshold to be set

Gradient: derivatives in multiple dimensions

$$\nabla g(w) = \begin{bmatrix} \frac{\partial g}{\partial w_0} \\ \frac{\partial g}{\partial w_1} \\ \vdots \\ \frac{\partial g}{\partial w_p} \end{bmatrix}$$

∇ ← gradient

$[w_0, w_1, \ldots]$

$(p+1)$-dimensional vector

partial derivatives is like a derivate with respect to w, treating all other variables as constants

Gradient example

$$g(w) = 5w_0 + 10w_0w_1 + 2w_1^2$$

$$\frac{\partial g}{\partial w_0} = 5 + 10w_1$$

$$\frac{\partial g}{\partial w_1} = 10w_0 + 4w_1$$

$$\nabla g(w) = \begin{bmatrix} 5 + 10w_1 \\ 10w_0 + 4w_1 \end{bmatrix}$$

Gradient descent

Algorithm

while not converged:

$$w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla g(w^{(t)})$$

Convergence: $\|\nabla g(w)\| < \varepsilon$

Computing the gradient of RSS

$$RSS(w_0, w_1) = \sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i]\right)^2$$

$$\frac{\partial}{\partial w} \sum_{i=1}^{N} g_i(w) = \sum_{i=1}^{N} \frac{\partial}{\partial w} g_i(w)$$

In our case

$$g_i(w) = (y_i - [w_0 + w_1 x_i])^2$$

$$\frac{\partial RSS(w)}{\partial w_0} = \sum_{i=1}^{N} \frac{\partial}{\partial w_0} \left(y_i - [w_0 + w_1 x_i]\right)^2$$

same for w_1

Taking the derivative w.r.t w_0

$$\sum_{i=1}^{N} 2\left(y_i - [w_0 + w_1 x_i]\right)^1 \cdot (-1) =$$

$$-2\sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i]\right)$$

Taking the derivative w.r.t w_1

$$\sum_{i=1}^{N} 2\left(y_i - [w_0 + w_1 x_i]\right)^1 \cdot (-x_i)$$

$$= -2\sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i]\right) x_i$$

$$\nabla RSS(w_0, w_1) = \begin{bmatrix} -2\sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i]\right) \\ -2\sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i]\right) x_i \end{bmatrix}$$

actual house sales observation

predicted value $\hat{y}_i(w_0, w_1)$

Approach 1

top line

$$\hat{w}_0 = \frac{\sum_{i=1}^{N} y_i}{N} - \hat{w}_1 \frac{\sum_{i=1}^{N} x_i}{N}$$

average house sales price

average sq. ft

estimate of the slop

plug in

Bottom term

$$\sum y_i x_i - \hat{w}_0 \sum x_i - \hat{w}_1 \sum x_i^2 = 0$$

$$\hat{w}_1 = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}}$$

Note:

$\sum y_i$

$\sum x_i$

$\sum y_i x_i$

$\sum_{i=1}^{N} x_i^2$

Approach 2: Gradient descent

$$\nabla RSS(w_0, w_1) = \begin{bmatrix} -2\sum y_i - \hat{y}_i(w_0, w_1) \\ -2\sum (y_i - \hat{y}_i(w_0, w_1)) x_i \end{bmatrix}$$

while not converged:

$$\begin{matrix} w_0^{(t)} \\ w_1^{(t)} \end{matrix} - \overbrace{+2\eta}\, \eta \begin{bmatrix} -2\sum (y_i - \hat{y}_i(w_0^{(t)}, w_1^{(t)})) \\ -2\sum (y_i - \hat{y}_i(w_0^{(t)}, w_1^{(t)})) x_i \end{bmatrix} =$$

$$= \begin{bmatrix} w_0^{(t+1)} \\ w_1^{(t+1)} \end{bmatrix}$$

If overall, underpredicting $\hat{y}_i$, then $\sum [y_i - \hat{y}_i]$ is positive → w_0 is going to increase. Similar intuition for w_1, but multiply with x_i

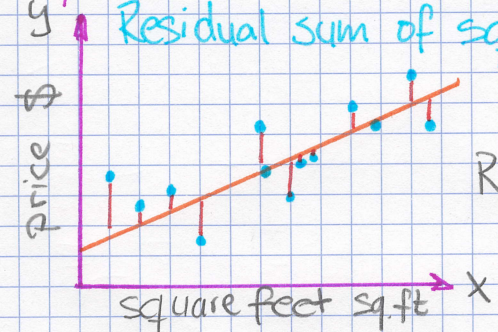
Comparing the approaches

- For most ML problems, cannot solve gradient = 0
- Even if solving gradient = 0 is feasible, gradient descent can be more efficient
- Gradient descent relies on choosing stepsize and convergence criteria

Asymmetric cost functions

Symmetric cost function

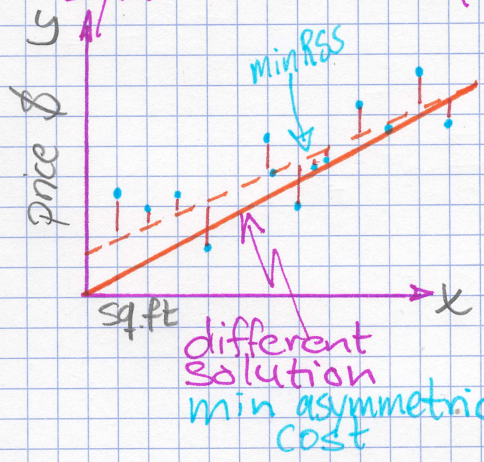
Residual sum of squares (RSS)



$$RSS(w_0, w_1) = \sum_{i=1}^{N} \left(y_i - [w_0 + w_1 x_i] \right)^2$$

Assumes cost of over-estimating sales price is same as under-estimating

Asymmetric cost functions



What if cost of listing house too high has bigger cost?

Too high → no offers $=0

Too low → offers for lower $