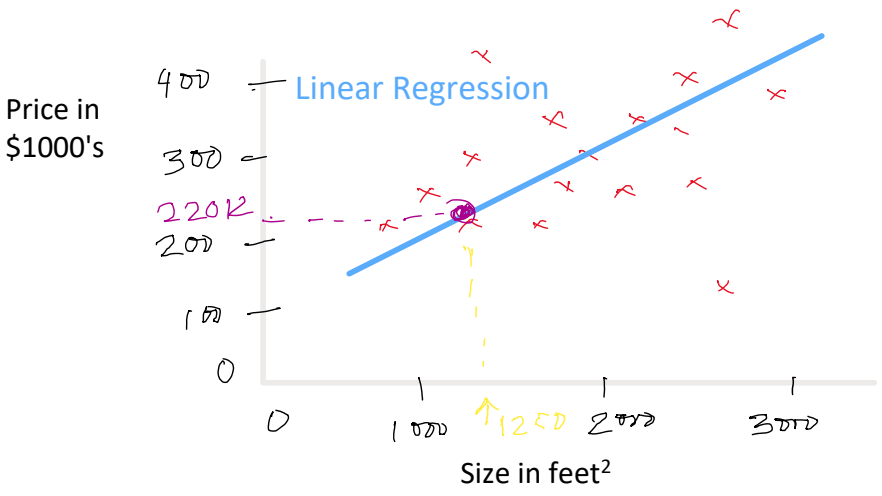
Linear Regression Model

Thursday, November 6, 2025 8:57 PM

House size and prices



Data table

Size in feet	Price in $1000

Supervised learning model

Data has the right answer

Regression model Predicts numbers
Infinitely many possible outputs

Classification models predicts
Categories
Small number of possible outputs

Terminology

Training Set: Data used to train model

x = "input" variable/Feature

Y = "output" variable/target

m = number of training example

(x- y) = single training example

$(x^{(i)}, y^{(i)}) =$ Is i^{th} position of the training example

Training set

Features
Target

Target

Leavening algorithm

"y hat"

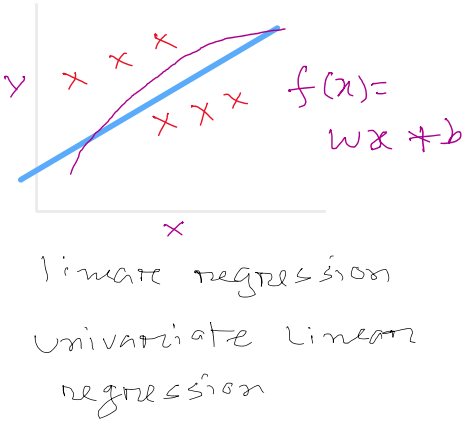
$\hat{y}$ is the prediction

Function / hypothesis

features Model Prediction (estimated y)

Hew to represent

$$f_{w,b}(x) = wx + b$$



linear regression

Univariate Linear regression

F is a function that takes x as an input and depending on values of w and b it will output some value for prediction y-hat same f(x).

W = parameter weight
B = parameter bias
We have to constrict a cost function

Cost function

Friday, November 7, 2025 12:22 PM

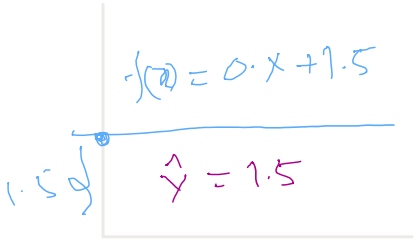
Cost function formula

Model: $f_{w,b}(x) = wx + b$

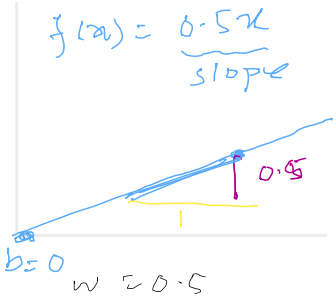
w, b: Parameters
Coefficient
weights

Depending on the value of w, b we get different function $f(x)$

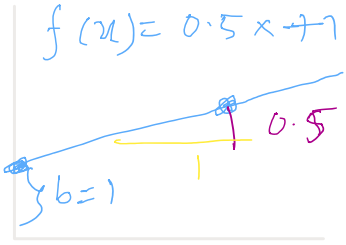
Parameters: are the variable that we can change during training.



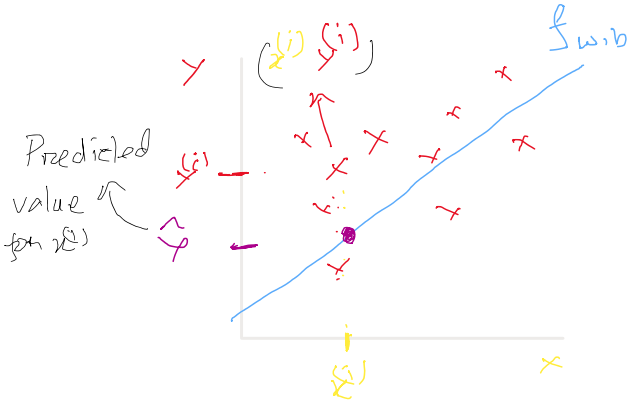
y intercept $w = 0$
$b = 1.5$



$w = 0.5$
$b = 0$



$w = 0.5$
$b = 1$



$\hat{y}^{(i)} = f_{w,b}(x^{(i)})$

$f_{w,b}(x^{(i)}) = wx^{(i)} + b$

find w, b:
$\hat{y}^{(i)}$ is close to $y^{(i)}$ for all $(x^{(i)}, y^{(i)})$

To find out how well it allignms with the traing data → Cost function

Cost function: $(\hat{y} - y)^2$ → error

$J_{w,b}$ sum of the all squared errors $\frac{1}{2m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - y^{(i)}\right)^2$ → avg squared error

m = number of training example

Squared error cost function

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2$$

Intuation of cost function
How the cost function can be used to find the
Best parameters for the model.

For linear regression we need to find the
Value of w and b that the regression line best fits our training data..

For that we find the cost function
It measures the difference between the models predictions and the
Actual true value for y.

Goal is to minimize J(w,b) $\underset{w,b}{\text{minimize}}\ J(w,b)$

when intercept $b = 0$

$$f_w(x) = wx \quad , \quad b = 0$$

$$J(w) = \frac{1}{2m}\sum_{i=1}^{m}\left(f_w(x^{(i)}) - y^{(i)}\right)^2$$

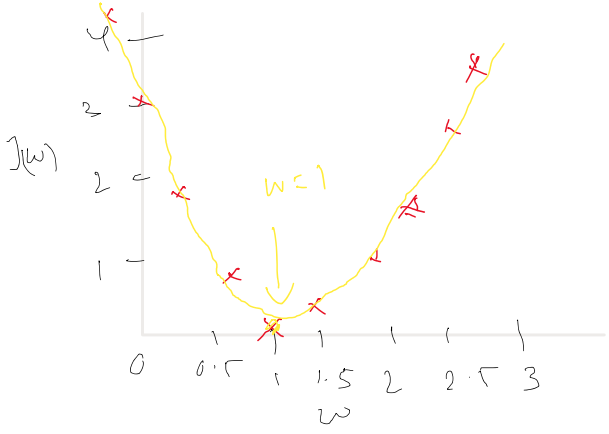
when $x = 0$
$f(x) = 0$ too.

$f_w(x)$
for fixed w, function of x
depends on input

$J(w)$
function of w
depends on parameter

goal of linear regression

$\underset{w,b}{\text{minimize}}\ J(w)$



Choose w to minimize J(w)

Practice Quiz - Regression

Friday, November 7, 2025 2:31 PM

1. Which or the following are the input, or features, that are fed into the model and with which the model is expected to make a prediction?

- [x] x
- [ ] m
- [ ] (x,y)
- [ ] w and b

2. For linear regression, if you find parameters w and b so that J(w,b) is very close to zero, what can you conclude?

- [x] The selected values of the parameters w and b cause the algorithm to fit the training set really well.
- [ ] The selected value of the parameters w and b cause the algorithm to fit the training set really poorly.
- [ ] This is never possible -- there must be a bug in the code.

Gradient Descent: Train the model with gradient descent

Friday, November 7, 2025 2:37 PM

Cost function $J(w,b)$ → linear regression

want $\min_{w,b} J(w,b)$

Gradient descent can be used for any function

So for other cost function which has more than two parameters

Gradient Descent is used.

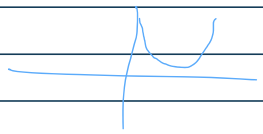
$\min_{w_1,\dots w_n,b} J(w, w_2, w_3, \dots, w_n, b)$

Outline:

Start with some w,b (Set $w=0$, $b=0$) — J is not always

Keep changing w,b to reduce $J(w,b)$

Until J settles at or near minimum



- There can be more than one minimum.

Gradient descent → Local minima

Implimentation

learning rate → how big of a step goes down will

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

derivative of cost function J

Repeat these two until converges

$$b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

Simultaniously update w and b

Correct: Simultanious Update

$$temp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

$$temp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

$$w = temp\_w$$

$$b = temp\_b$$

Gradient descent is always simultanious update.

Gradient Descent Intuition

$$w = w + \alpha \frac{\partial}{\partial w} J(w,b)$$

learning rate

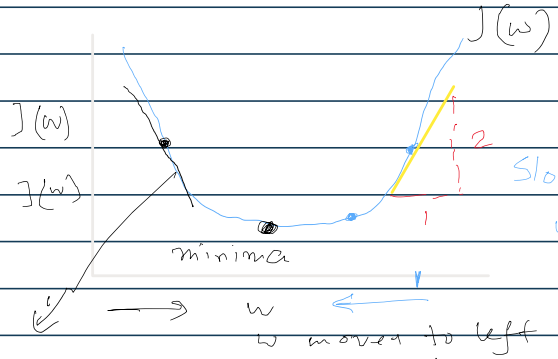
partial derivative of $J(w,b)$ with respect to w.

$$b = b + \alpha \frac{\partial}{\partial b} J(w,b)$$

cost function $J(w)$

$$w = w - \alpha \left[\frac{\partial}{\partial w} J(w) \right]$$

$\min_w J(w)$



$\text{Slope} = \frac{\text{height}}{\text{width}}$

when the tangent line is pointing up into right
means positive slope
derivative is a positive number > 0

$w = w - \alpha \cdot (\text{positive number})$

w is smaller α learning rate is a positive number

$w = w - \alpha \cdot (\text{negative number})$

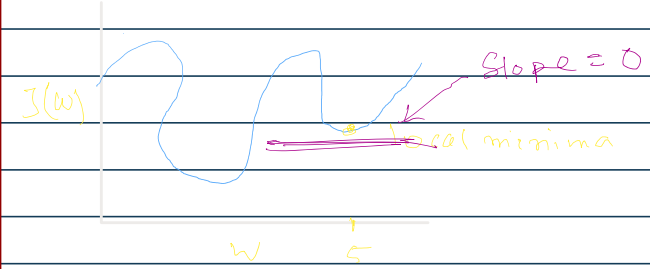
Causes w to increase
w moves to right towards the minima

? How do we choose learning rate α

Learning Rate α

$$w = w - \alpha \frac{\partial}{\partial w} J(w)$$

if α is too small
Gradient descent will be slow



if α is too large

$w = w - \alpha \cdot \frac{\partial}{\partial w} J(w) \rightarrow \text{Slope} = 0$

$w = w - \alpha \cdot 0$

$w = w$

if GD is already in local minima

minima

if GD is already in local minima
then $w = w$ it stays in the local minima for every learning rate

- It may fail to converge or even diverge

Can reach local minima with fixed learning rate.

Near the local minima,
- Derivatives becomes smaller
- Update steps become smaller even with fixed learning rate.

Can reach local minima without decreasing learning rate α.

Gradient Descent For Linear Algorithm

Squared error cost function with gradient descent for Linear regression model.

$f_{w,b}(x) = wx + b$ — model

$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2$ — cost function (Squared error cost)

repeat until convergence {

$w = w - \alpha \left[\frac{\partial}{\partial w} J(w,b)\right] \rightarrow \frac{1}{m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)x^{(i)}$

$b = b - \alpha \left[\frac{\partial}{\partial b} J(w,b)\right] \rightarrow \frac{1}{m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)$

} Gradient descent function

$$\frac{\partial}{\partial \omega} J(w,b) = \frac{\partial}{\partial w}\frac{1}{2m}\sum_{i=1}^{m} f_{w,b}\left(\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$$\frac{\partial}{\partial \omega} J(w,b) = \frac{\partial}{\partial w}\frac{1}{2m}\sum_{i=1}^{m}\left(\mathrm{w}x^{(i)} + \mathrm{b} - y^{(i)}\right)^2$$

$$\frac{\partial}{\partial \omega} J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(\mathrm{w}x^{(i)} + \mathrm{b} - y^{(i)}\right) 2x^{(i)}$$

$$\frac{\partial}{\partial \omega} J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\left(\mathrm{w}x^{(i)} + \mathrm{b} - y^{(i)}\right) x^{(i)}$$

Gradient Descent Algorithm

Repeat untill convergence

$$w = w - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}(x^{(i)}) - y^{(i)} \right) x^{(i)}$$

$\frac{d}{dw} J(w,b)$

$$b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}(x^{(i)}) - y^{(i)} \right)$$

$\frac{\partial}{\partial b} J(w,b)$

$f_{w,b}(x^{(i)}) = wx^{(i)} + b$

Squared error cost function is a convex function so it only has a global minima / no local minima

So if the learning rate is chosen properly it will go towards the global minima

Running gradient descent

Batch gradient descent.

For each step of gradient descent uses all the training example

$\sum_{i=1}^{m}$

Other version of gradient descent doesn't use all training example.

Practice Quiz - Train the model with gradient descent

Friday, November 7, 2025 10:32 PM

1. Gradient descent is an algorithm for finding values of parameter w and b that minimize the cost function J.

   Repeat until convergence {

   $$w = w - \alpha \frac{\partial}{\partial w} J(w, b)$$

   $$b = b - \alpha \frac{\partial}{\partial b} J(w, b)$$

   }

   When $\frac{\partial}{\partial w} J(w, b)$ is a ngative number (less than zero) what happens to w after one update step?

- [ ] w stays the same
- [ ] it is not possible to tell if w will increase or decrease
- [ ] w decreases
- [x] w increases

2. For linear regression what is the update step for parameter b?

- [ ] $\mathrm{b} = \mathrm{b} - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(\mathrm{w} x^{(i)} + \mathrm{b} - y^{(i)} \right) x^{(i)}$

- [x] $b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(w x^{(i)} + b - y^{(i)} \right)$