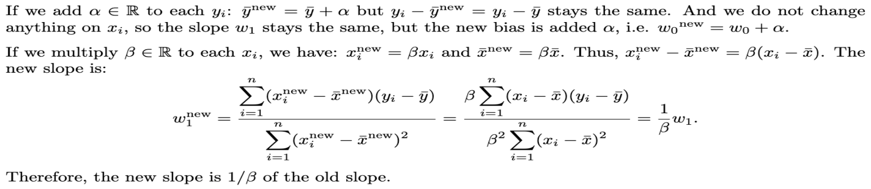
How constants Affect MSE: (MSE minimized by mean, MAE minimized by MAE)

Multiplying by c: MSE will be stretched/compressed vertically since when we take dR/dh, we set it equal to 0 to find the local minima

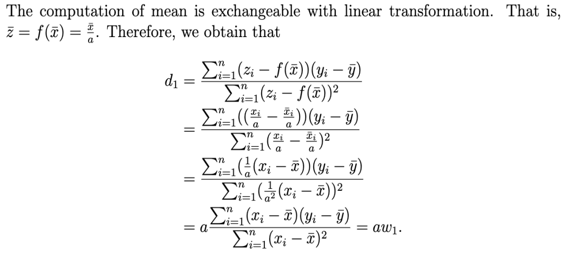


- If we add $\alpha \in \mathbb{R}$ to each y_i: $\bar{y}^{\text{new}} = \bar{y} + \alpha$ but $y_i - \bar{y}^{\text{new}} = y_i - \bar{y}$ stays the same. And we do not change anything on x_i, so the slope w_1 stays the same, but the new bias is added α, i.e. $w_0{}^{\text{new}} = w_0 + \alpha$.
- If we multiply $\beta \in \mathbb{R}$ to each x_i, we have: $x_i^{\text{new}} = \beta x_i$ and $\bar{x}^{\text{new}} = \beta\bar{x}$. Thus, $x_i^{\text{new}} - \bar{x}^{\text{new}} = \beta(x_i - \bar{x})$. The new slope is:

$$w_1^{\text{new}} = \frac{\sum_{i=1}^{n}(x_i^{\text{new}} - \bar{x}^{\text{new}})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i^{\text{new}} - \bar{x}^{\text{new}})^2} = \frac{\beta\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\beta^2\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{1}{\beta}w_1.$$

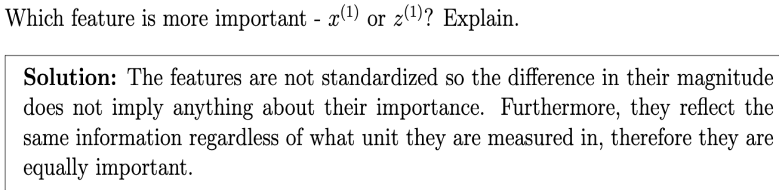
Therefore, the new slope is $1/\beta$ of the old slope.

Ex: Have prediction rule h1(x) = w0 + w1x, convert x1 to transformation by setting z1 = x1/a → h2(z) = d0 + d1z. What is d1?



The computation of mean is exchangeable with linear transformation. That is, $\bar{z} = f(\bar{x}) = \frac{\bar{x}}{a}$. Therefore, we obtain that

$$\begin{aligned} d_1 &= \frac{\sum_{i=1}^{n}(z_i - f(\bar{x}))(y_i - \bar{y})}{\sum_{i=1}^{n}(z_i - f(\bar{x}))^2} \\ &= \frac{\sum_{i=1}^{n}((\frac{x_i}{a} - \frac{\bar{x}_i}{a}))(y_i - \bar{y})}{\sum_{i=1}^{n}(\frac{x_i}{a} - \frac{\bar{x}_i}{a})^2} \\ &= \frac{\sum_{i=1}^{n}(\frac{1}{a}(x_i - \bar{x}))(y_i - \bar{y})}{\sum_{i=1}^{n}(\frac{1}{a^2}(x_i - \bar{x}))^2} \\ &= a\frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = aw_1. \end{aligned}$$



Which feature is more important - $x^{(1)}$ or $z^{(1)}$? Explain.

Solution: The features are not standardized so the difference in their magnitude does not imply anything about their importance. Furthermore, they reflect the same information regardless of what unit they are measured in, therefore they are equally important.

How adding data points affect MSE/MAE:

Consider a dataset of 23 points A = {y1 - y23} in order. Create B by having 2 of each point B = {y1, y1 - y23, y23}

If Rh = MAE and h* = 5 for A, then h* = 5 for B since MAE is minimized by median

In addition, the std does not change based on the std dev formula: sum(deviations) is doubled, by then we divide by 2 since n = n * 2

MAE(A) = MAE(B) because

$$R_A(h) = \frac{1}{23}\sum_{i=1}^{23}|y_i - h|$$

$$\begin{aligned} R_B(h) &= \frac{1}{46}\sum_{i=1}^{46}(|y_1 - h| + |y_1 - h| + |y_2 - h| + |y_2 - h| + ... + |y_{23} - h| + |y_{23} - h|) \\ &= \frac{1}{46}\left(2\sum_{i=1}^{23}|y_i - h|\right) \end{aligned}$$

Loss Functions:

Cubed loss has no minimizer since it goes to -infinity

Mean of 12 non-negative numbers is 45. Suppose remove 2 numbers. What is the largest possible value of the mean of the remaining 10 numbers? Recall that the sum of the 12 number set is 12 · 45; the maximum possible mean of the remaining 10 is $\frac{12 \cdot 45 - 2 \cdot 0}{10} = \frac{6}{5} \cdot 45 = 54$

Reminder: behavior of loss functions will tell you their minimizer: e^(h+1)^2 anything squared > 0; h* = -1

Convexity:

Jensen's Inequality: (1-t)f(x1) + tf(x2) >= f((1-t)x1 + tx2)

In multi-step proofs, remember to state all prior conditions are true

Gradient Descent: Loss function must be differentiable, convex

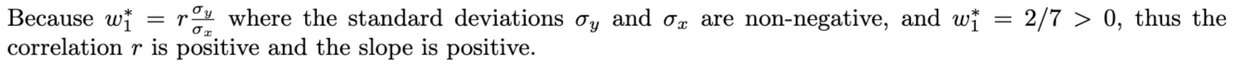
Given k data points, the order of x0...xk does not need to be monotonically increasing or decreasing; At any step, the value of the function can increase if non-convex or if a is too large

You can terminate if the change in objective is close to 0 or if the gradient is too small/the norm of gradient is small

Given f(x), gradient: (closed form: set gradient = 0)

$$f(\vec{x}) = (\vec{a} \cdot \vec{x})^2 \quad \nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1}(a_1x_1 + a_2x_2 + ... + a_n * x_n)^2 \\ \frac{\partial f}{\partial x_2}(a_1x_1 + a_2x_2 + ... + a_n * x_n)^2 \\ \vdots \\ \frac{\partial f}{\partial x_n}(a_1x_1 + a_2x_2 + ... + a_n * x_n)^2 \end{bmatrix} = \begin{bmatrix} 2 \cdot (a_1x_1 + a_2x_2 + ... + a_n * x_n)^1 \cdot a_1 \\ 2 \cdot (a_1x_1 + a_2x_2 + ... + a_n * x_n)^1 \cdot a_2 \\ \vdots \\ 2 \cdot (a_1x_1 + a_2x_2 + ... + a_n * x_n)^1 \cdot a_3 \end{bmatrix} \quad h_i = h_{i-1} - \alpha \cdot \frac{dR}{dh}(h_{i-1})$$

Linear Regression:



Because $w_1^* = r\frac{\sigma_y}{\sigma_x}$ where the standard deviations σ_y and σ_x are non-negative, and $w_1^* = 2/7 > 0$, thus the correlation r is positive and the slope is positive.

Standard deviation = sqrt((xi - x_var)^2 / n), If you multiply all x's by c, then std = c(old_std)

r has no units

Moving points up by c will move intercept up by c

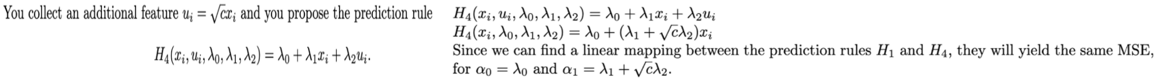
Assuming Features not standardized: features of smaller units will have more weight

Assuming Features standardized: feature most correlated with response var will have most impact

Adding features won't increase MSE bc we cannot fit the data any worse since we could set the coeff = 0

Adding c data points that exactly fit prediction rule will not change MSE since their errors are all 0

Closed Form: $\vec{w} = (X^TX)^{-1}X^T\vec{y},$



You collect an additional feature $u_i = \sqrt{c}x_i$ and you propose the prediction rule

$$H_4(x_i, u_i, \lambda_0, \lambda_1, \lambda_2) = \lambda_0 + \lambda_1x_i + \lambda_2u_i.$$

$H_4(x_i, u_i, \lambda_0, \lambda_1, \lambda_2) = \lambda_0 + \lambda_1x_i + \lambda_2u_i$

$H_4(x_i, \lambda_0, \lambda_1, \lambda_2) = \lambda_0 + (\lambda_1 + \sqrt{c}\lambda_2)x_i$

Since we can find a linear mapping between the prediction rules H_1 and H_4, they will yield the same MSE, for $\alpha_0 = \lambda_0$ and $\alpha_1 = \lambda_1 + \sqrt{c}\lambda_2$.

If we have non-linear prediction rule, we can apply functions to it to make it linear; ex: w0e^w1x → apply log transformation

K-Means:

C(a, b) = (x1 - a)^2 + (y1 - b)^2 + ... + (xn - a)^2 + (yn - b)^2

a* = (x1 + ... xn) / n, b* = (y1 + ... yn) / n

Pick a value of k and randomly initialize k centroids.

Keep the centroids fixed, and update the groups.

Assign each point to the nearest centroids

Keep the groups fixed, and update the centroids

Move each centroid to the center of its group by averaging their coordinates

Repeat steps 2 and 3 until the centroids stop changing
Solution to address convergence not being optimal: Run K-Means several times, each with different randomly chosen initial centroids. Keep track of the inertia of the final result in each attempt. Choose the attempt with the lowest inertia or Choose one initial centroid at random, and choose the remaining initial centroids by maximizing distance from all other centroids.