Univariate linear Regression

ARYAN KUMAR

Machine learning

ARYAN KUMAR

- Given an experience E, performance measure P, task T.
- A machine is said to be learning if its P wrt a task T improves with experience, E.

Types of Machine learning

- Supervised :

Data is labelled or we know the correct answers as it provided.

Types:

Output → Regression: → Response is quantitative

Output → Classification: → Response is categorical

- Unsupervised :

Data is not labelled. We usually find:
- Structure
- Patterns
- Groupings (Cluster's)
- Outliers (Anomalies)

Types:
- Clustering
- Anomaly Detection
- Dimensionality Reduction

Reinforcement learning ?

To be Updated

Here is a basic example:

Eg. Teaching a dog to play fetch using treats as a reward

Univariate linear Regression

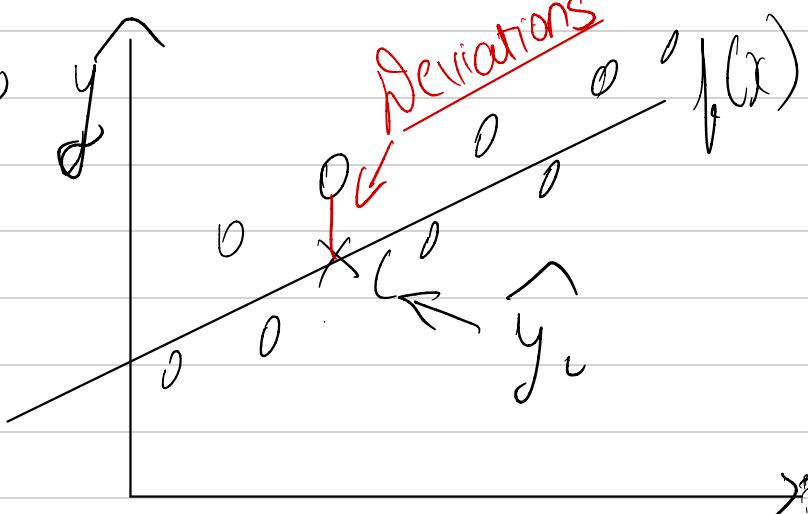
How to do Univariate linear Regression:

We could use python and do it one line using:
Sklearn, Statsmodel, pytorch, tensorflow, numpy.

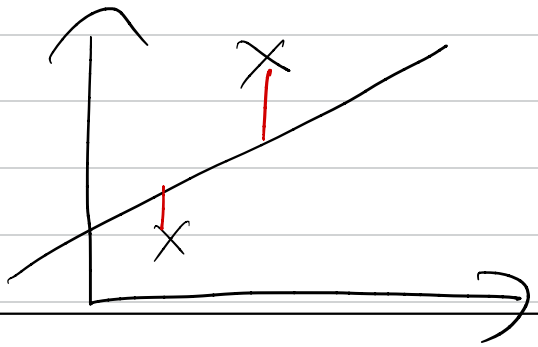
1. Pick a form for our model f.
2. Choose a cost func. J. J will help us in getting the best model parameters.
3. Train the model. Apply Gradient Descent to find optimal parameters.

Pick a general form for our model, f:

- $f_{w,b}(x_i) = wx_i + b + \varepsilon$ ← Irreducible Error
- w & b are model parameters.
- w (weight or β_1 in stats) $y = \beta_0 + x\beta_1 + \varepsilon$
- w is slope, $\tan\theta$, y
- b is y intercept



- We have observed value y_i & predicted $\hat{y}_i$
- We want a model that minimizes these deviations



- Define a cost function: We choose squared error func.

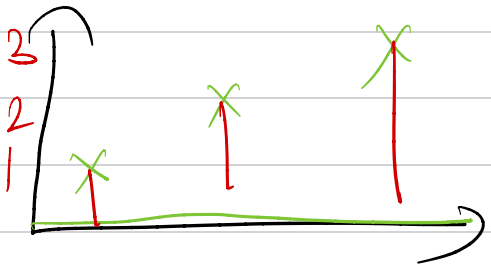
$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2 \quad f(x)$$

$$= \frac{1}{2m}\sum_{i=1}^{m}(wx_i + b - y_i)^2$$

$$= \underset{w,b}{\operatorname{argmin}}\ J(w,b)$$

Eg $f(x) = wx$

$w = 0 \Rightarrow \boxed{y = 0}$ for all x



$$J = \frac{1}{2m}\sum_{1}^{3}(wx_i - y_i)^2$$

$$= \frac{1}{2m}\left((-1)^2 + (-2)^2 + (3)^2\right)$$

$$J(0) = 14/6$$

$w = 1/2$, $\boxed{y = x/2}$

$$J = \frac{1}{2m}\sum_{1}^{3}(wx_i - y_i)^2$$

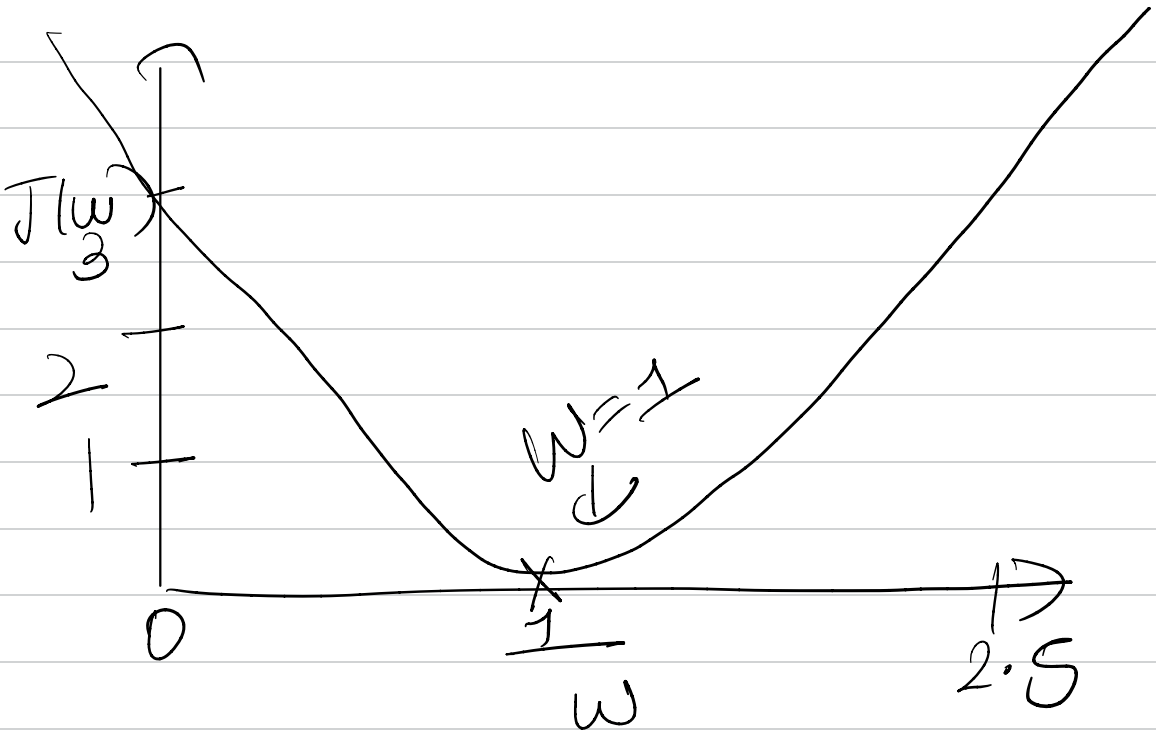
$$= \frac{1}{2m}\left(\left(\frac{1}{2}-1\right)^2 + (1-2)^2 + \left(\frac{3}{2}-3\right)^2\right)$$

$$= \frac{1}{2m}\left(\frac{1}{4} + 1 + \frac{9}{4}\right) = 14/24$$

$$= 7/12$$

$w = 1$

$J(1) = 0$



We see $J(w)$ / w is a convex parabola

- We manually calculated minimum value of $J(w)$.
- Not feasible so we use gradient descent.

Gradient Descent

1. Initialize random w
2. Repeat till convergence.

$$w = w - \alpha \, \partial J / \partial w$$

$\alpha \in \mathbb{R} > 0$, learning rate.

Finding $\partial J(w)/\partial w$: → This is just slope.

$$J = \frac{1}{2m}\sum_{i=1}^{m}(f(x)-y_i)^2$$
$$= \frac{1}{2m}\sum(wx_i-y_i)^2$$
$$= \frac{1}{2m}\left((wx_1-y_1)^2+(wx_2-y_2)^2\cdots + (wx_m-y_m)^2\right)$$

Take one at a time

$$\partial J/\partial w = \frac{1}{2m}\left(\frac{\partial((wx_1-y_1)^2)}{\partial w}\right)$$

$\Rightarrow k = wx-y \Rightarrow dk/dw = \boxed{x}$

$J \Rightarrow \frac{1}{2m}k^2$

$\partial J/\partial k \Rightarrow k \;\Big|\Rightarrow \partial J/\partial w = (\partial J/\partial k)(dk/dw) \Rightarrow (k)(x)$

$$\boxed{\partial J/\partial w \Rightarrow (wx-y)x}$$

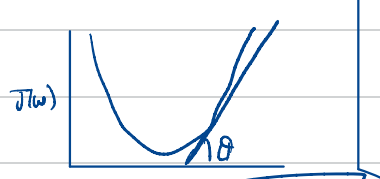
$$\partial J/\partial w = \frac{1}{m}\sum_{i=1}^{m}(\underbrace{wx_i}_{\hat{y}}-y_i)x_i$$

$$\Rightarrow \partial J/\partial w = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i-y_i)x_i$$

$\hat{y}_i \curvearrowleft a_i$

$$w = w-\alpha\,\partial J/\partial w \Rightarrow w = w-\alpha\left(\frac{1}{m}\sum a_i x_i\right)$$

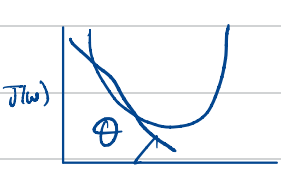
Repeating Till convergence



$\tan\theta > 0$

$w > \text{minima}$ $\quad w = w-\alpha\frac{\partial J}{\partial w}$

So w decreases $\quad \tan\theta$



$\tan\theta < 0 \;|\; \partial J/\partial w < 0$

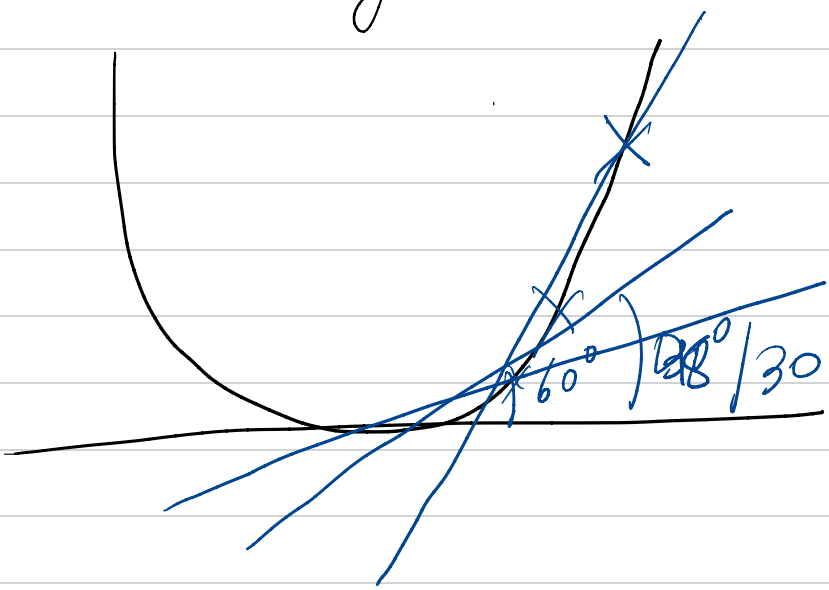
$w = w-\alpha\,\partial J/\partial w$

w increases

Learning rate α

- If α is too small it converges slowly
- If α is too large it might not converge.

Convergence



$\tan 60° = \frac{\sqrt{3}}{2}$

$w = w - \alpha \tan 60°$

$w = w - \alpha \tan 30°$

$w = w - \alpha \tan 30°$

* We see as $\theta \downarrow$, $\tan\theta \downarrow$

So slope or $\partial J / \partial w \downarrow$

∴ We do not need to change α and the rate at which it convergences reduces as we approach minima.

$f(x) = wx + b$

$J(w,b) = \frac{1}{2m} \sum_{1}^{m} (f(x_i) - y_i)^2$

$= \frac{1}{2m} \sum_{1}^{m} (wx_i + b - y_i)^2$

Gradient Descent:

- Initialize random w, b
- Repeat till convergence: $w = w - \alpha \partial J / \partial w$

  $b = b - \alpha \partial J / \partial b$

$\partial J / \partial w$

$J \Rightarrow \frac{1}{2m}\left((wx_1 + b - y_1)^2 + (wx_2 + b - y_2)^2 \cdots (wx_m + b - y_m)^2 \right)$

$\partial J_1 / \partial w \Rightarrow \frac{1}{m}\left(x_1 (wx_1 + b - y_1) \right) \quad | \quad \partial J / \partial w = \frac{1}{m} \sum_1^m \left(f(x_i) - y_i \right)$

$J_1 = \frac{1}{2m} (wx_1 + b - y_1)^2 \Rightarrow J_1 = \frac{1}{2m} \times u^2 \quad | \quad u = wx_1 + b - y_1$

$du / db = 1 \quad | \quad dJ / du = \underline{2u}$

$dJ_1 / db = \frac{1}{m} (wx_1 + b + y_1)$

$dJ / db = \frac{1}{m} \sum_{i=1}^m (wx_i + b - y_i)$

Convergence

$w = w - \alpha \, \partial J / \partial w \Rightarrow w - \alpha \left(\frac{1}{m} \Sigma \left(f(\hat{y}_i - y_1) x_i \right) \right)$

$b = b - \alpha \, \partial J / \partial b \Rightarrow b - \alpha \left(\frac{1}{m} \Sigma (\hat{y}_i - y) \right)$

→ They need to be calculated at the same iteration