Review about Training

Teeradaj Racharak (เอ็กซ์)
r.teeradaj@gmail.com

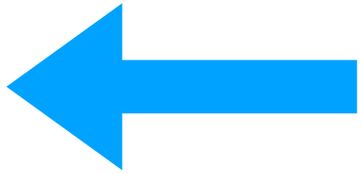
Recap

- Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

- Parameters:

$$\theta_0, \theta_1$$



These will be trained !

- Cost Function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \Sigma_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

- Goal:

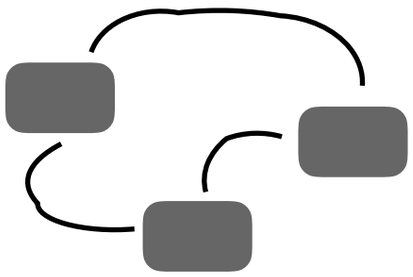
$$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

How are they trained?

θ_0, θ_1
(Initialized with some values)

How are they trained?

TF Graph

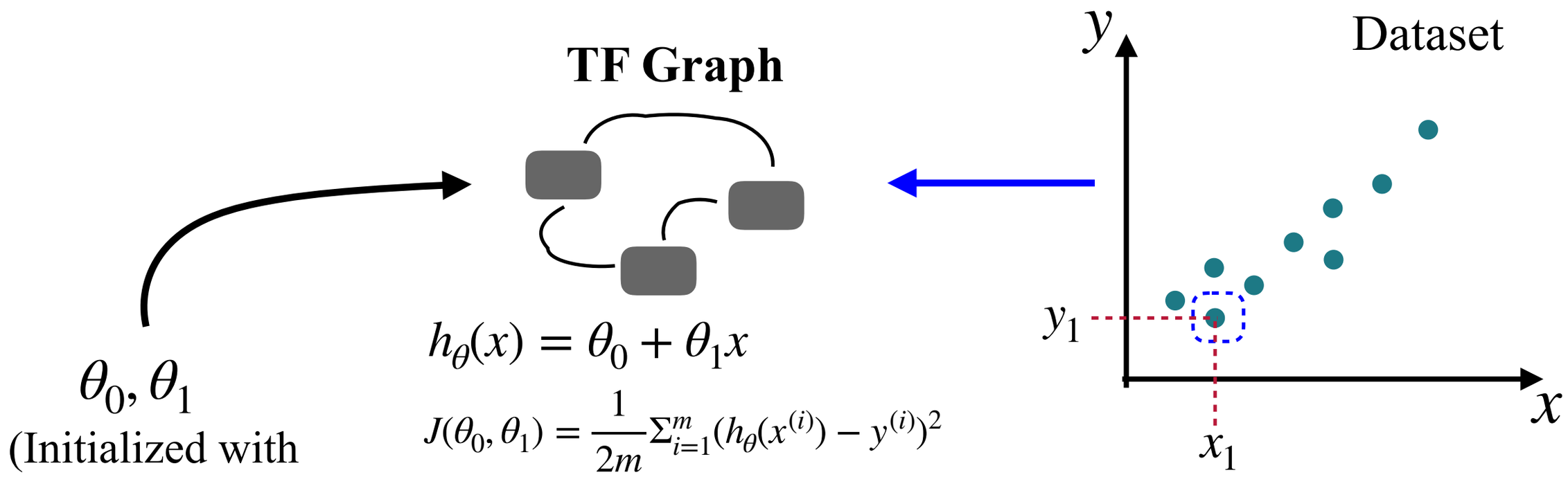


$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m}\Sigma_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

θ_0, θ_1
(Initialized with some values)

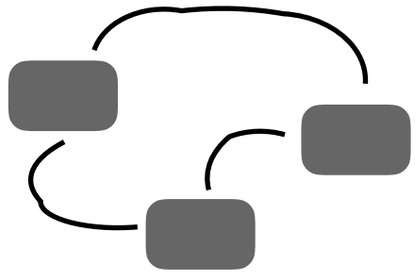
How are they trained?



How are they trained?

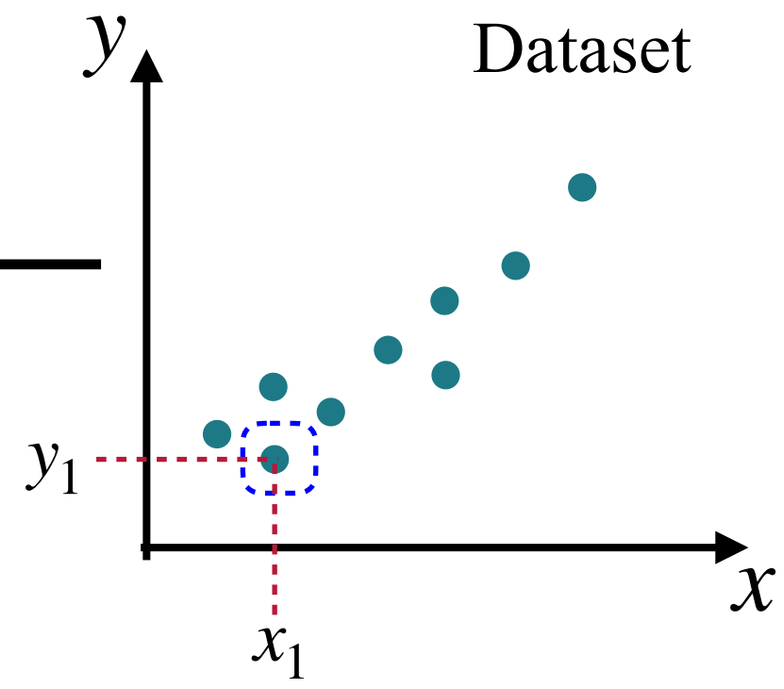
θ_0, θ_1
(Initialized with some values)

TF Graph

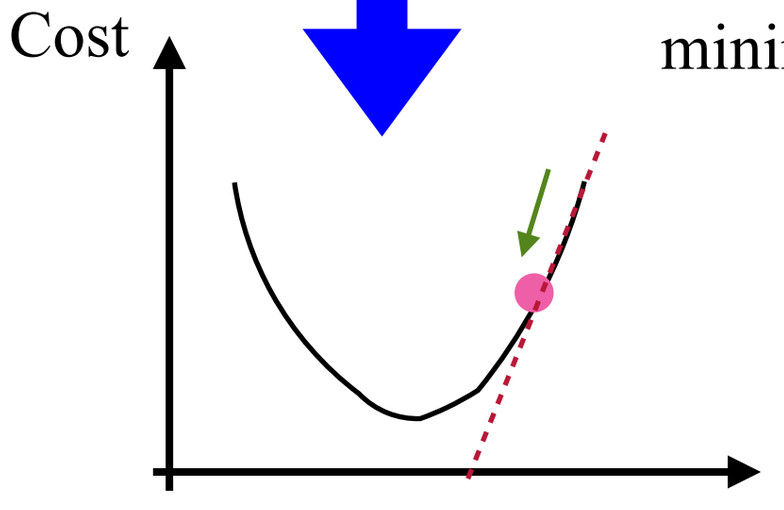


$h_\theta(x) = \theta_0 + \theta_1 x$

$J(\theta_0, \theta_1) = \frac{1}{2m}\Sigma_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$



Train: tf.train.gradientDescentOptimizer(0.01).
minimize(cost_function)



Tweak θ_0, θ_1

Reduce cost for (x_1, y_1)

How are they trained?

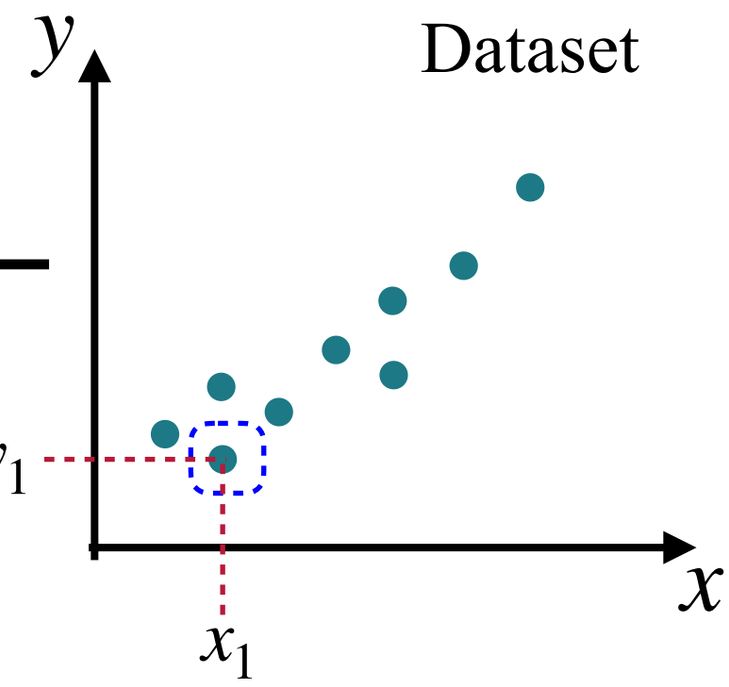
TF Graph

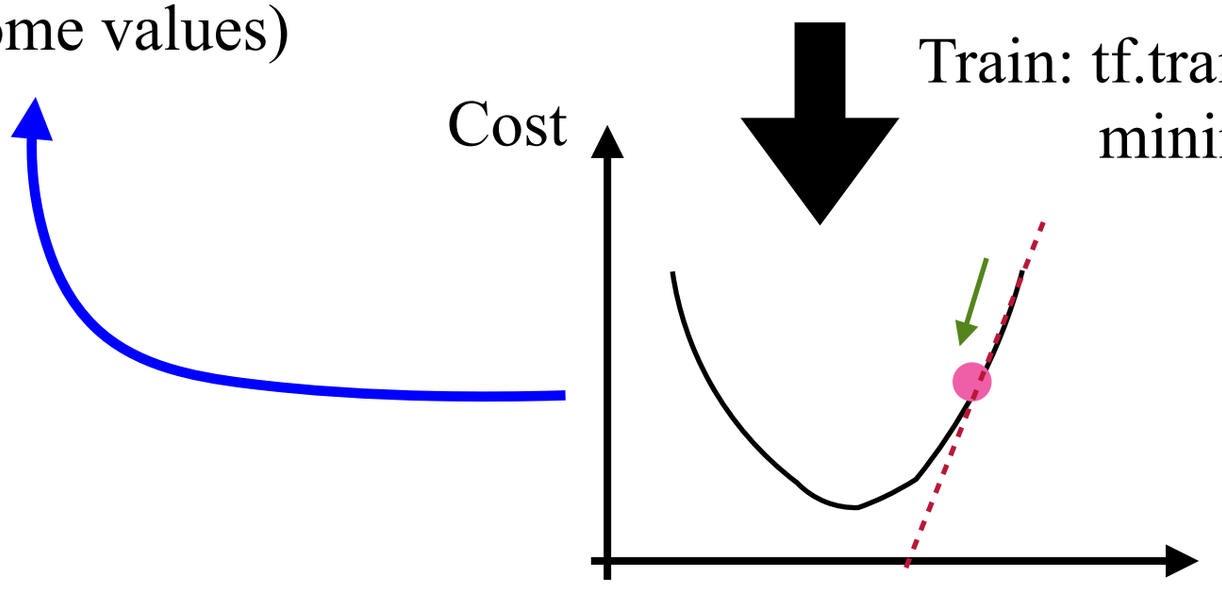
$h_\theta(x) = \theta_0 + \theta_1 x$

$J(\theta_0, \theta_1) = \frac{1}{2m}\Sigma_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$

θ_0, θ_1
(Initialized with some values)

Train: tf.train.gradientDescentOptimizer(0.01).minimize(cost_function)





Tweak θ_0, θ_1

Reduce cost for (x_1, y_1)

This is 1 training step

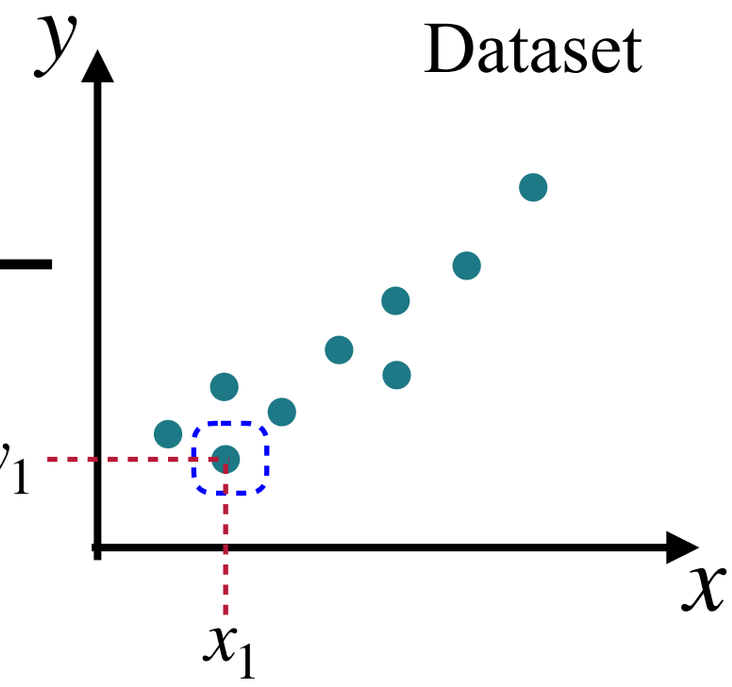
TF Graph

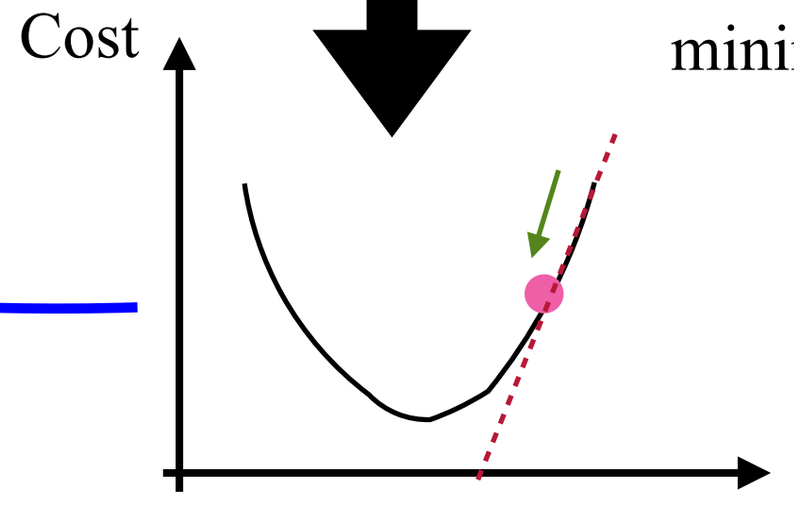
$h_\theta(x) = \theta_0 + \theta_1 x$

$J(\theta_0, \theta_1) = \frac{1}{2m}\Sigma_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$

θ_0, θ_1
(Initialized with some values)

Train: tf.train.gradientDescentOptimizer(0.01).minimize(cost_function)





Tweak θ_0, θ_1

Reduce cost for (x_1, y_1)

Begin 2nd training step

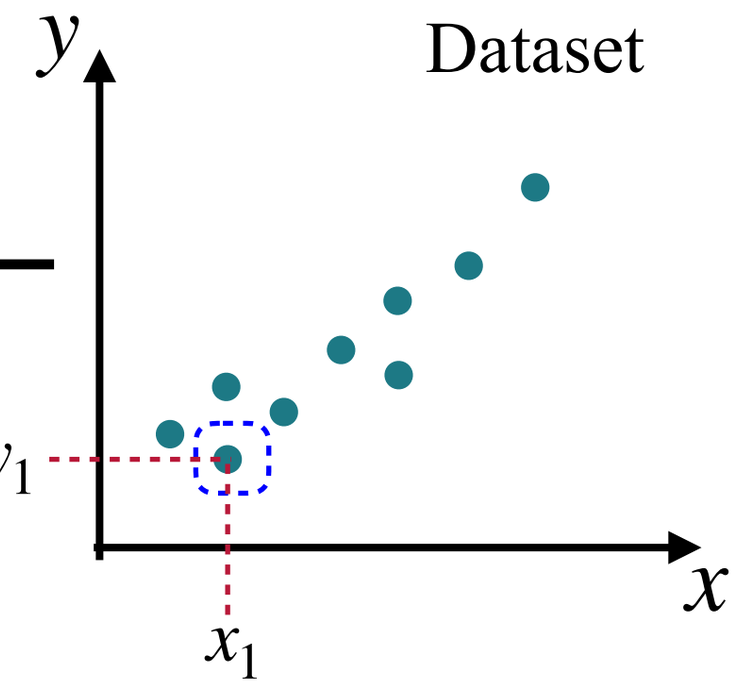
TF Graph

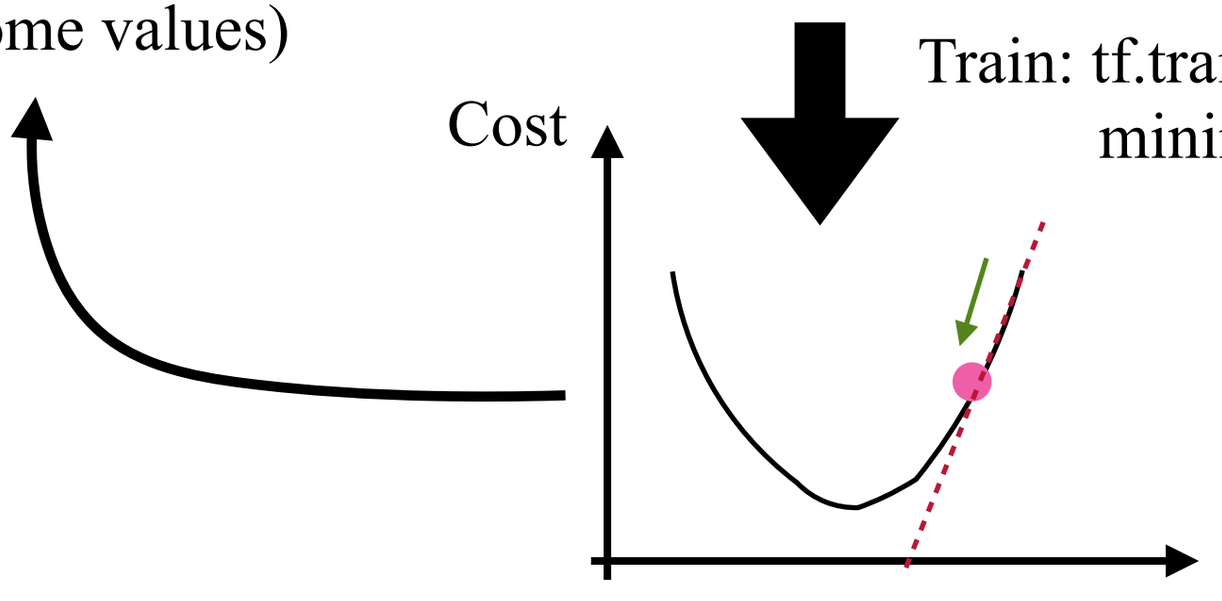
$h_\theta(x) = \theta_0 + \theta_1 x$

$J(\theta_0, \theta_1) = \frac{1}{2m}\Sigma_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$

θ_0, θ_1
(Initialized with some values)

Train: tf.train.gradientDescentOptimizer(0.01).minimize(cost_function)

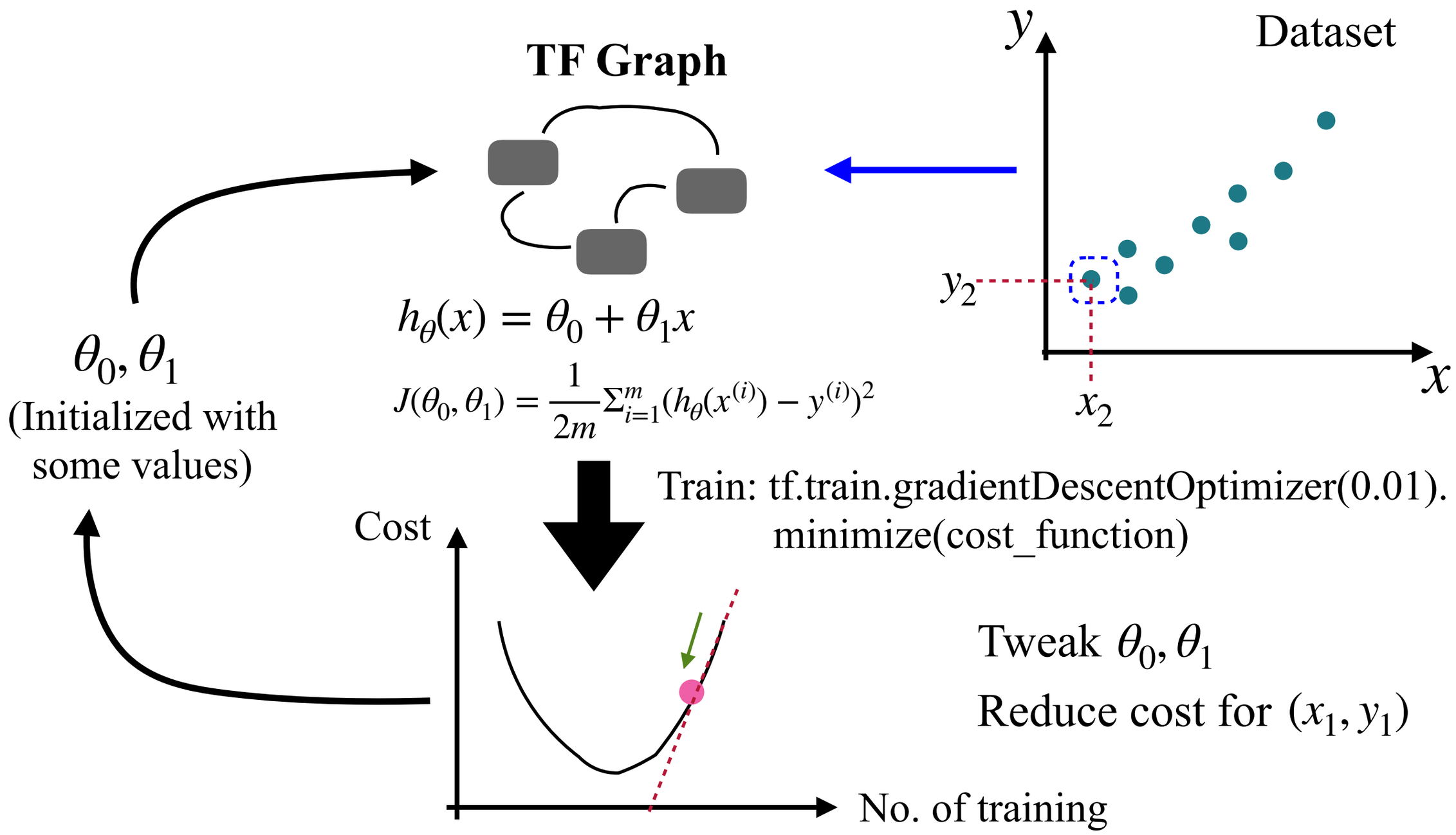




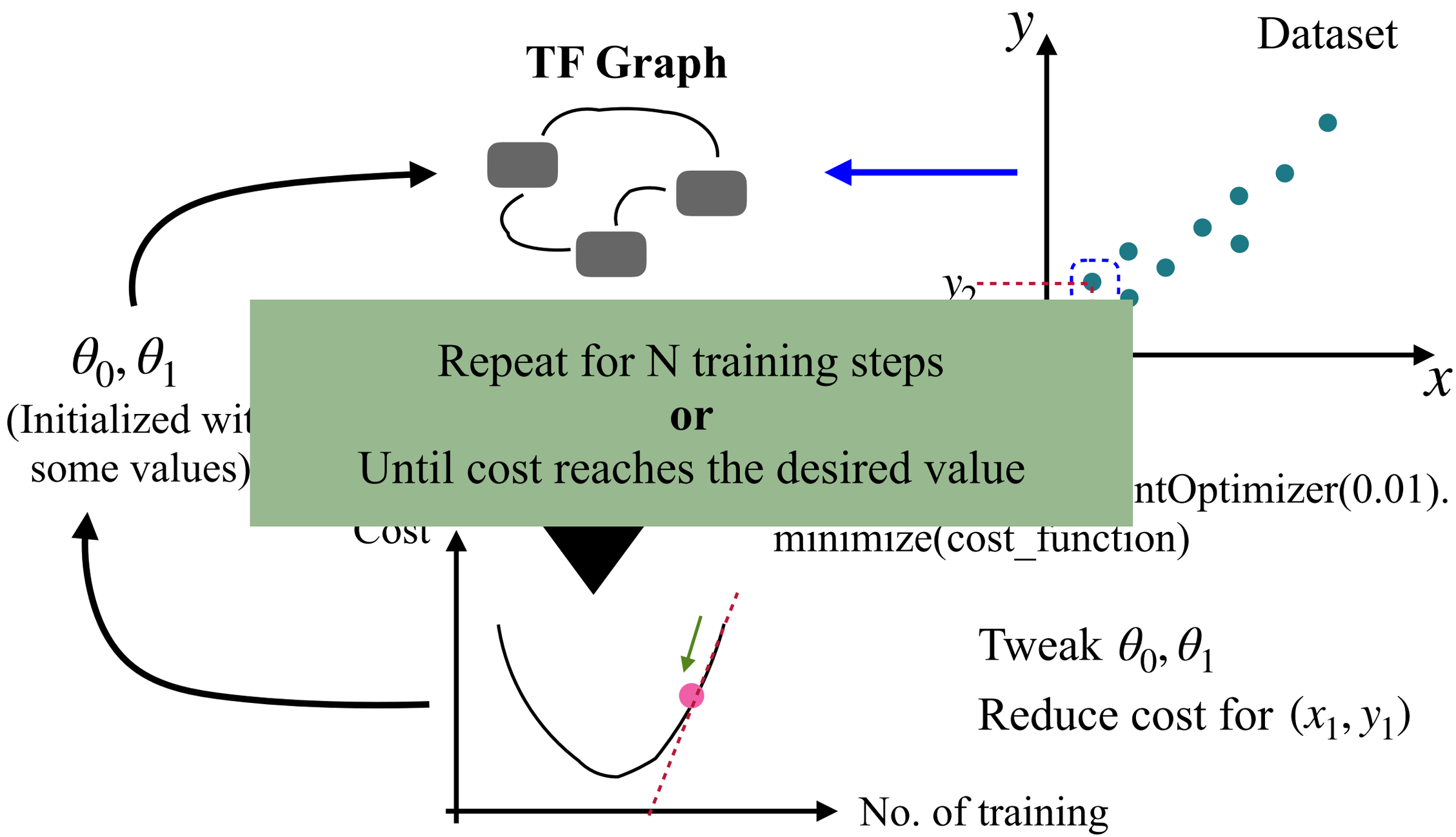
Tweak θ_0, θ_1

Reduce cost for (x_1, y_1)

Begin 2nd training step

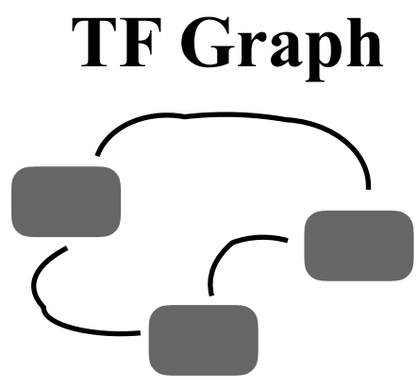


Begin 2nd training step



Batch Size

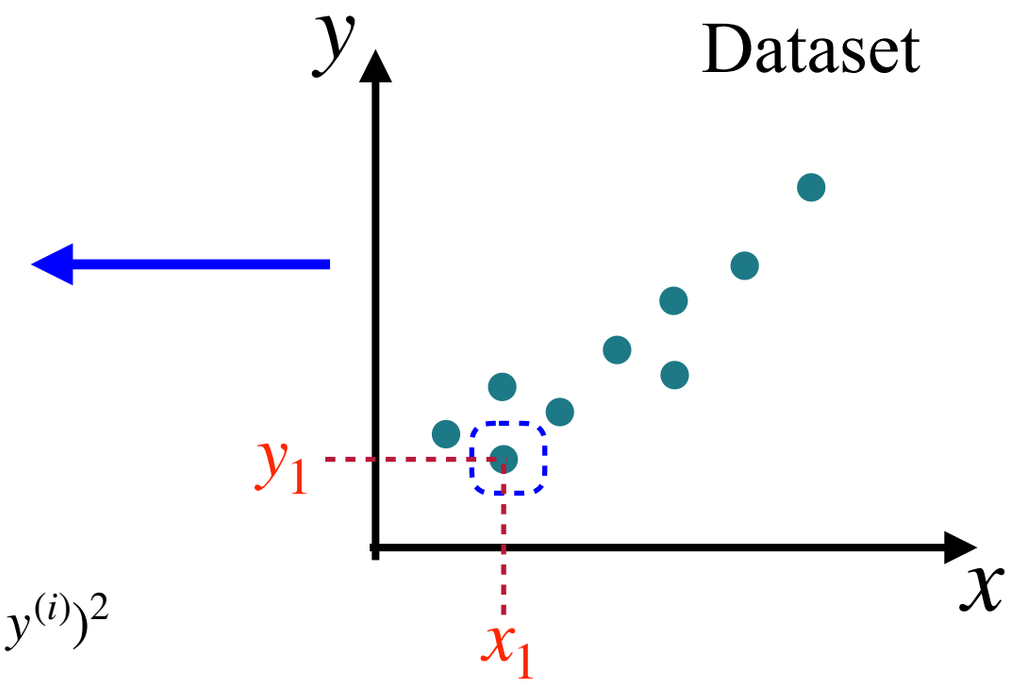
Stochastic Gradient Descent



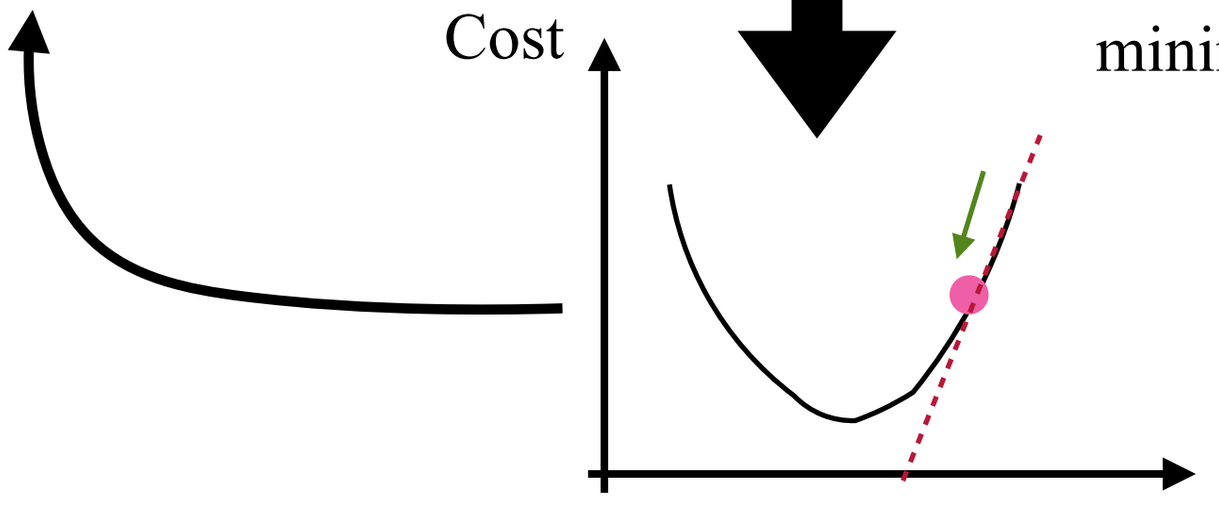
$h_\theta(x) = \theta_0 + \theta_1 x$

$J(\theta_0, \theta_1) = \frac{1}{2m} \Sigma_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

θ_0, θ_1
(Initialized with some values)



Train: tf.train.gradientDescentOptimizer(0.01).
minimize(cost_function)



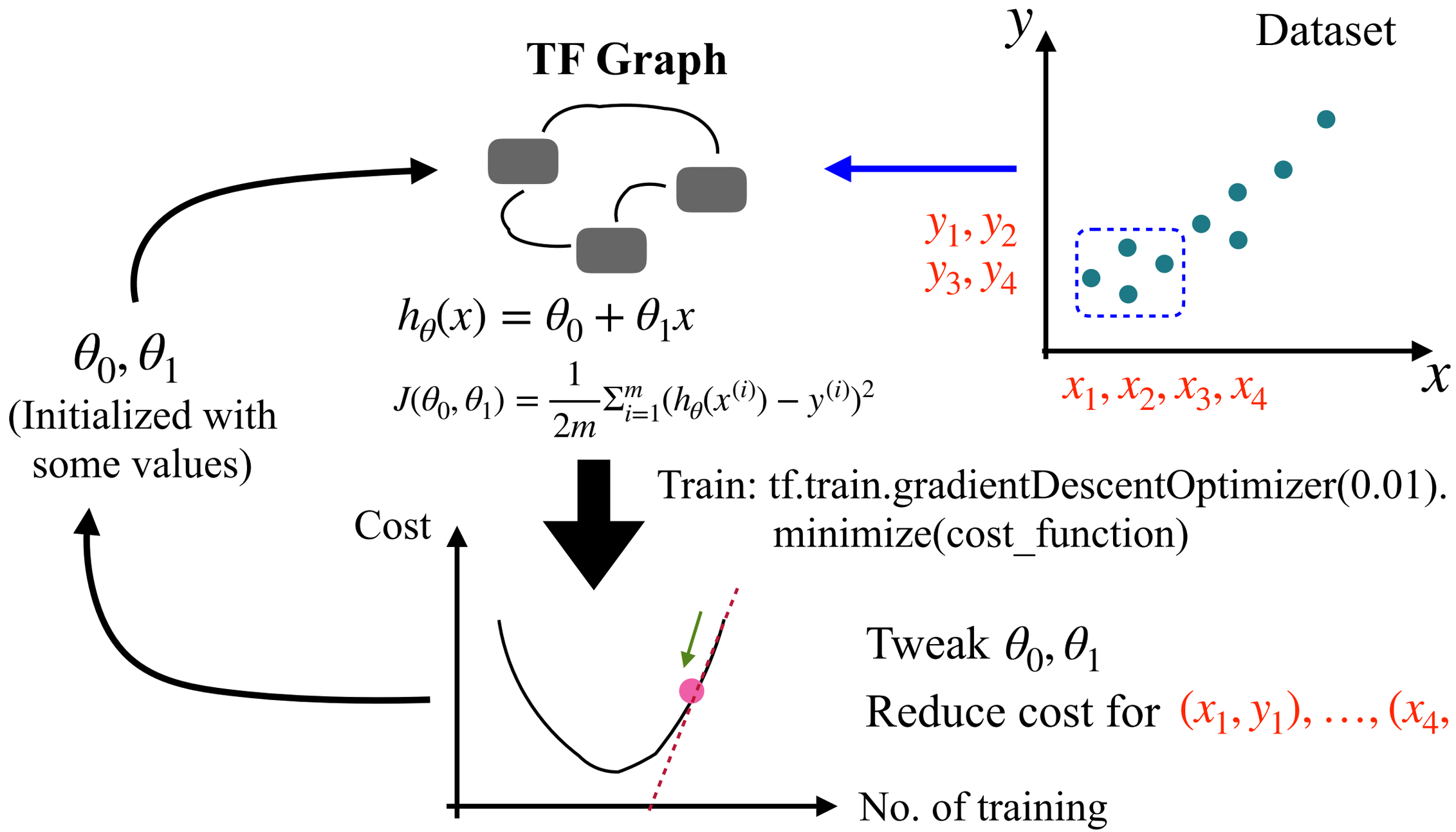
Tweak θ_0, θ_1

Reduce cost for (x_1, y_1)

Stochastic Gradient Descent

- It is called ‘stochastic’ because samples are selected randomly (or shuffled) instead of as a single group (as standard gradient descent)
- As the algorithm sweeps through the training set, it perform gradient update for each training example.
- Data can be shuffled if all data are passed through to prevent cycles.
- A compromise between computing the true gradient and the gradient at a single example is to compute the gradient against more than one training example (called ‘mini-batch’) at each step.
  - It may result in smoother convergence, as the gradient computed at each step is averaged over more training examples.

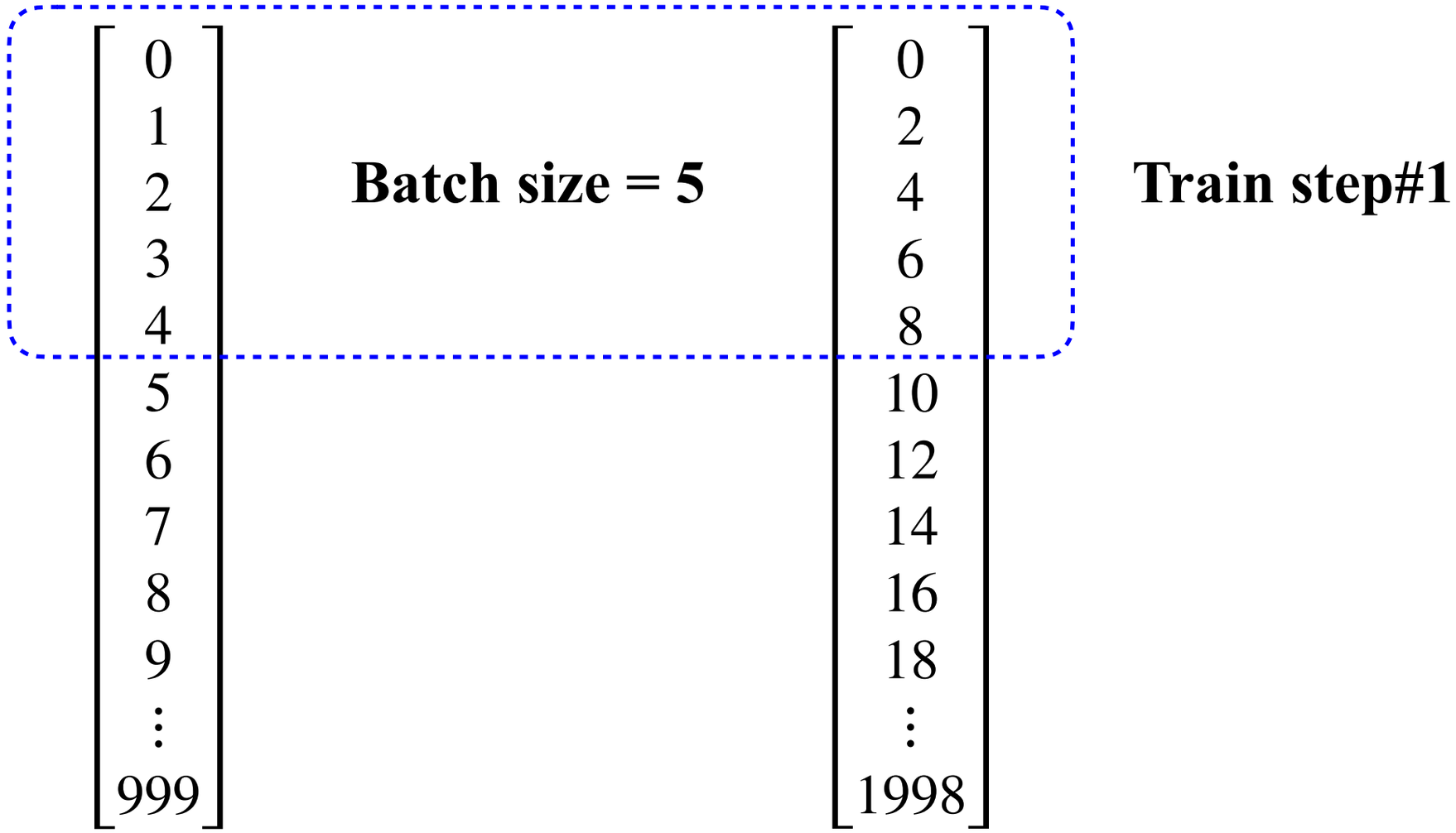
Mini-batch Gradient Descent



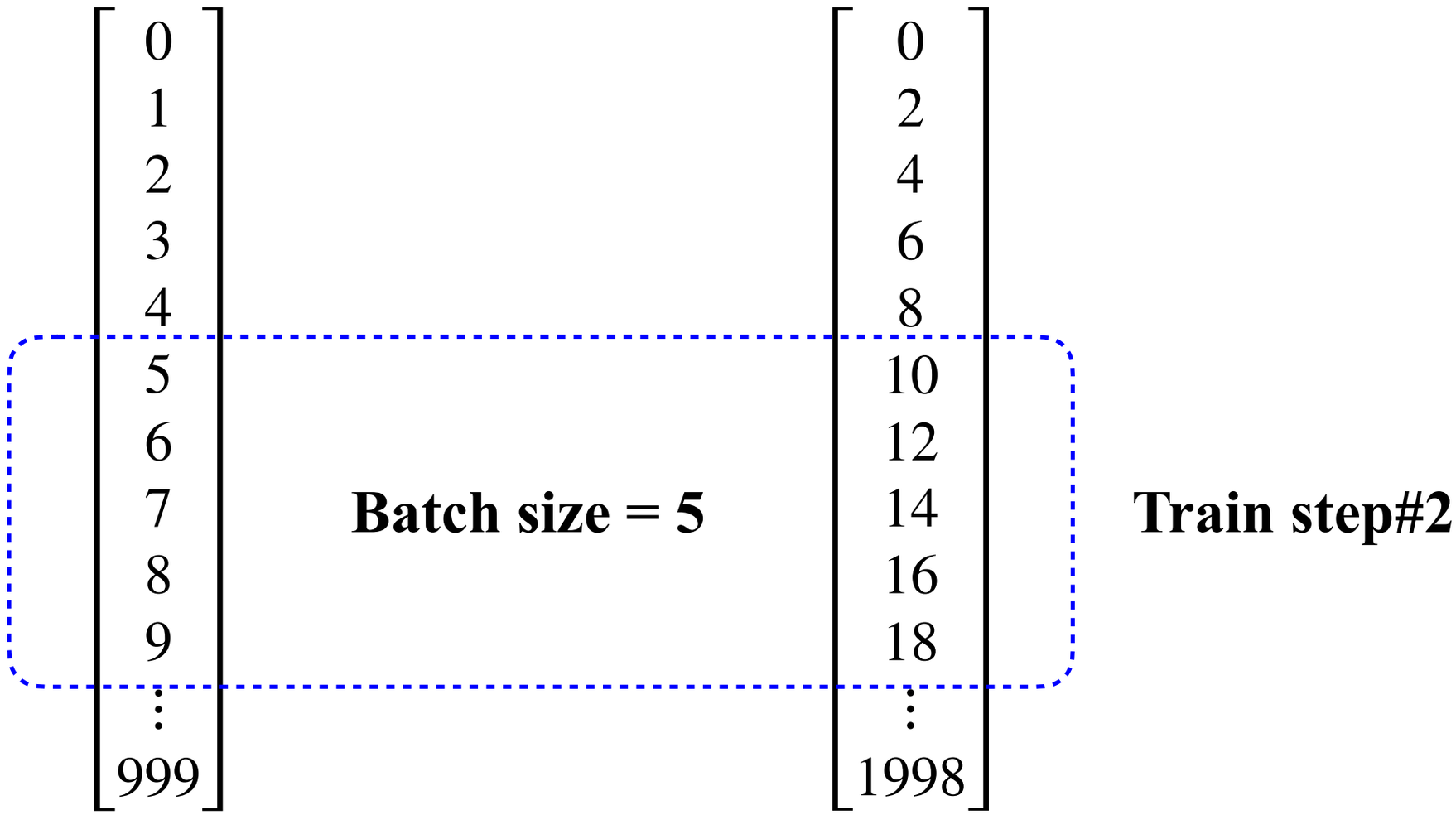
Mini-batch Gradient Descent

$$\begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ \vdots \\ 999 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 2 \\ 4 \\ 6 \\ 8 \\ 10 \\ 12 \\ 14 \\ 16 \\ 18 \\ \vdots \\ 1998 \end{bmatrix}$$

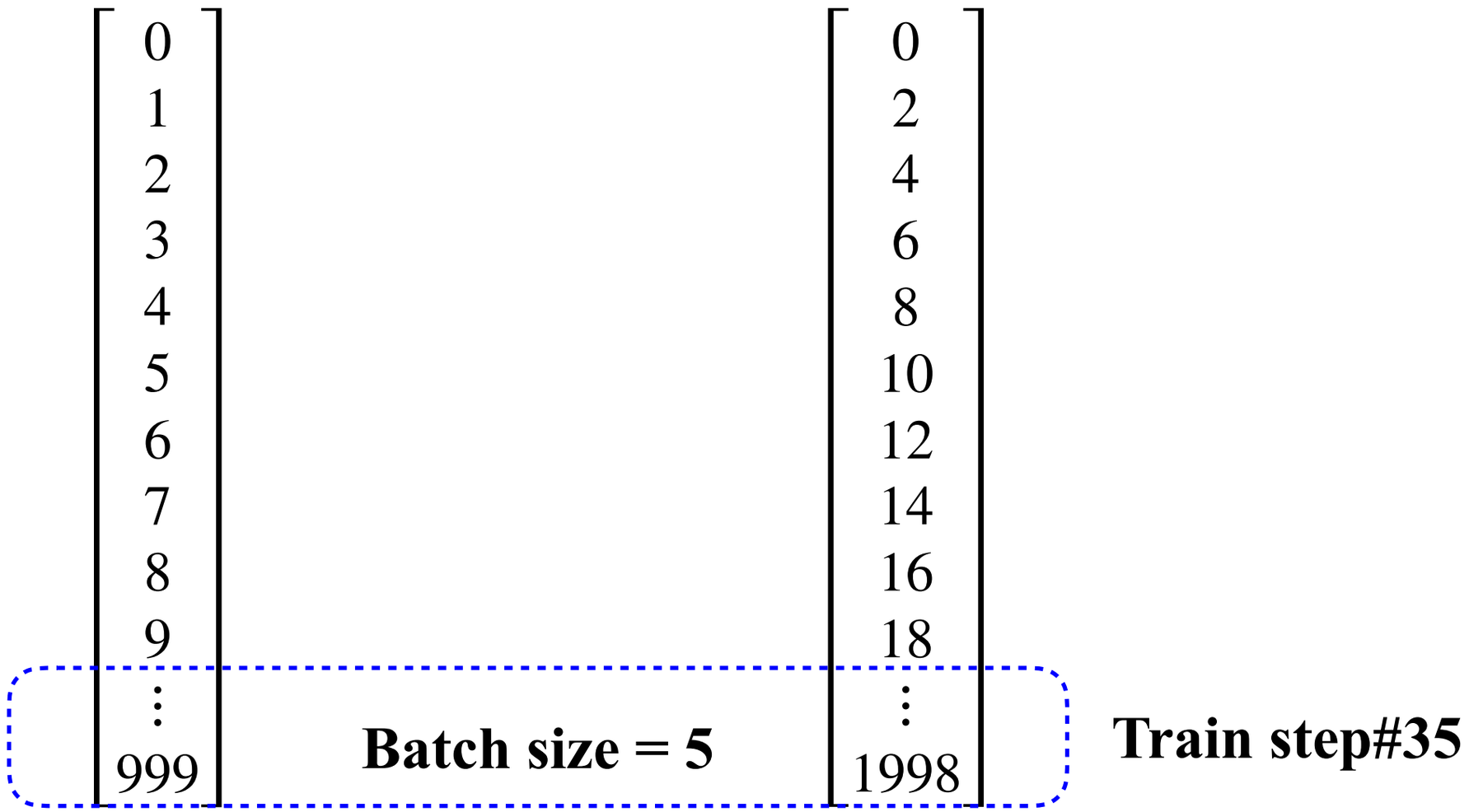
Mini-batch Gradient Descent



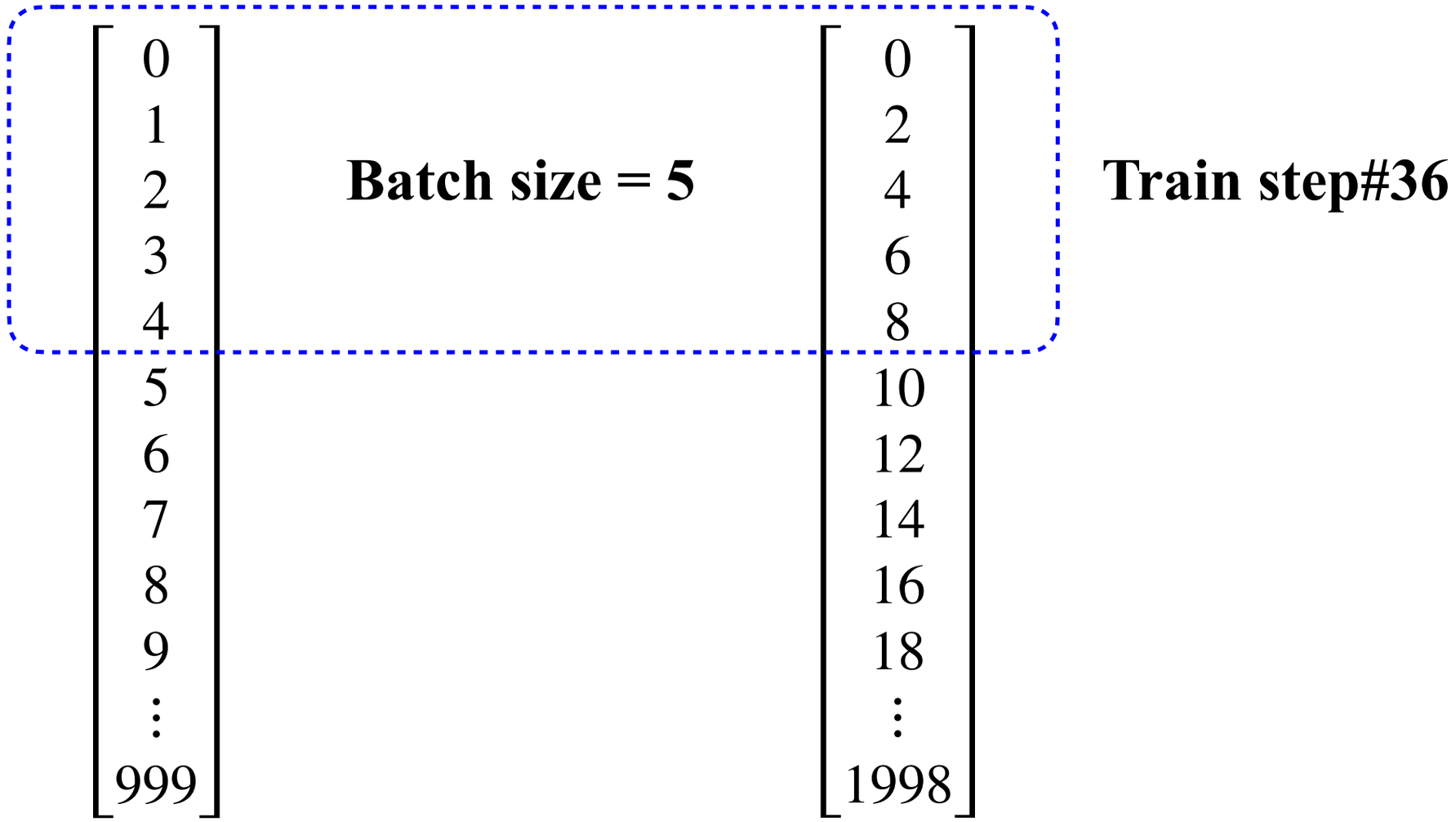
Mini-batch Gradient Descent



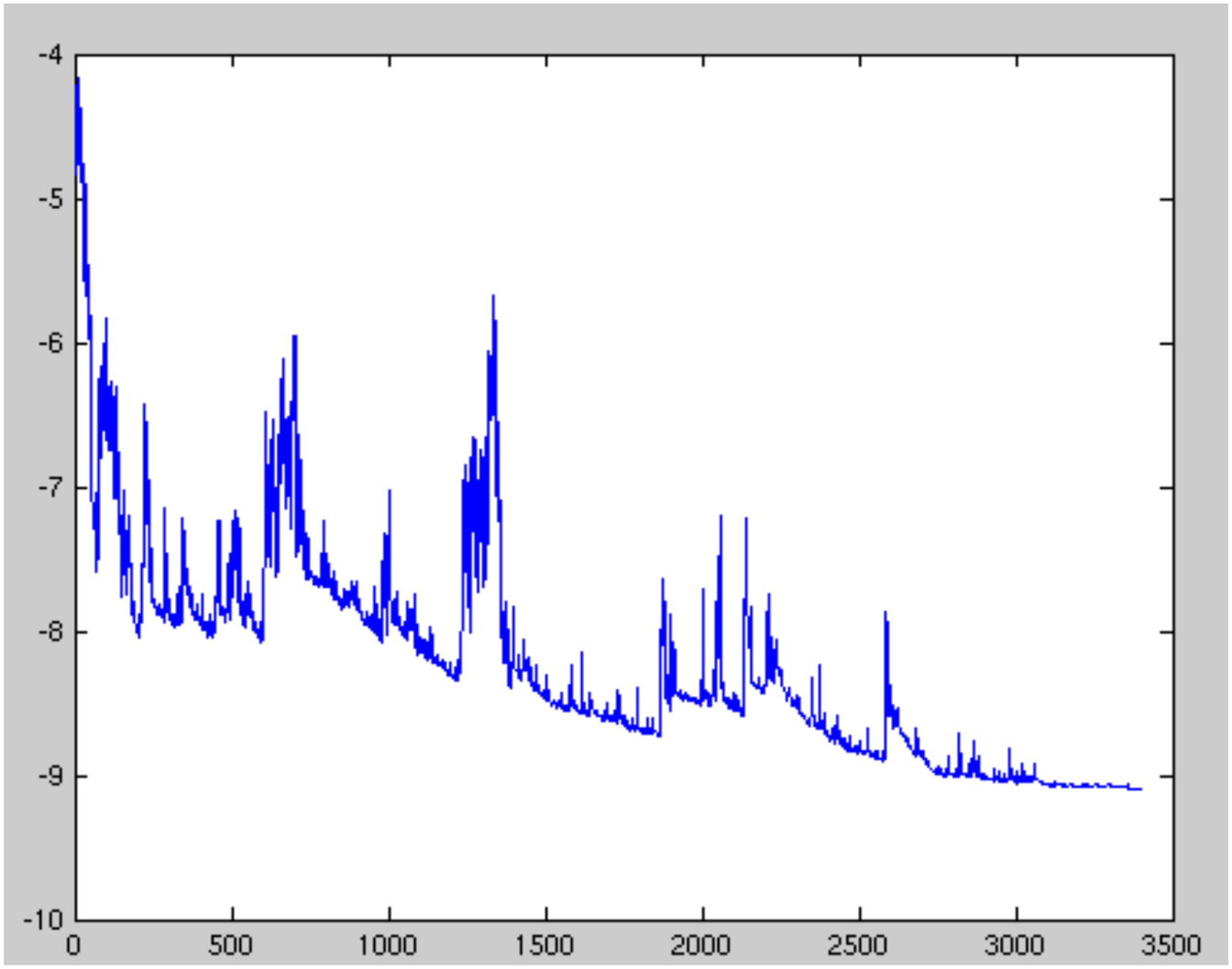
Mini-batch Gradient Descent



Mini-batch Gradient Descent



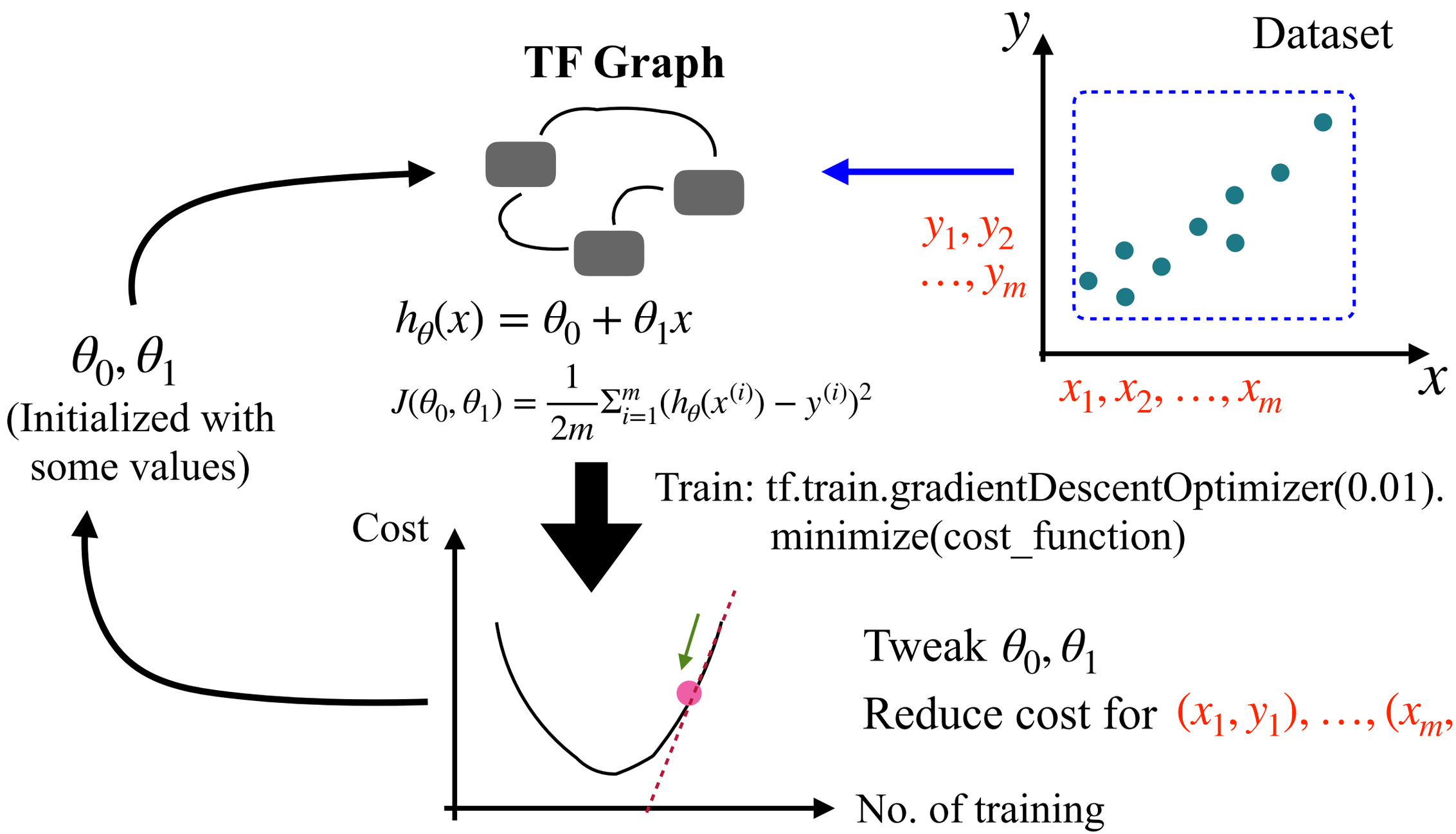
Mini-batch Gradient Descent



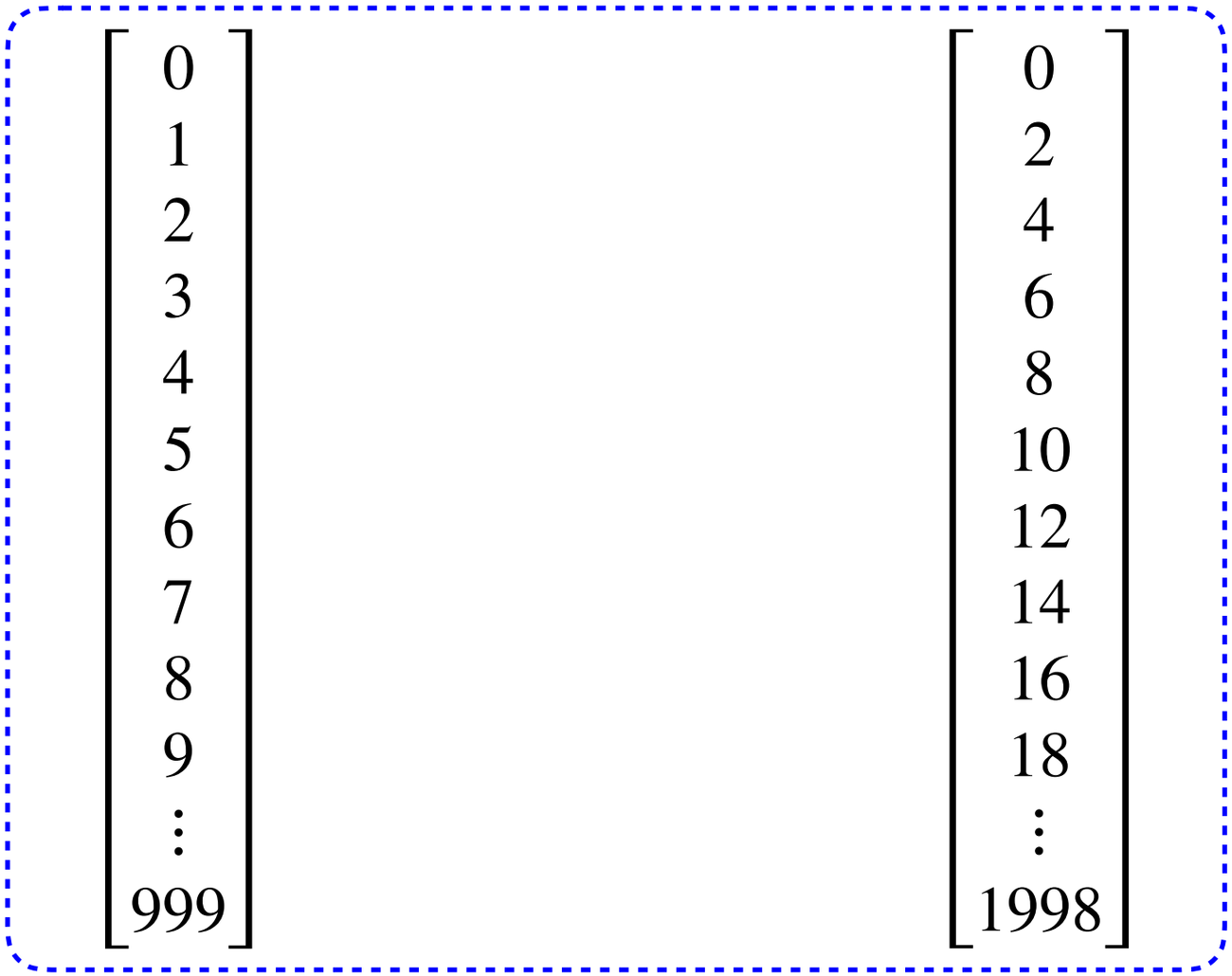
(source: Wikipedia)

Fluctuations in the total objective function as gradient steps with respect to mini-batches are taken.

Batch Gradient Descent



Batch Gradient Descent



Batch Gradient Descent

$$\begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ \vdots \\ 999 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 2 \\ 4 \\ 6 \\ 8 \\ 10 \\ 12 \\ 14 \\ 16 \\ 18 \\ \vdots \\ 1998 \end{bmatrix}$$

Train step#2

Batch Gradient Descent