

→ The cost function will tell us how well the model is doing

$$\text{model: } f_{w,b}(x) = wx + b$$

Cost function: Squared error cost function

$$J(w,b) = \frac{1}{2m} \sum_{i=1}^m \left(\underset{\substack{\text{prediction} \\ \text{error}}}{\hat{y}^{(i)}} - y^{(i)} \right)^2$$

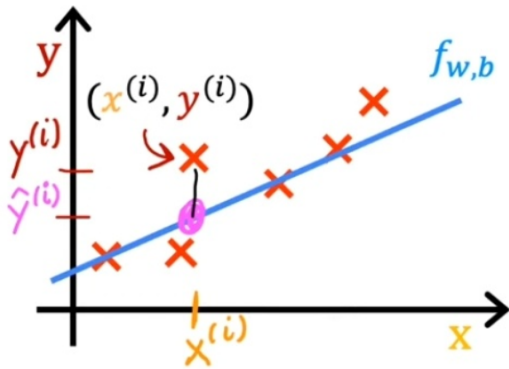
m = number of training examples

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

↑ intuition (next!)

Find w, b :

$\hat{y}^{(i)}$ is close to $y^{(i)}$ for all $(x^{(i)}, y^{(i)})$.



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

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

→ For every $x^{(i)}$ the model will predict $y^{(i)}$ based on the linear line that we get.

→ To draw the linear line we need to find the value of w and b .

→ To find the w and b we need the help of cost function.

→ So, it will guess the line by finding the minimum error.

Cost function:

$$J(w, b) = \frac{1}{2m} \sum^m (\underbrace{\hat{y}^{(i)} - y^{(i)}}_{\text{error}})^2$$

Diagram illustrating the components of the cost function $J(w, b)$:

- $\frac{1}{2m}$: average
- \sum^m : for all training data
- $(\hat{y}^{(i)} - y^{(i)})^2$: for extra neat