ED5340 - Data Science: Theory and Practise

L17 - Linear Regression: Univariate

Ramanathan Muthuganapathy (https://ed.iitm.ac.in/~raman)
Course web page: https://ed.iitm.ac.in/~raman/datascience.html
Moodle page: Available at https://courses.iitm.ac.in/

Linear Regression

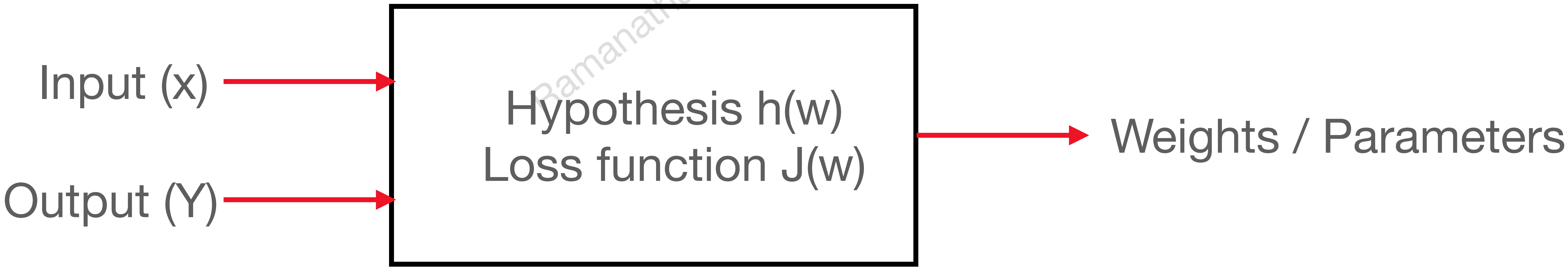
Supervised Leaning

- Ground truth data - Input feature / output $(\mathbf{x}, \mathbf{y})$ are the knowns
- Use a model / hypothesis as $h(w)$
- Develop an error / cost / loss function $J(w) = J(\mathbf{y}, \bar{\mathbf{y}}) = J(\mathbf{y}, h(w))$
- The weights are identified by
    - min $J(w)$
- Essentially, ML problem is now reduced to an optimization problem.
- Weights are identified using Optimization.

Linear Regression

Supervised Leaning

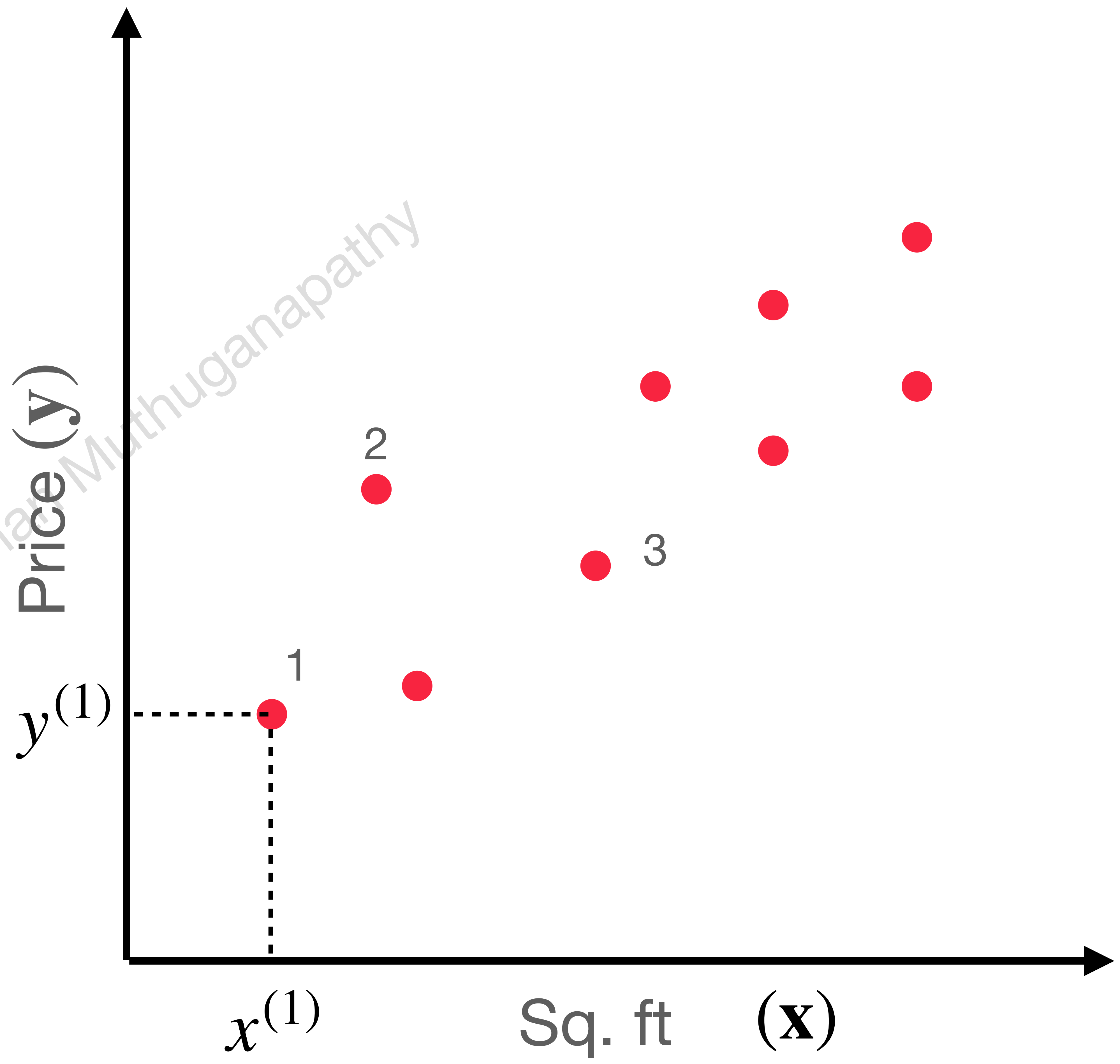
- Ground truth data - Input feature / output $(\mathbf{x}, \mathbf{y})$ are the knowns
- Use a model / hypothesis as $h(w)$ and cost function $J(w)$
-



Linear Regression

Supervised Leaning

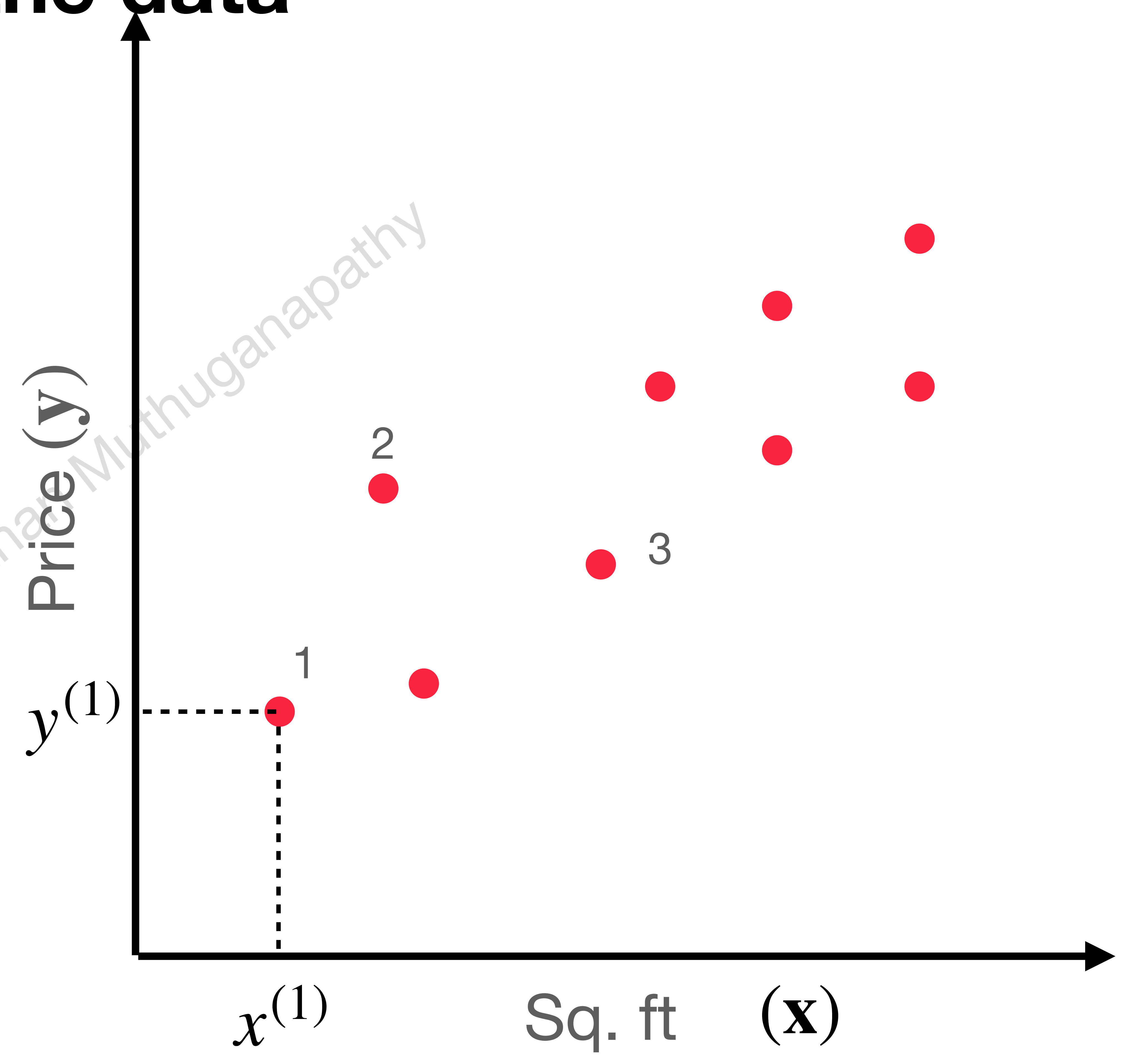
- $(\mathbf{x}, \mathbf{y})$ - (Sq. ft, Price)
- Datapoints / Training samples
- $\mathbf{x} = (x^{(1)}, x^{(2)}, x^{(3)}, \ldots . x^{(m)})$
- $\mathbf{y} = (y^{(1)}, y^{(2)}, y^{(3)}, \ldots . y^{(m)})$
- m training sample



Linear Regression

Goal: Approximation that fits the data

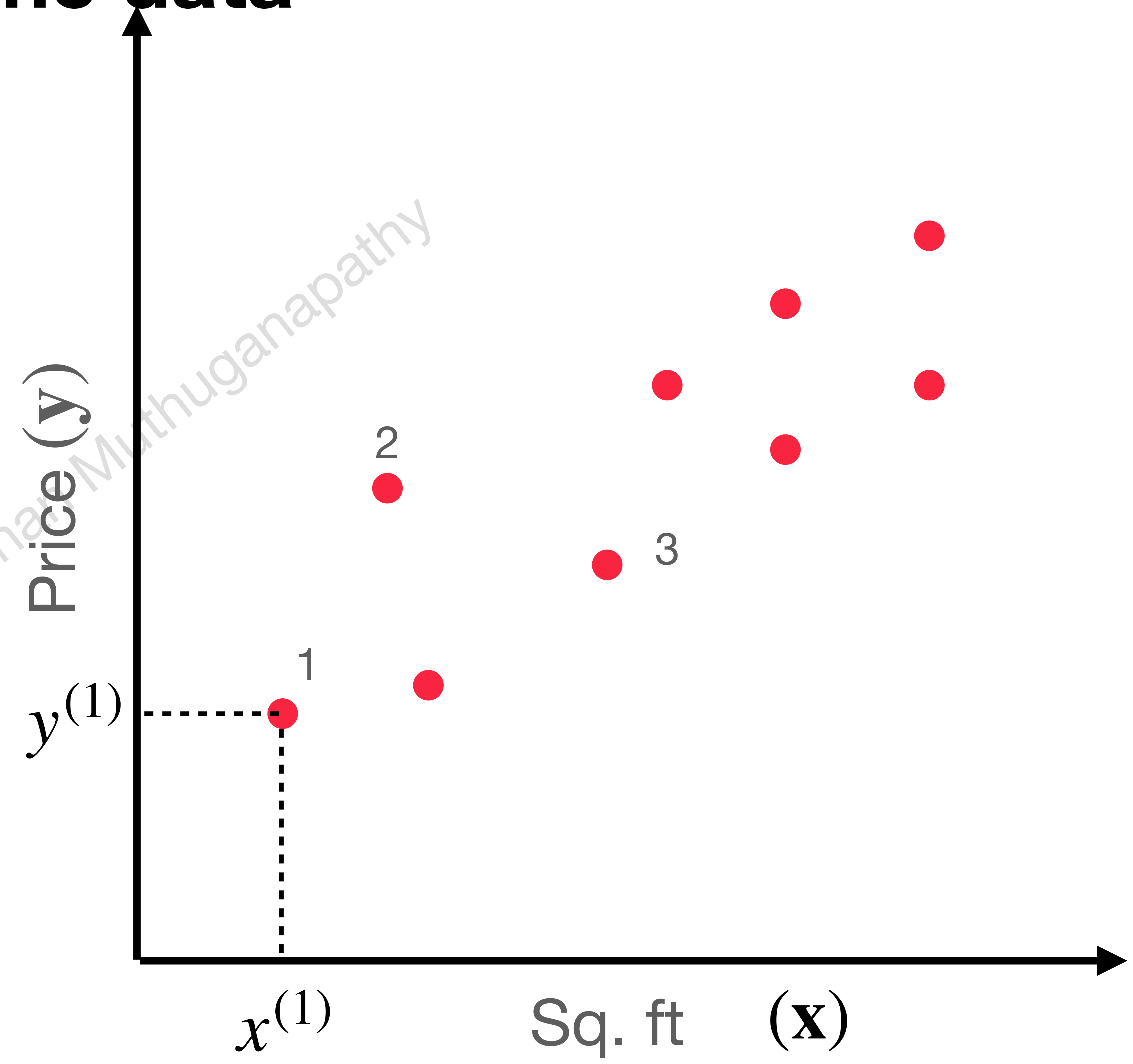
- $\mathbf{x} = (x^{(1)}, x^{(2)}, x^{(3)}, \ldots . x^{(m)})$
- $\mathbf{y} = (y^{(1)}, y^{(2)}, y^{(3)}, \ldots . y^{(m)})$
- Look at the data, a straight line fit is probably good



Linear Regression

Goal: Approximation that fits the data

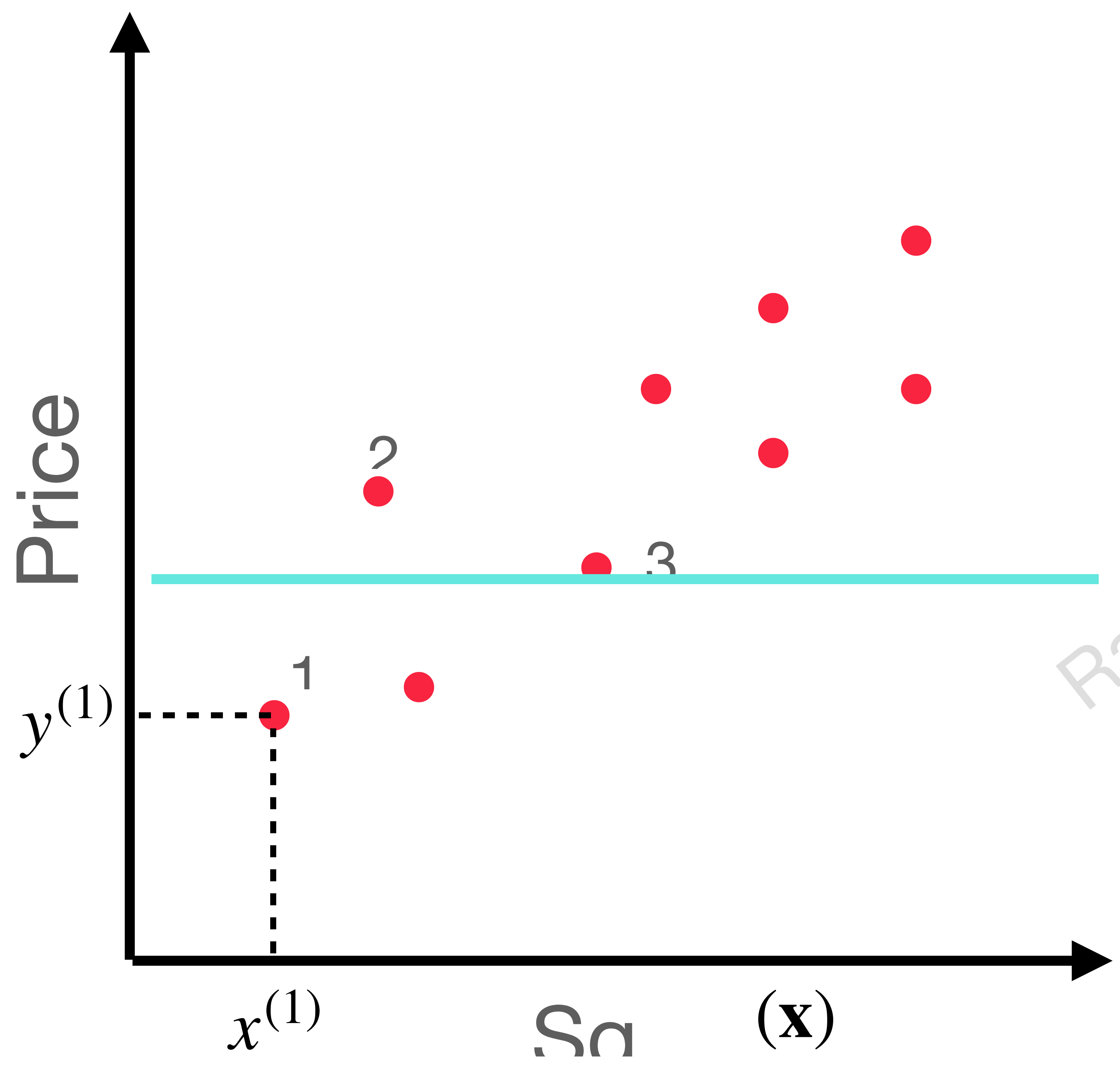
- Hypothesis function
- $h_w(x) = w_0 + w_1 x$
- Goal: Determine weights (w_0, w_1)
- w_0 - bias



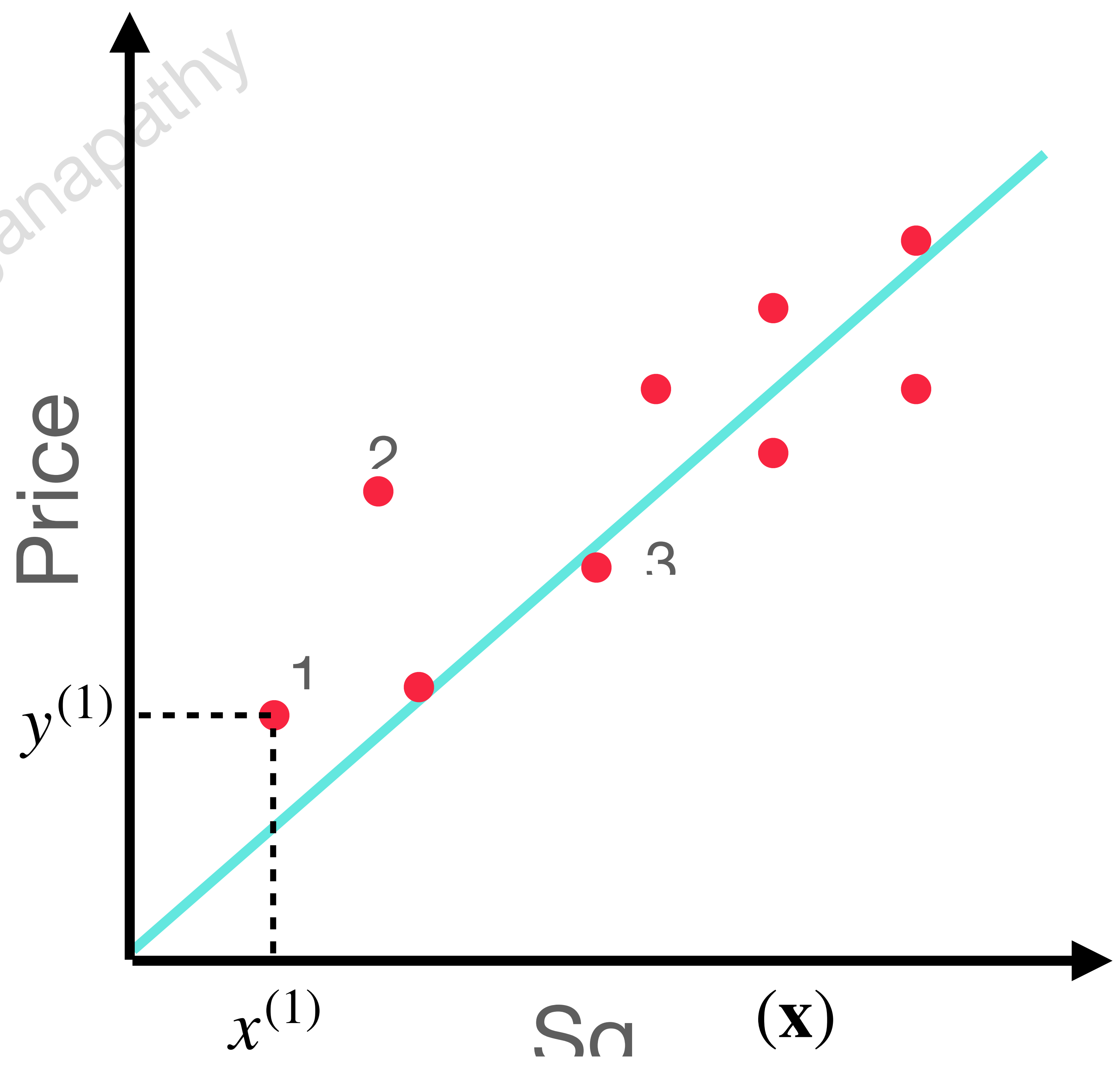
Ramanathan Muthuganapathy, Department of Engineering Design, IIT Madras

Linear Regression

Some candidates



$$(w_0 = const, w_1 = 0)$$

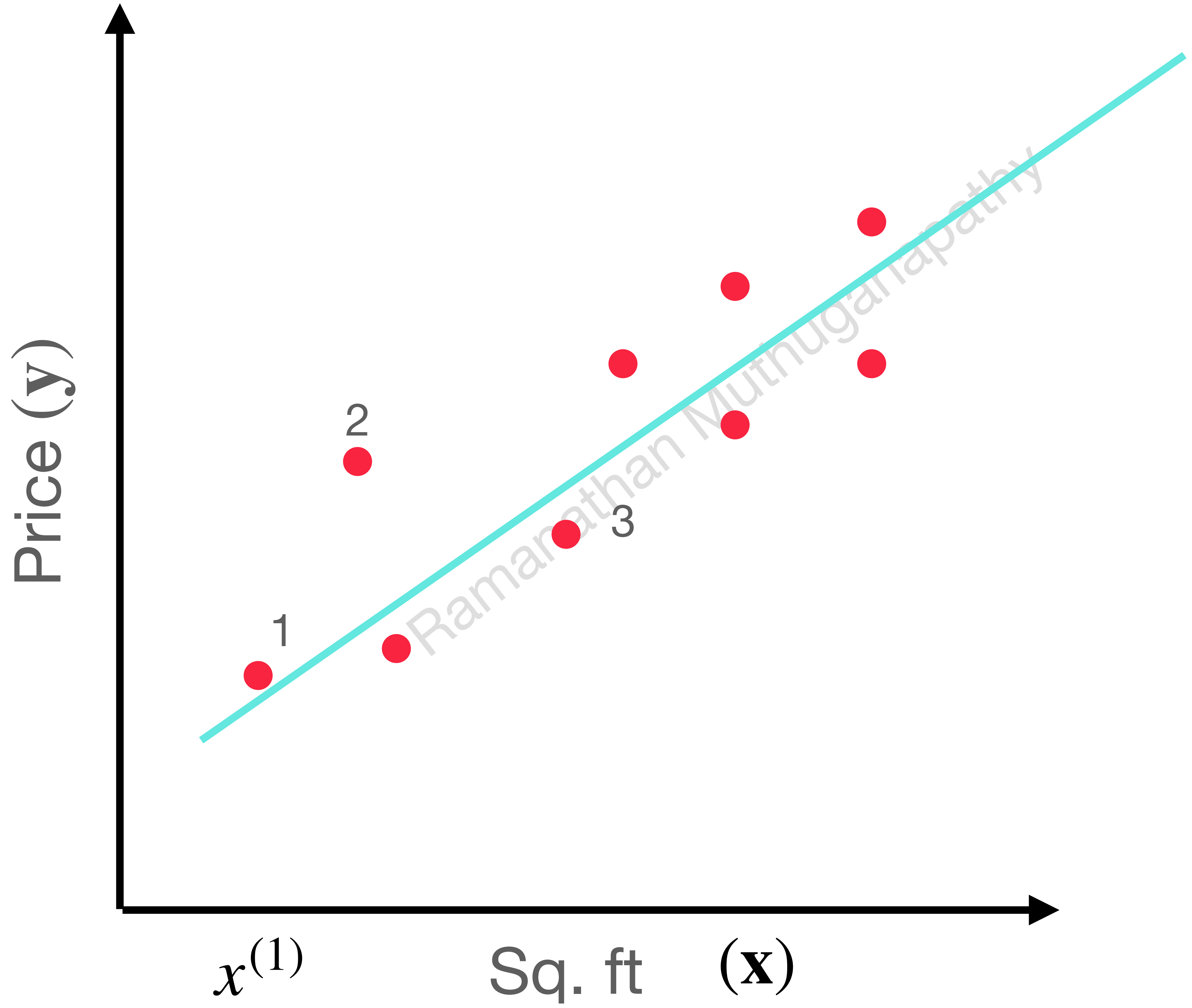


$$(w_0 = 0, w_1 = 1)$$

Ramanathan Muthuganapathy, Department of Engineering Design, IIT Madras

Linear Regression

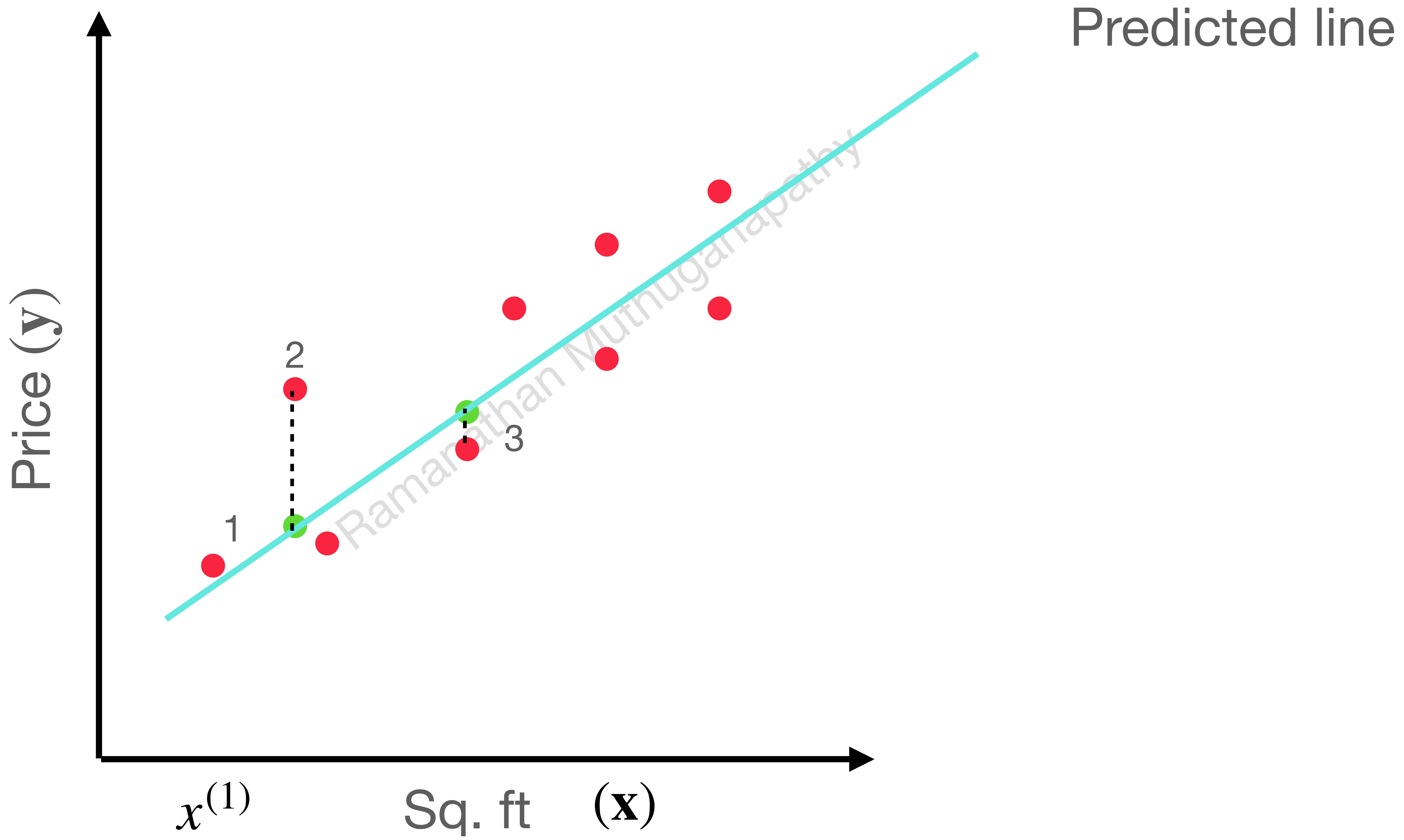
Case of best fit!



$$(w_0 = ?, w_1 = ?)$$

Linear Regression

Cost function



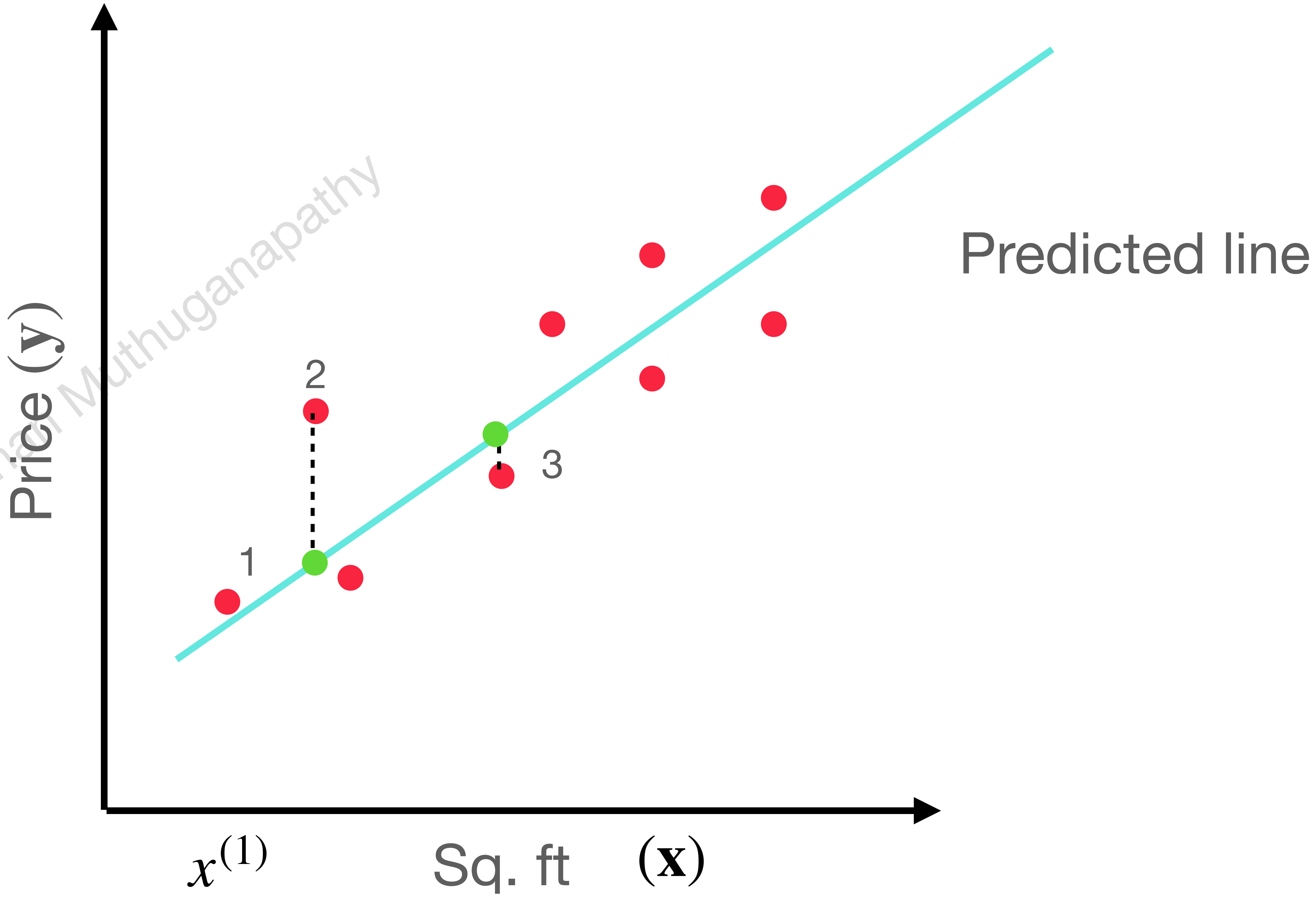
$$(w_0 = ?, w_1 = ?)$$

Ramanathan Muthuganapathy, Department of Engineering Design, IIT Madras

Linear Regression

Predicted values

- $\mathbf{x} = (x^{(1)}, x^{(2)}, x^{(3)}, \ldots . x^{(m)})$
- $\mathbf{y} = (y^{(1)}, y^{(2)}, y^{(3)}, \ldots . y^{(m)})$
- $\bar{\mathbf{y}} = (\bar{y}^{(1)}, \bar{y}^{(2)}, \bar{y}^{(3)}, \ldots . \bar{y}^{(m)})$
- $\bar{y}^{(i)} = h_w(x^{(i)}) = w_0 + w_1 x^{(i)}$
- Goal: Determine weights (w_0, w_1)



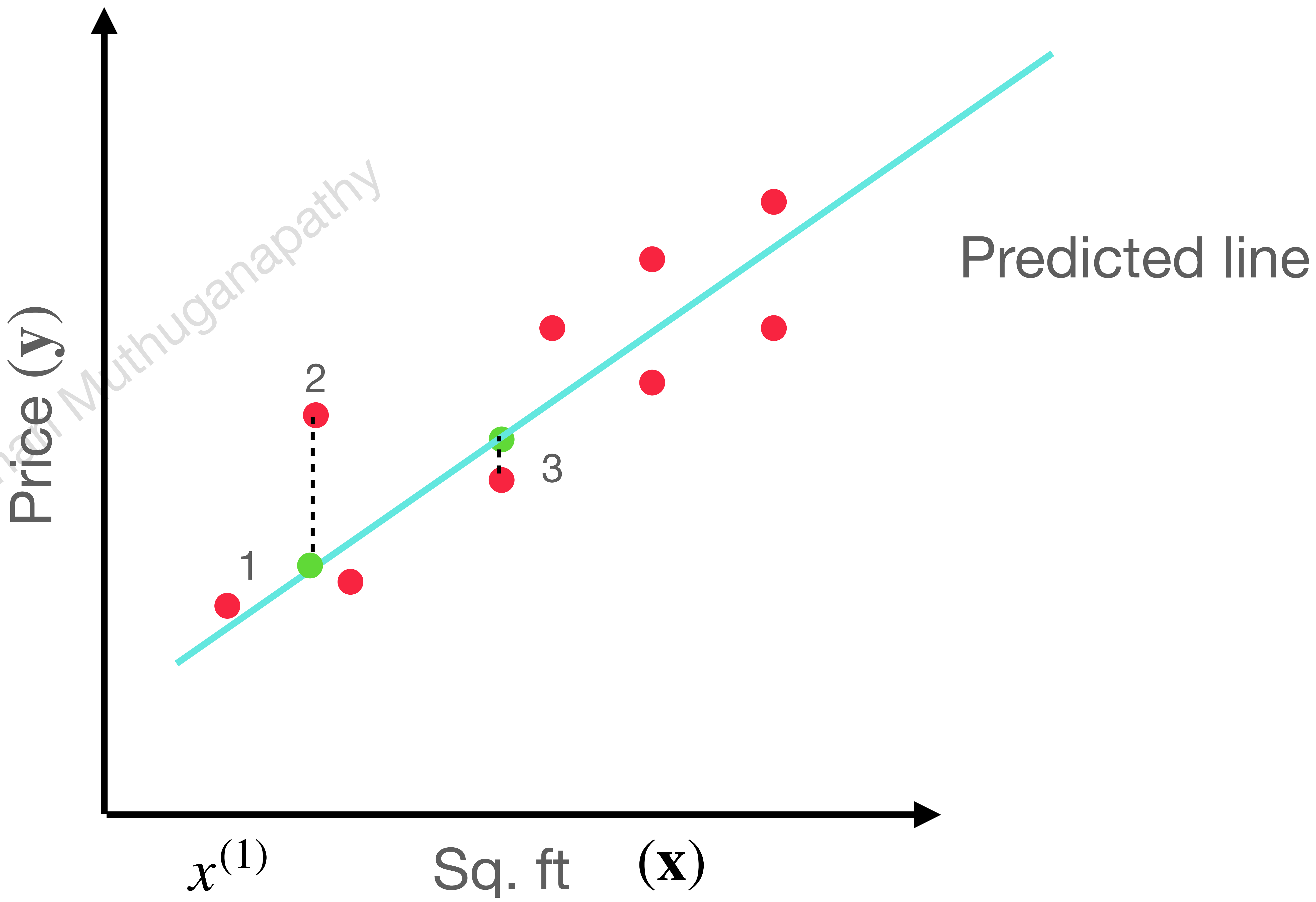
$(w_0 = ?, w_1 = ?)$

Ramanathan Muthuganapathy, Department of Engineering Design, IIT Madras

Linear Regression

Cost function

- Minimize the distance between $(\mathbf{y}, \bar{\mathbf{y}})$
- $(\bar{y}^{(i)} - y^{(i)})^2$ - for every sample
- Compute the sum
- Take the average

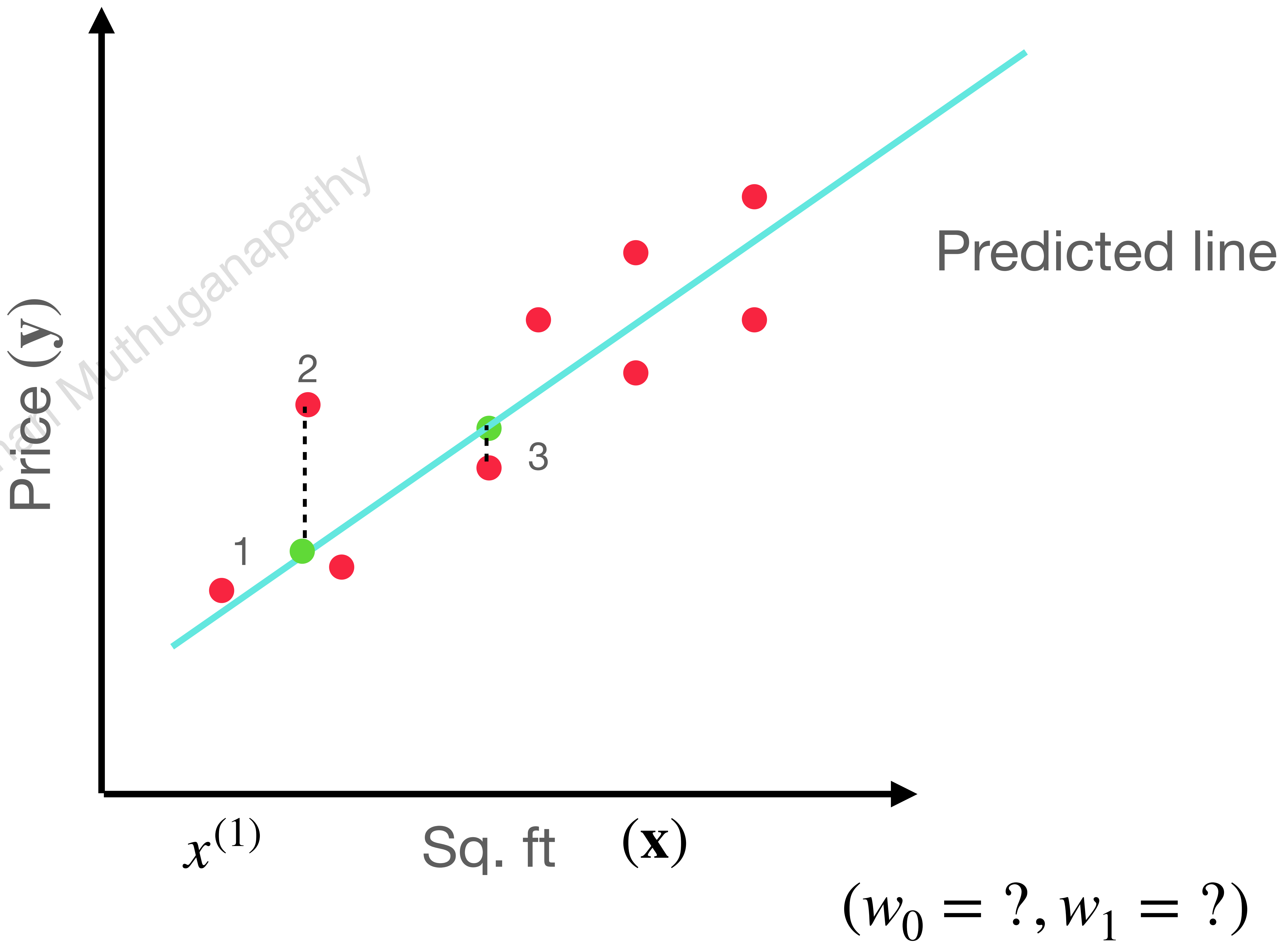


$(w_0 = ?, w_1 = ?)$

Linear Regression

Cost function

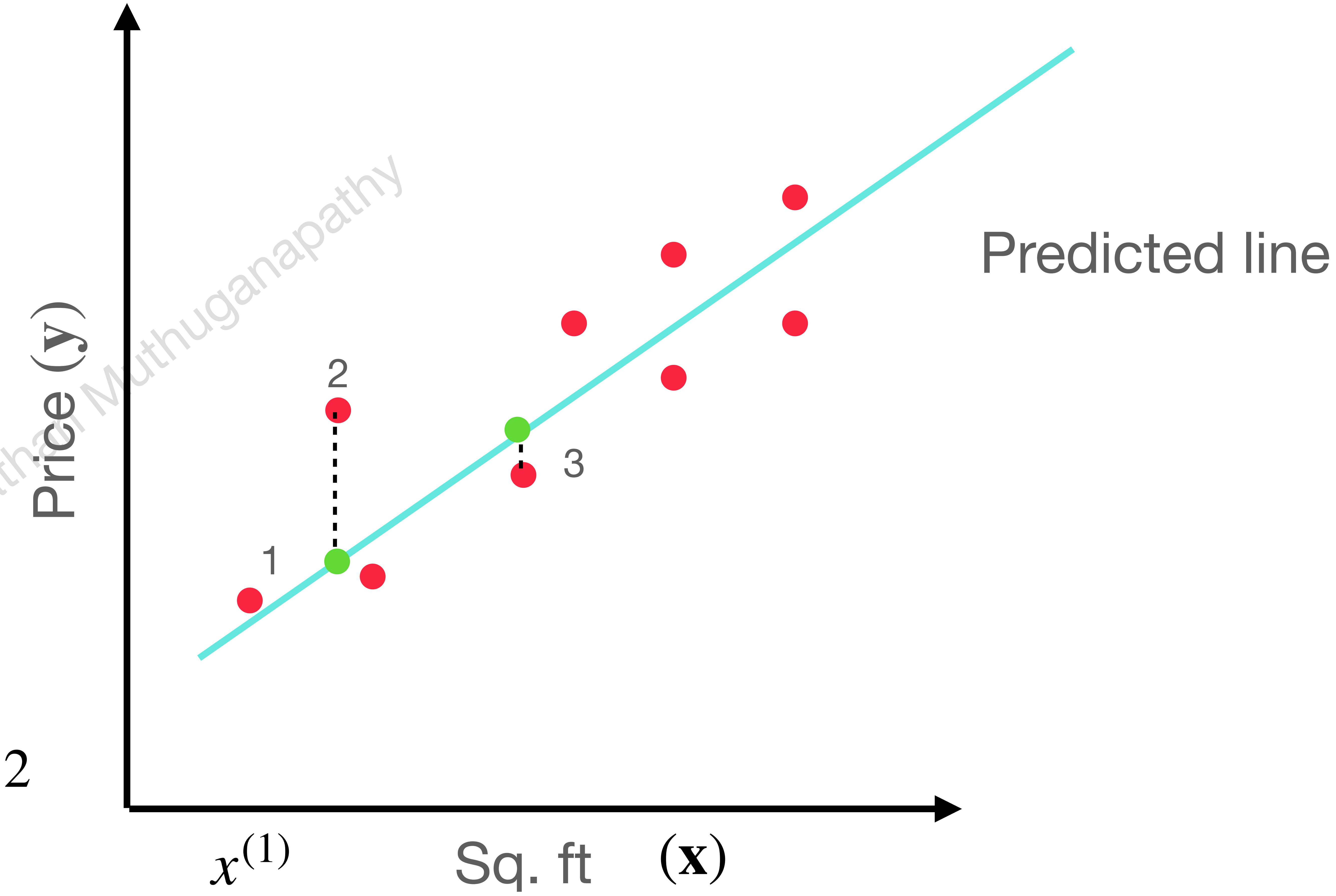
- Minimize the distance between $(\mathbf{y}, \bar{\mathbf{y}})$
- $J(\mathbf{y}, \bar{\mathbf{y}}) = \sum_{i=1}^{m} \frac{1}{2m} (\bar{y}^{(i)} - y^{(i)})^2$



Linear Regression

Cost function

- Minimize the distance between $(\mathbf{y}, \bar{\mathbf{y}})$
- $J(w) = J(\mathbf{y}, \bar{\mathbf{y}}) = J(\mathbf{y}, h(w))$
- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(\bar{y}^{(i)} - y^{(i)})^2$
- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(h_w(x^{(i)}) - y^{(i)})^2$



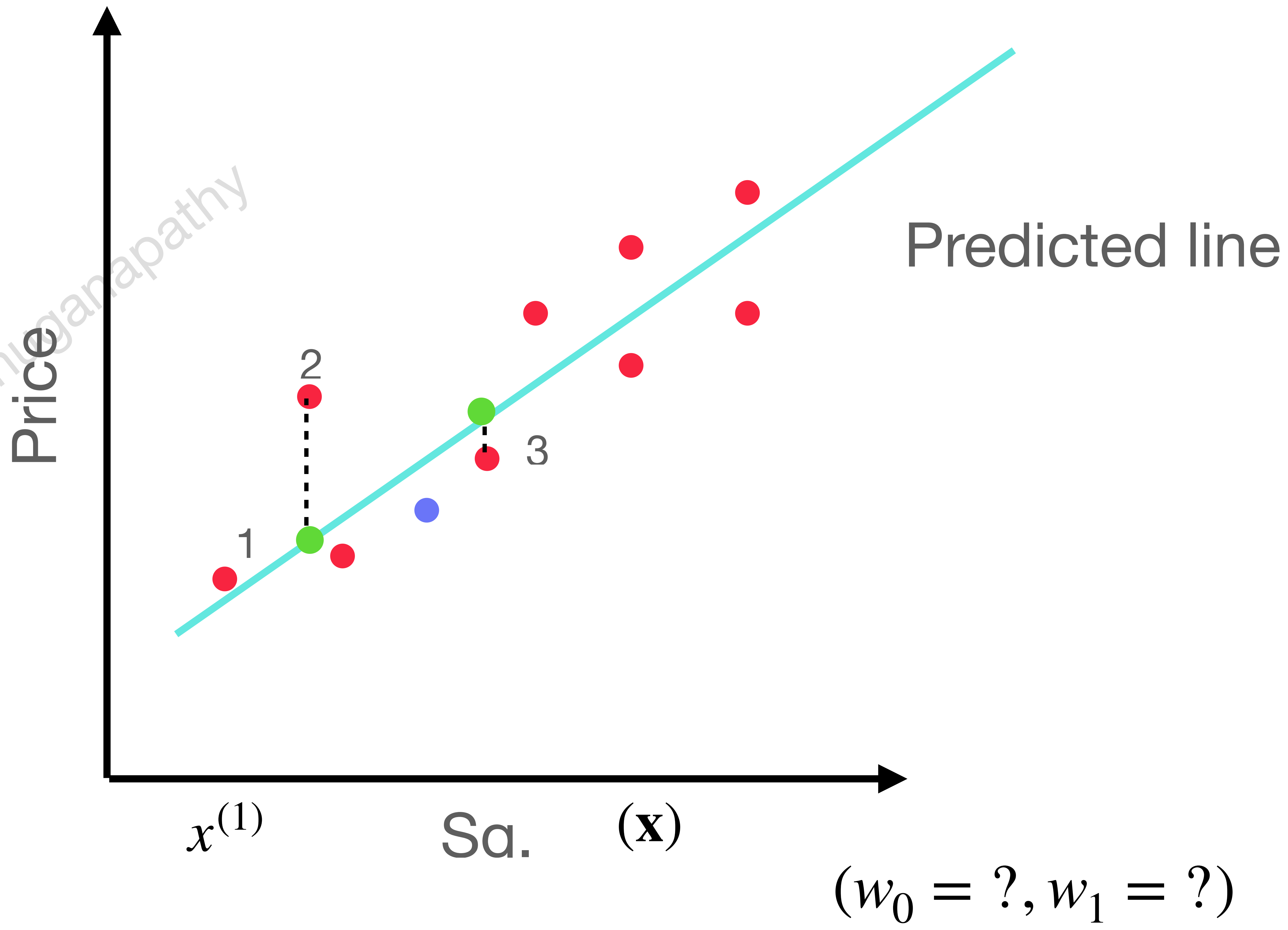
$(w_0 = ?, w_1 = ?)$

Ramanathan Muthuganapathy, Department of Engineering Design, IIT Madras

Linear Regression

Cost function

- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(h_w(x^{(i)}) - y^{(i)})^2$
- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$

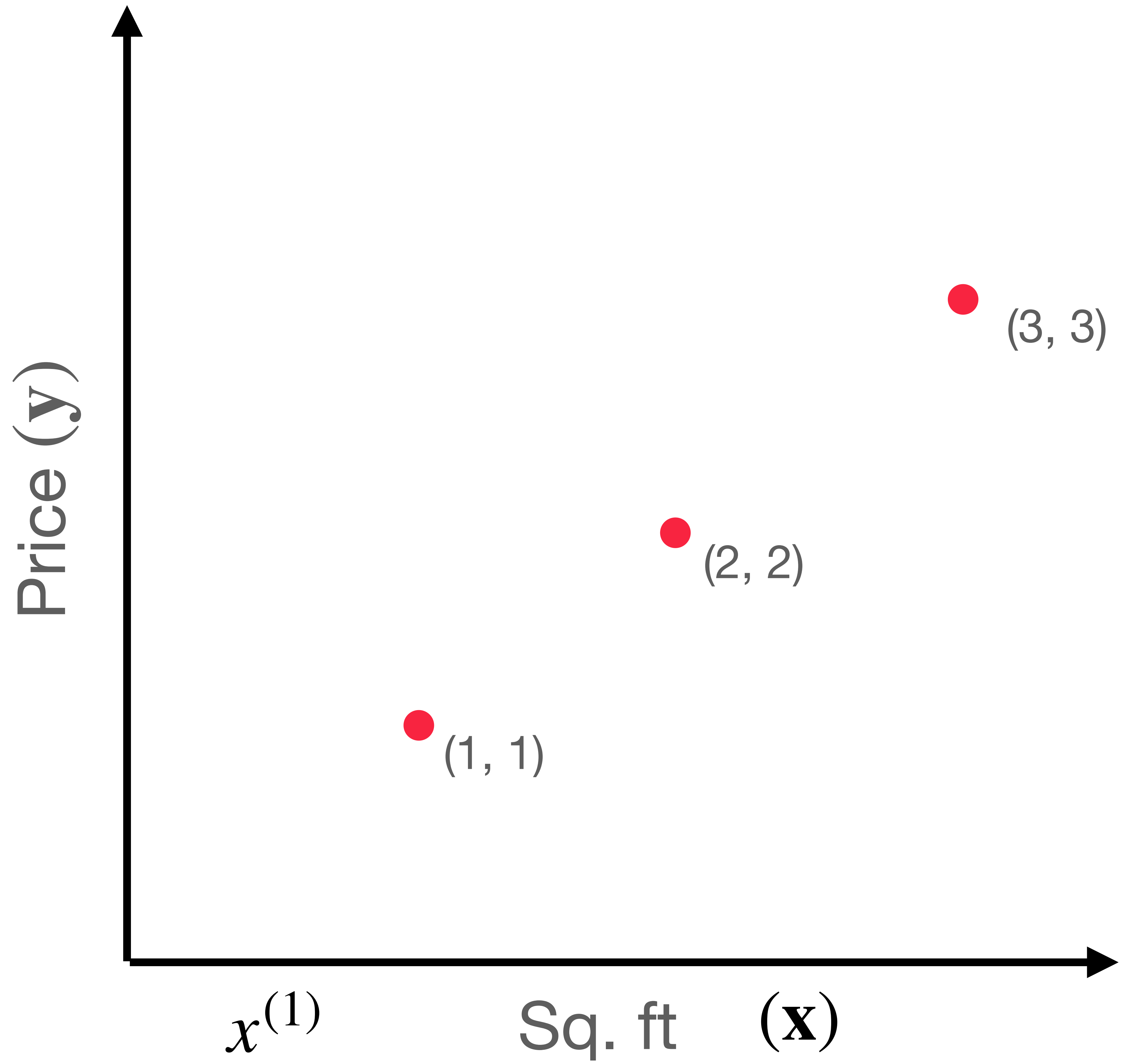


Linear Regression

Minimize the cost function

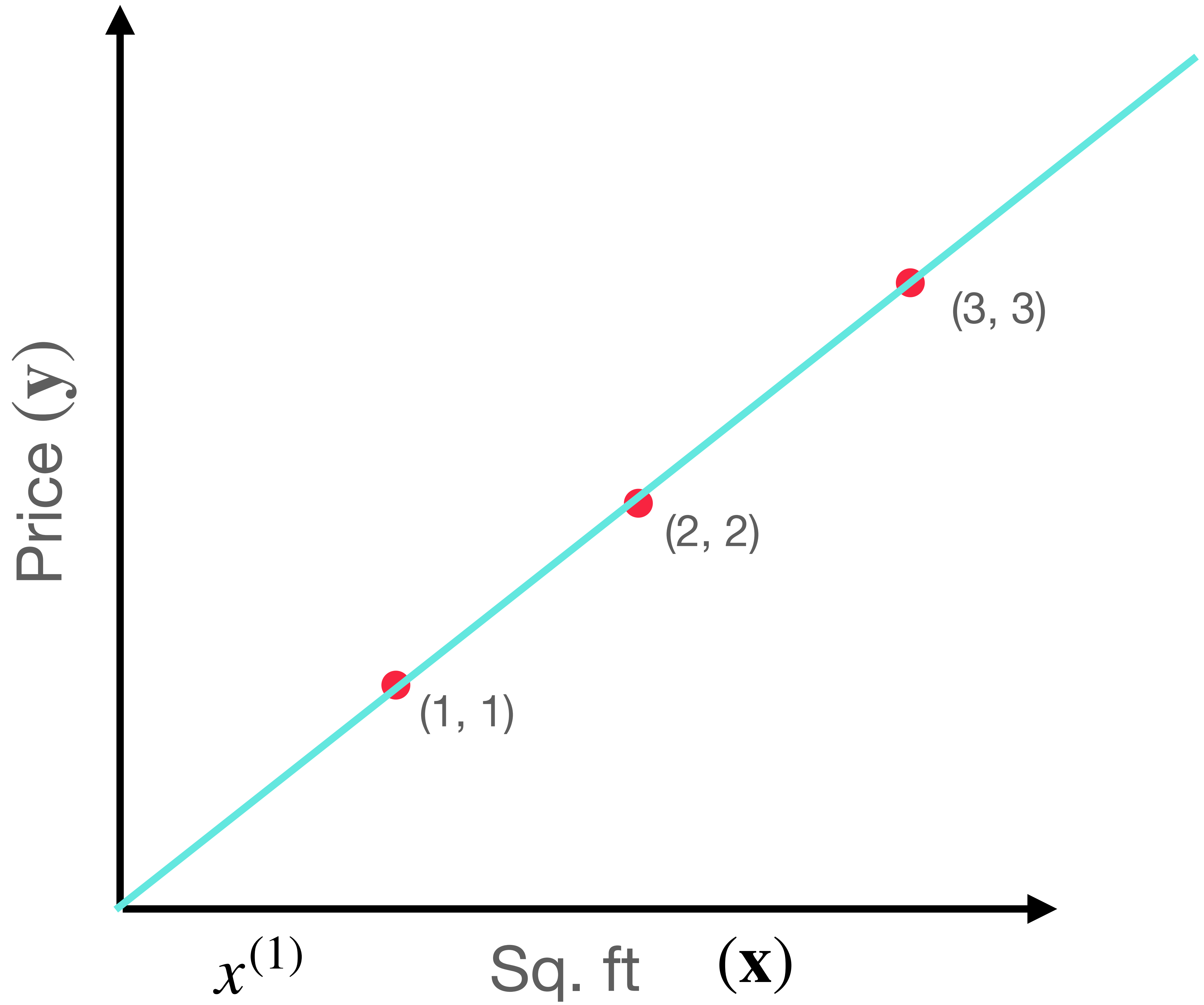
- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$
- min $J(w)$

Example data



$(w_0 = ?, w_1 = ?)$

Example data



$(w_0 = ?, w_1 = ?)$

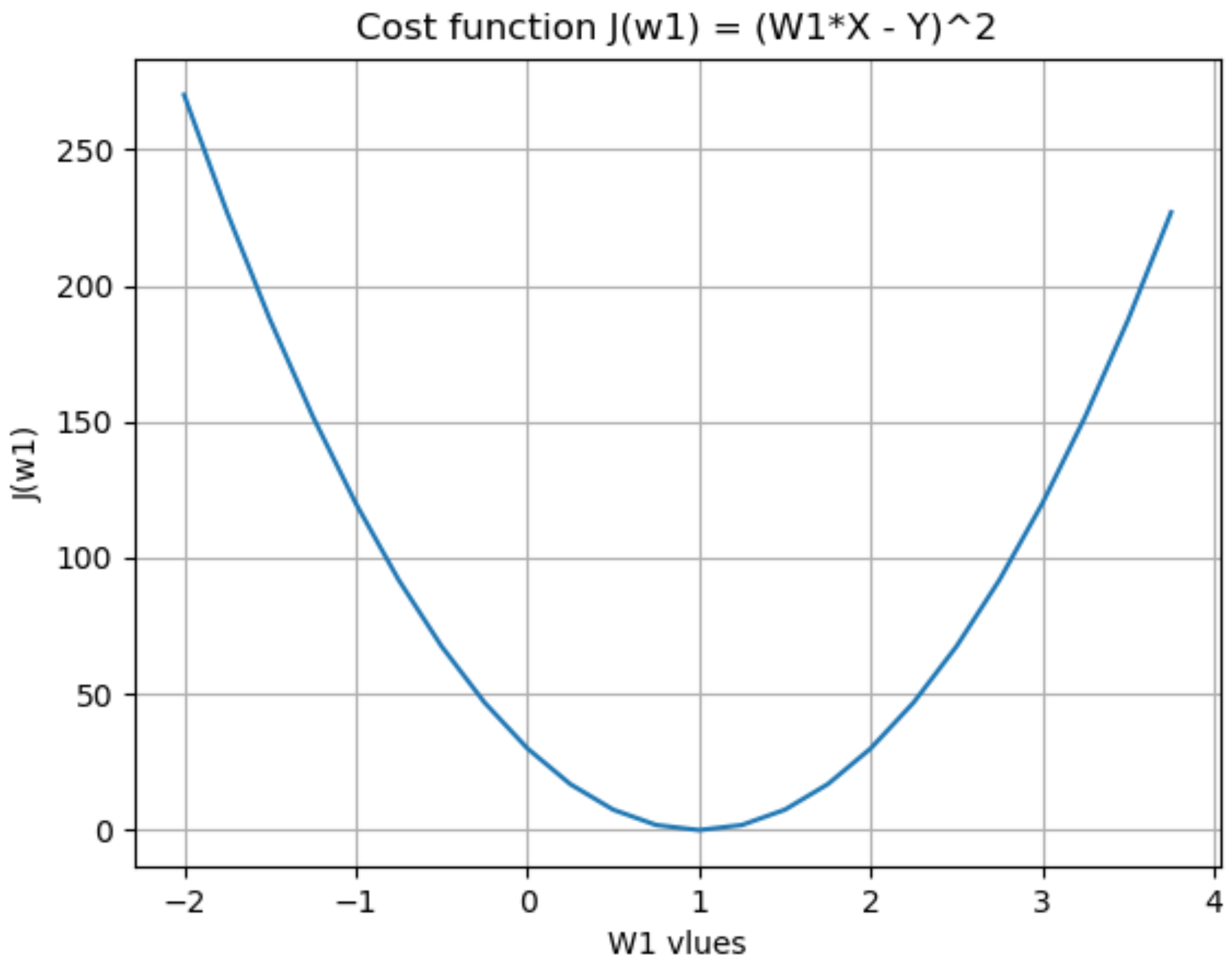
Linear Regression

Plotting the cost function

- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$
-

Linear Regression - Cost function

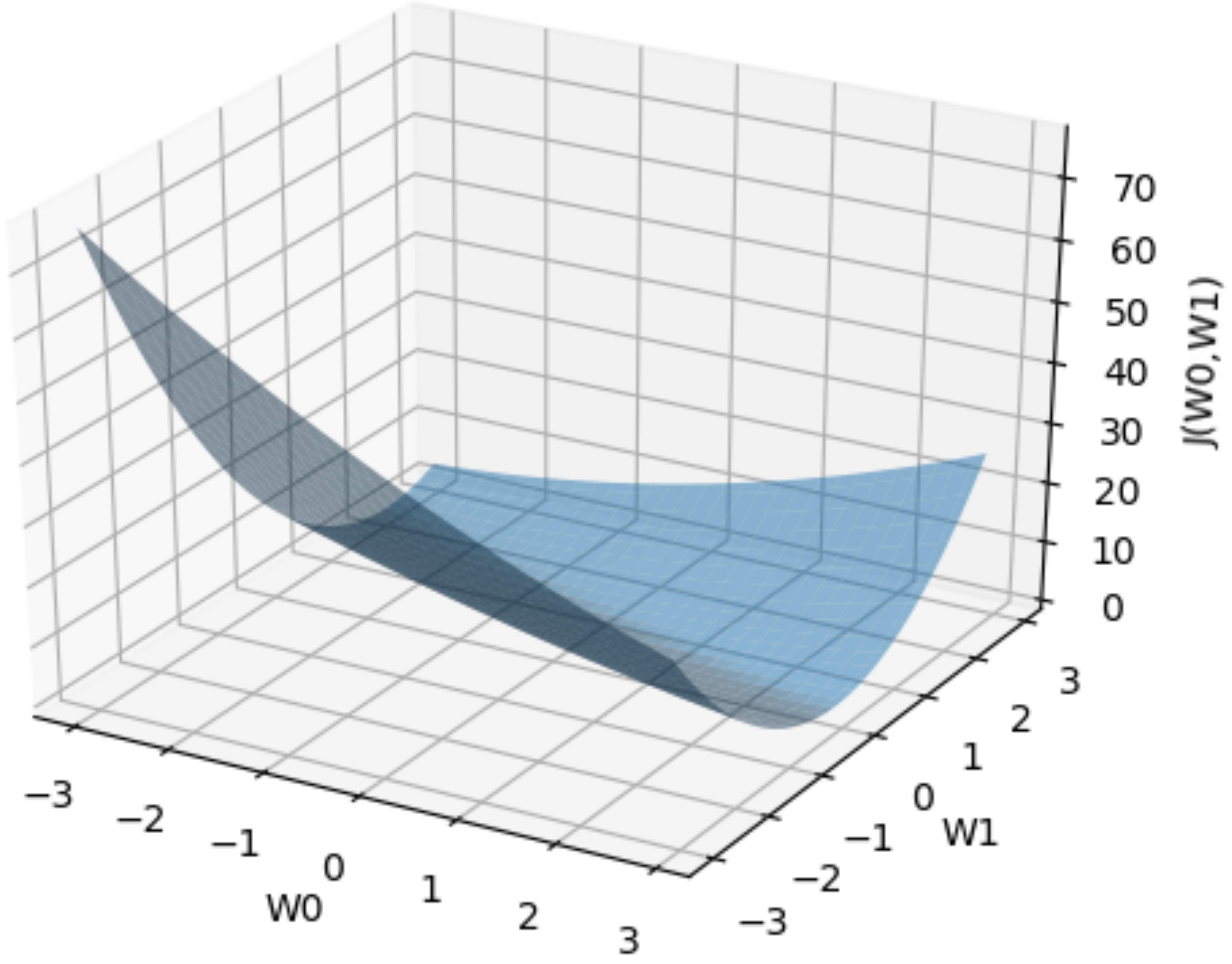
$J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$ $(w_0 = 0, w_1 =$ **varying)**



Ramanathan Muthuganapathy, Department of Engineering Design, IIT Madras

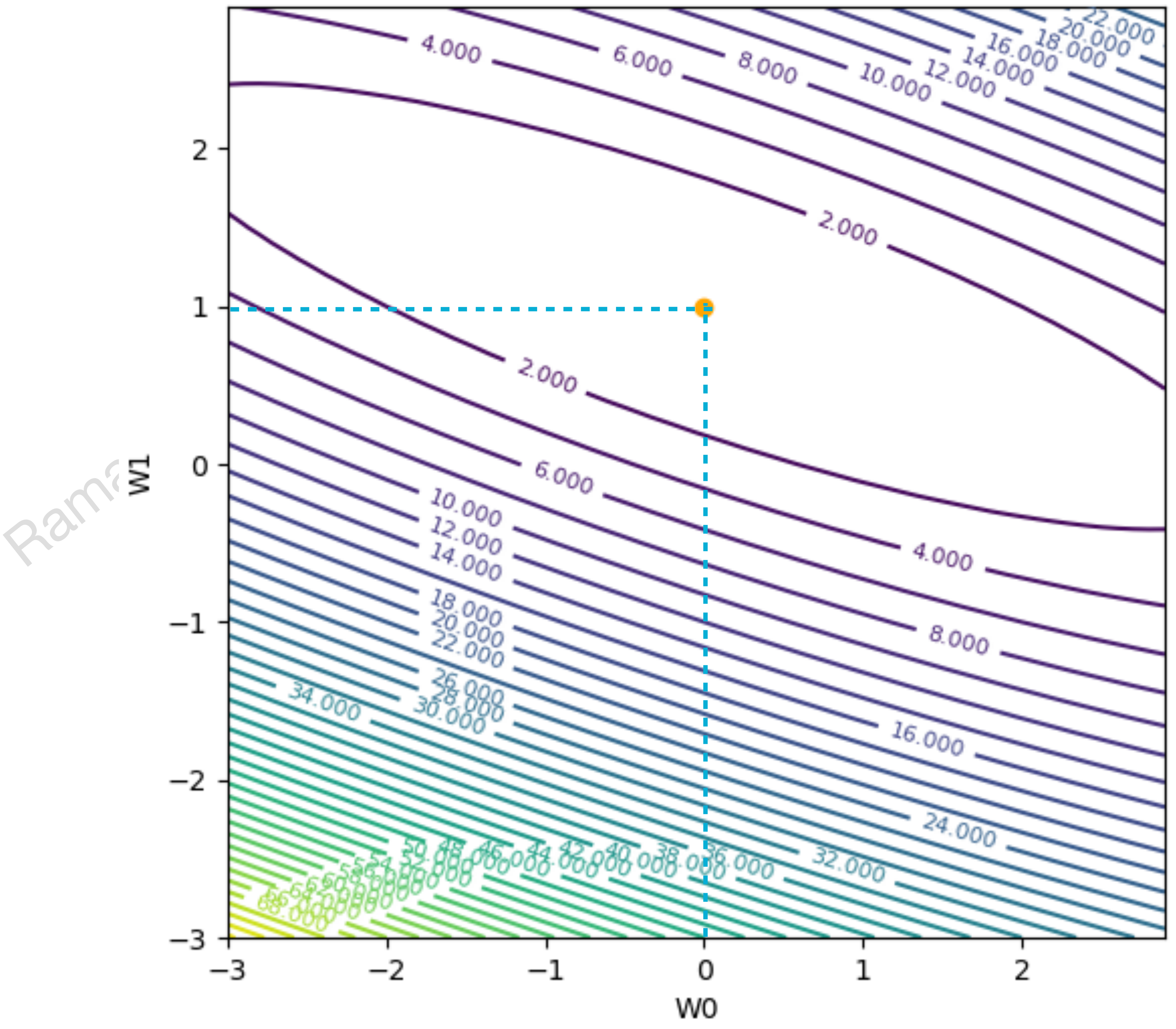
Linear Regression - Cost function

$J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$ w_0 & w_1 **are varying)**



Linear Regression - Cost function

$J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$ w_0 & w_1 **are varying)**



Linear Regression

Gradient descent

- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$
- Find $\nabla J(w_0, w_1) = \left(\frac{\partial J}{\partial w_0}, \frac{\partial J}{\partial w_1}\right)$

Linear Regression
Gradient descent

- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$
- Find $\nabla J(w_0, w_1) = \left(\frac{\partial J}{\partial w_0}, \frac{\partial J}{\partial w_1}\right)$
- $\frac{\partial J}{\partial w_0} = \frac{1}{m} \sum_{i=1}^{m} (w_0 + w_1 x^{(i)} - y^{(i)})$
- $\frac{\partial J}{\partial w_1} = \frac{1}{m} \sum_{i=1}^{m} (w_0 + w_1 x^{(i)} - y^{(i)}) x^{(i)}$

Linear Regression

Gradient descent

- $J(w) = \sum_{i=1}^{m} \frac{1}{2m}(w_0x^{(0)} + w_1x^{(i)} - y^{(i)})^2, x^{(0)} = 1$
- Find $\nabla J(w_0, w_1) = \left(\frac{\partial J}{\partial w_0}, \frac{\partial J}{\partial w_1}\right)$
- $\frac{\partial J}{\partial w_0} = \frac{1}{m}\sum_{i=1}^{m}(w_0x^{(0)} + w_1x^{(i)} - y^{(i)})x^{(0)}$
- $\frac{\partial J}{\partial w_1} = \frac{1}{m}\sum_{i=1}^{m}(w_0x^{(0)} + w_1x^{(i)} - y^{(i)})x^{(i)}$

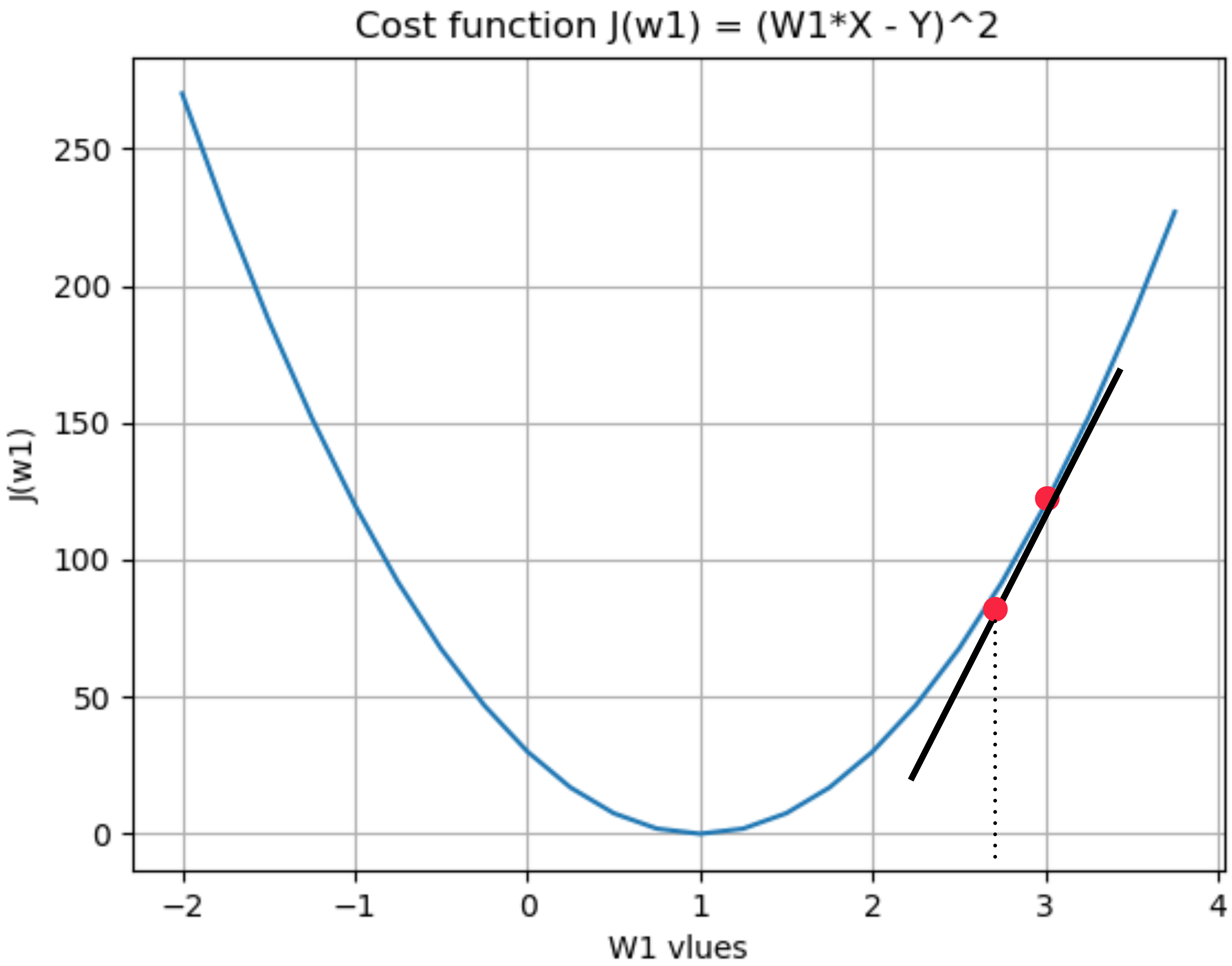
Gradient descent

- Starting point $w* = (w_0^*, w_1^*)$
- Compute $J, \ - \nabla J$ at $w_k^* = w^*$.
- Update w's
  - $w_{k+1}^* = w_k^* - \alpha_k \nabla J$ (Fix a value for α_k (= 0.01), learning rate)
- Check for stopping criteria
- Else continue the iteration

Gradient descent

Learning rate

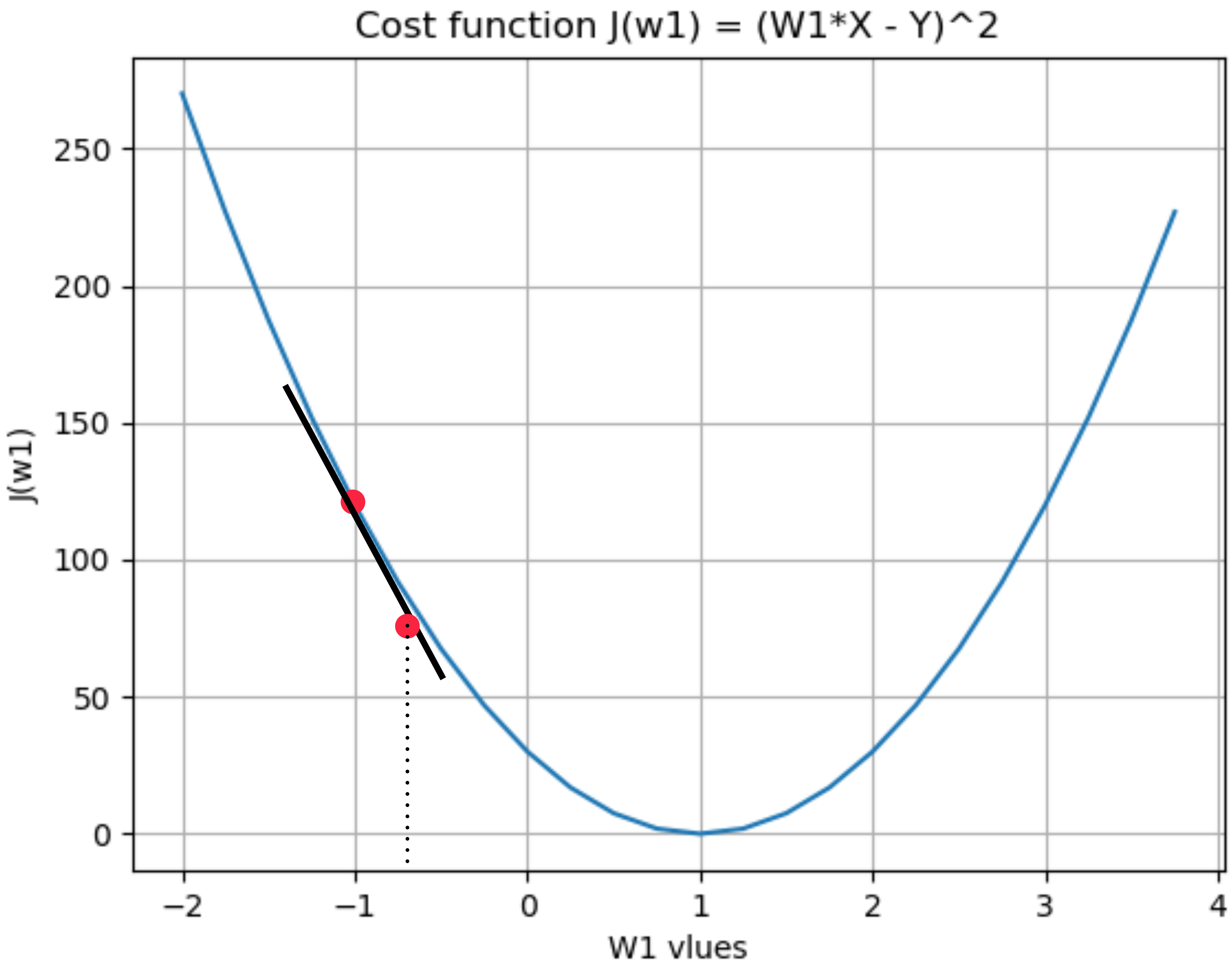
- Update w's
  - $w_1^{k+1} = w_1^k - \alpha_k \nabla J$
  - w_1^{k+1} will decrease



Gradient descent

Learning rate

- Update w's
  - $w_1^{k+1} = w_1^k - \alpha_k \nabla J$
  - w_1^{k+1} will increase



Machine Learning Refined

- https://github.com/jermwatt/machine_learning_refined