



Linear regression and Gradient descent algorithm

Fatemeh Seyyedsalehi

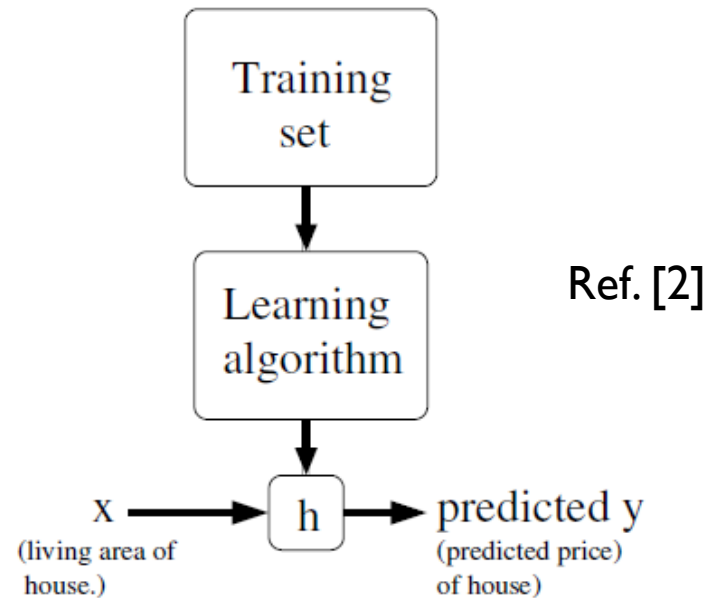
A supervised problem

- ▶ Predict the residential electrical usage as a function of living area and house members.
- ▶ Input feature $\mathbf{x}^{(i)} \in \mathcal{X}$
- ▶ Output or target $y^{(i)} \in \mathcal{Y}$
- ▶ Training set $D = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$

Living area (m^2)	#House members	Residential Electrical usage(kwh)
40	2	4
70	2	5
70	4	7
200	3	8
150	4	7
.	.	.
.	.	.

A supervised problem

- ▶ Our goal is to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$
- ▶ $h(x)$ should be a good predictor for the corresponding y
- ▶ h is called a **hypothesis**



Linear Regression

- ▶ Regression problem: The target is continuous
- ▶ We begin by the class of linear functions as a hypothesis
 - ▶ Easy to extend to generalized linear and so cover more complex regression functions

Linear Regression

- ▶ Coming back to our problem
- ▶ \mathbf{x} is a two dimensional vector in \mathbb{R}^2
- ▶ We first decide how to represent the hypothesis h
 - ▶ A function family

Living area (m^2)	#House members	Residential Electrical usage(kwh)
40	2	4
70	2	5
70	4	7
200	3	8
150	4	7
.	.	.
.	.	.

Linear Regression: the hypothesis space

- ▶ $h_w(\mathbf{x})$ as a linear function of \mathbf{x}

$$h_w(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_2$$

- ▶ Where w_i s are called parameters or weights of the model which parameterize the space of linear functions mapping from \mathcal{X} to \mathcal{Y} .

Linear Regression: the hypothesis space

- ▶ To have a better notation we consider $x_0 = 1$ (the intercept term),

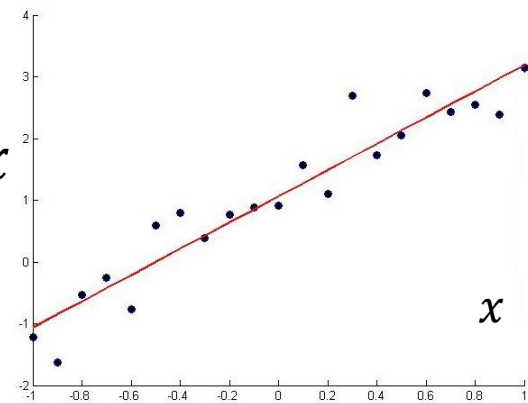
$$h_w(\mathbf{x}) = \sum_{i=0}^d w_i x_i$$

- ▶ How to learn parameters w_i s, given a training set,
 - ▶ Make $h(\mathbf{x})$ close to y

Linear Regression: the hypothesis space

► Univariate

$$h: \mathbb{R} \rightarrow \mathbb{R} \quad h_w(\mathbf{x}) = w_0 + w_1 x$$



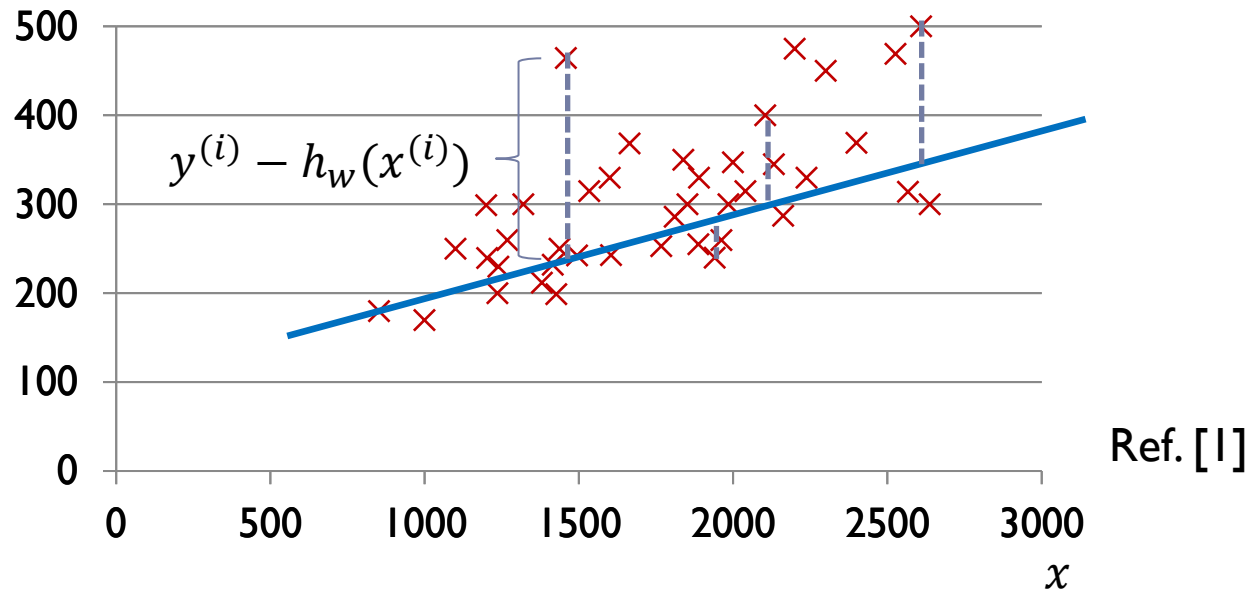
► Multivariate

$$h: \mathbb{R}^d \rightarrow \mathbb{R} \quad h_w(\mathbf{x}) = w_0 + w_1 x_1 + \dots w_d x_d$$

$\mathbf{w} = [w_0, w_1, \dots, w_d]^T$ are parameters we need to set.

Ref. [1]

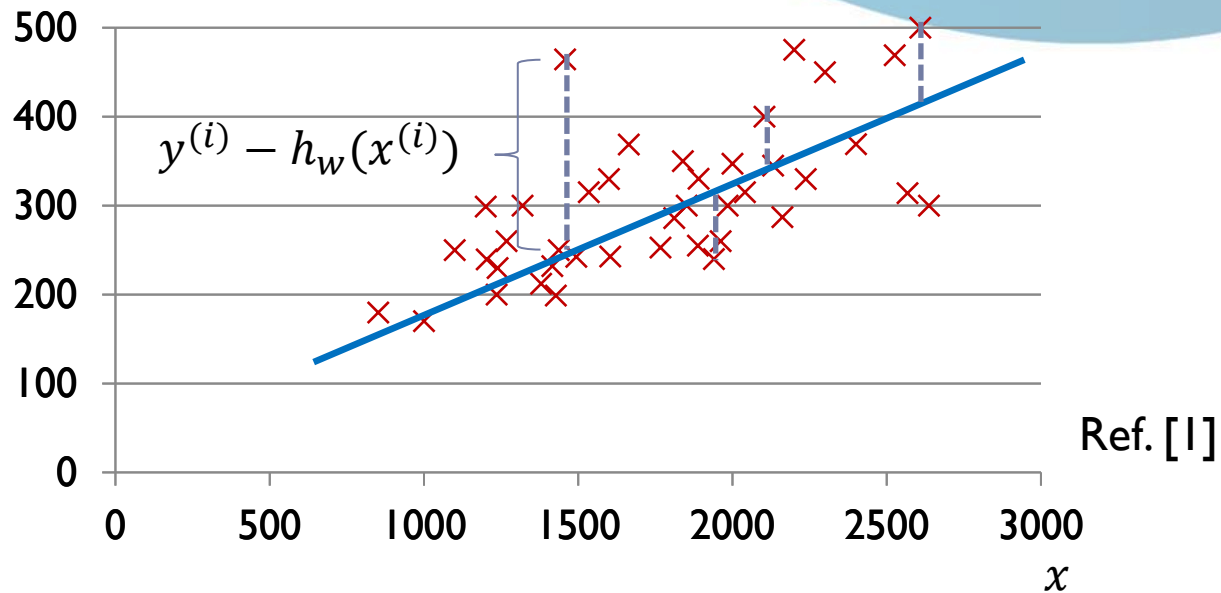
How to measure the error



- ▶ A loss function between the ground truth and the estimated output

$$\text{Loss} = (y - h_w(x))^2$$

How to measure the error



Cost function (sum of squared error(SSE)):

$$\begin{aligned} J(\mathbf{w}) &= \sum_{i=1}^n (y^{(i)} - h_w(x^{(i)}))^2 \\ &= \sum_{i=1}^n (y^{(i)} - w_0 - w_1 x^{(i)})^2 \end{aligned}$$

Linear Regression: the learning algorithm

- ▶ Choose \mathbf{w} so as to minimize the $J(\mathbf{w})$

$$J(\mathbf{w}) = \sum_{i=1}^n (y^{(i)} - h_{\mathbf{w}}(\mathbf{x}^{(i)}))^2$$

- ▶ The learning algorithm: optimization of the cost function
 - ▶ Explicitly taking the cost function derivative with respect to the w_i s, and setting them to zero.
- ▶ Parameters of the best hypothesis for the training set:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} J(\mathbf{w})$$

Cost function optimization: univariate

$$J(\mathbf{w}) = \sum_{i=1}^n (y^{(i)} - w_0 - w_1 x^{(i)})^2$$

- Necessary conditions for the “optimal” parameter values:

$$\frac{\partial J(\mathbf{w})}{\partial w_0} = 0$$

$$\frac{\partial J(\mathbf{w})}{\partial w_1} = 0$$

Optimality conditions: univariate

$$J(\mathbf{w}) = \sum_{i=1}^n (y^{(i)} - w_0 - w_1 x^{(i)})^2$$

$$\frac{\partial J(\mathbf{w})}{\partial w_1} = \sum_{i=1}^n 2(y^{(i)} - w_0 - w_1 x^{(i)})(-x^{(i)}) = 0$$

$$\frac{\partial J(\mathbf{w})}{\partial w_0} = \sum_{i=1}^n 2(y^{(i)} - w_0 - w_1 x^{(i)})(-1) = 0$$

- A systems of 2 linear equations

Cost function optimization: multivariate

$$J(\mathbf{w}) = \sum_{i=1}^n (y^{(i)} - h_{\mathbf{w}}(\mathbf{x}^{(i)}))^2 = \sum_{i=1}^n (\textcolor{blue}{y}^{(i)} - \mathbf{w}^T \textcolor{red}{x}^{(i)})^2$$

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_d^{(1)} \\ 1 & x_1^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$$

Cost function optimization: multivariate

- Explicitly taking the cost function derivative with respect to the \mathbf{w} s, and setting them to zero.

$$J(\mathbf{w}) = \|\mathbf{y} - X\mathbf{w}\|_2^2$$

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = -2X^T(\mathbf{y} - X\mathbf{w})$$

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = \mathbf{0} \Rightarrow X^T X \mathbf{w} = X^T \mathbf{y}$$

$$\mathbf{w} = (X^T X)^{-1} X^T \mathbf{y}$$

- Is $(X^T X)$ invertible?

Cost function optimization

- ▶ Another approach,
 - ▶ Start from an initial guess and iteratively change \mathbf{w} to minimize $J(\mathbf{w})$.
 - ▶ The gradient descent algorithm
- ▶ Steps:
 - ▶ Start from \mathbf{w}^0
 - ▶ Repeat
 - ▶ Update \mathbf{w}^t to \mathbf{w}^{t+1} in order to reduce J
 - ▶ $t \leftarrow t + 1$
 - ▶ until we hopefully end up at a minimum



Review:

Gradient descent

- ▶ In each step, takes steps proportional to the negative of the gradient vector of the function at the current point \mathbf{w}^t :

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla J(\mathbf{w}^t)$$

- ▶ $J(\mathbf{w})$ decreases fastest if one goes from \mathbf{w}^t in the direction of $-\nabla J(\mathbf{w}^t)$
- ▶ Assumption: $J(\mathbf{w})$ is defined and differentiable in a neighborhood of a point \mathbf{w}^t

Gradient ascent takes steps proportional to (the positive of) the gradient to find a local maximum of the function

- ▶ Continue to find

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} J(\mathbf{w})$$

Review:

Gradient descent

- ▶ Minimize $J(\mathbf{w})$

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla_{\mathbf{w}} J(\mathbf{w}^t)$$

Step size
(Learning rate parameter)

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = \begin{bmatrix} \frac{\partial J(\mathbf{w})}{\partial w_1} \\ \vdots \\ \frac{\partial J(\mathbf{w})}{\partial w_d} \end{bmatrix}$$

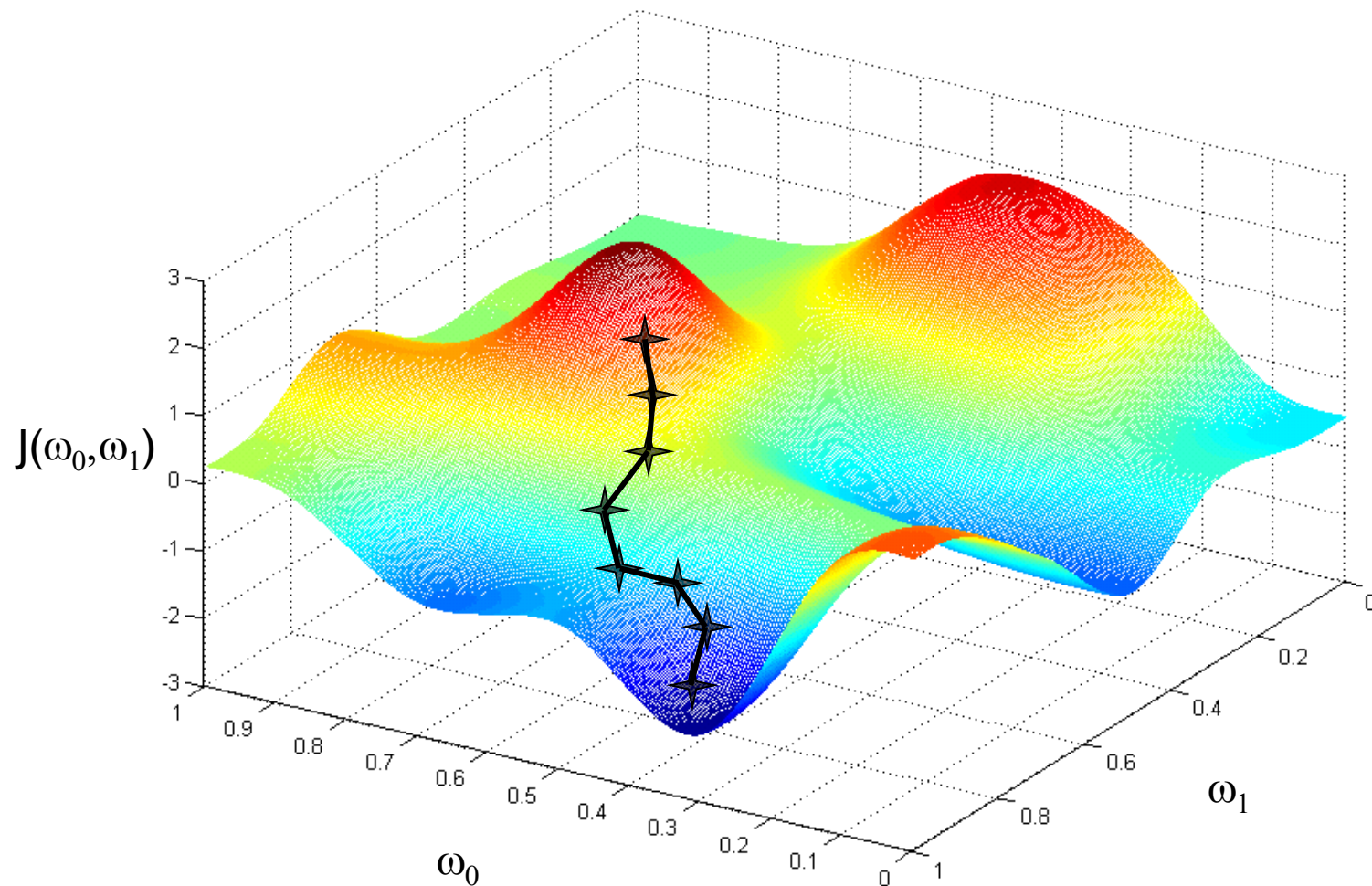
- ▶ If η is small enough, then $J(\mathbf{w}^{t+1}) \leq J(\mathbf{w}^t)$.
- ▶ η can be allowed to change at every iteration as η_t .

Review:

Gradient descent disadvantages

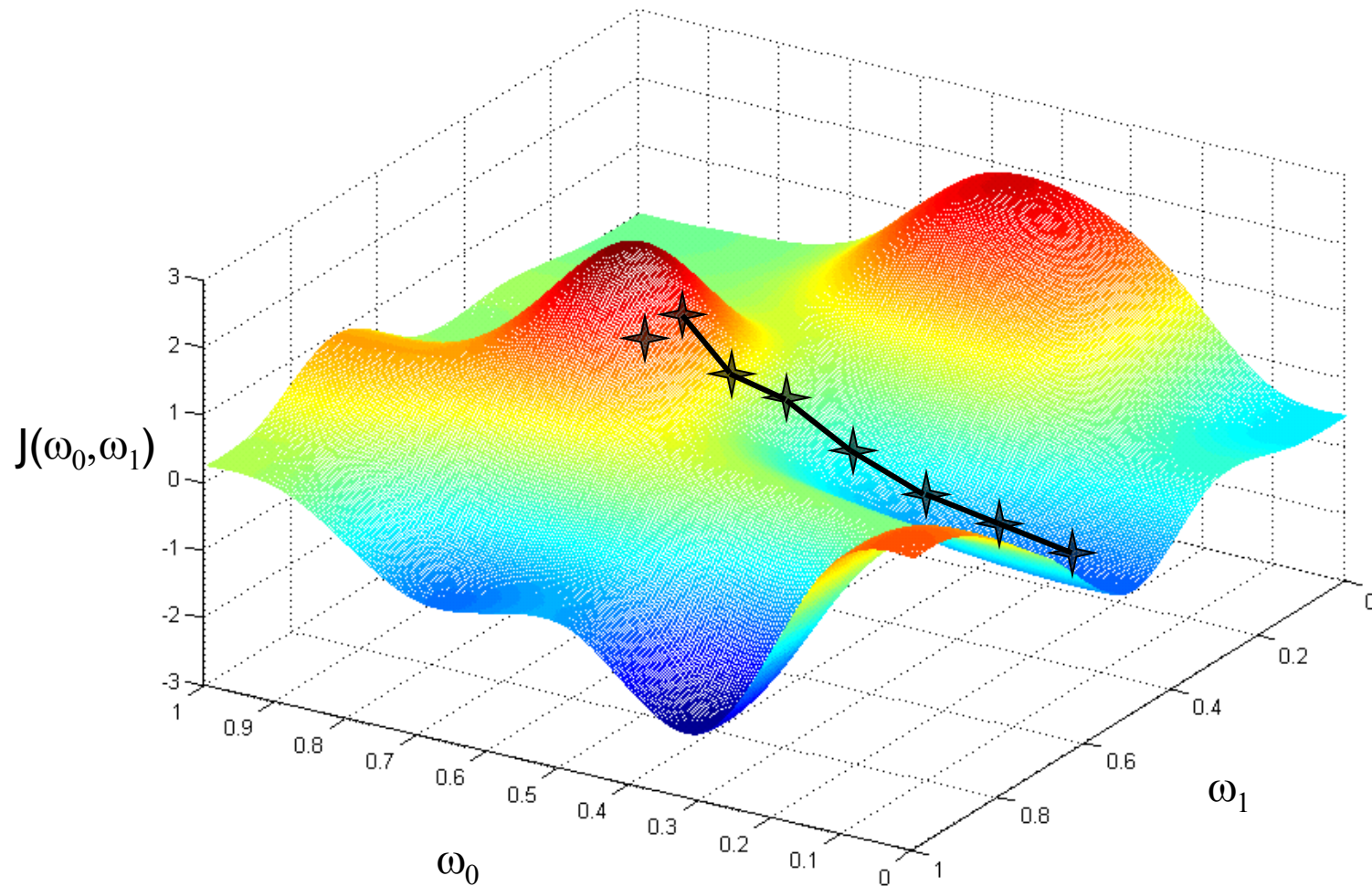
- ▶ Local minima problem
- ▶ However, when J is convex, all local minima are also global minima \Rightarrow gradient descent can converge to the global solution.

Review: Problem of gradient descent with non-convex cost functions



Ref. [2]

Review: Problem of gradient descent with non-convex cost functions



Ref. [2]

Cost function optimization

- ▶ Minimize $J(\mathbf{w})$

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla_{\mathbf{w}} J(\mathbf{w}^t)$$

- ▶ $J(\mathbf{w})$: Sum of squares error

$$J(\mathbf{w}) = \sum_{i=1}^n \left(y^{(i)} - h_{\mathbf{w}}(\mathbf{x}^{(i)}) \right)^2$$

- ▶ Weight update rule for $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$:

$$\mathbf{w}^{t+1} = \mathbf{w}^t + \eta \sum_{i=1}^n \left(y^{(i)} - \mathbf{w}^{tT} \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}$$

Cost function optimization

- ▶ Weight update rule: $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$

$$\mathbf{w}^{t+1} = \mathbf{w}^t + \eta \sum_{i=1}^n (y^{(i)} - \mathbf{w}^T \mathbf{x}^{(i)}) \mathbf{x}^{(i)}$$

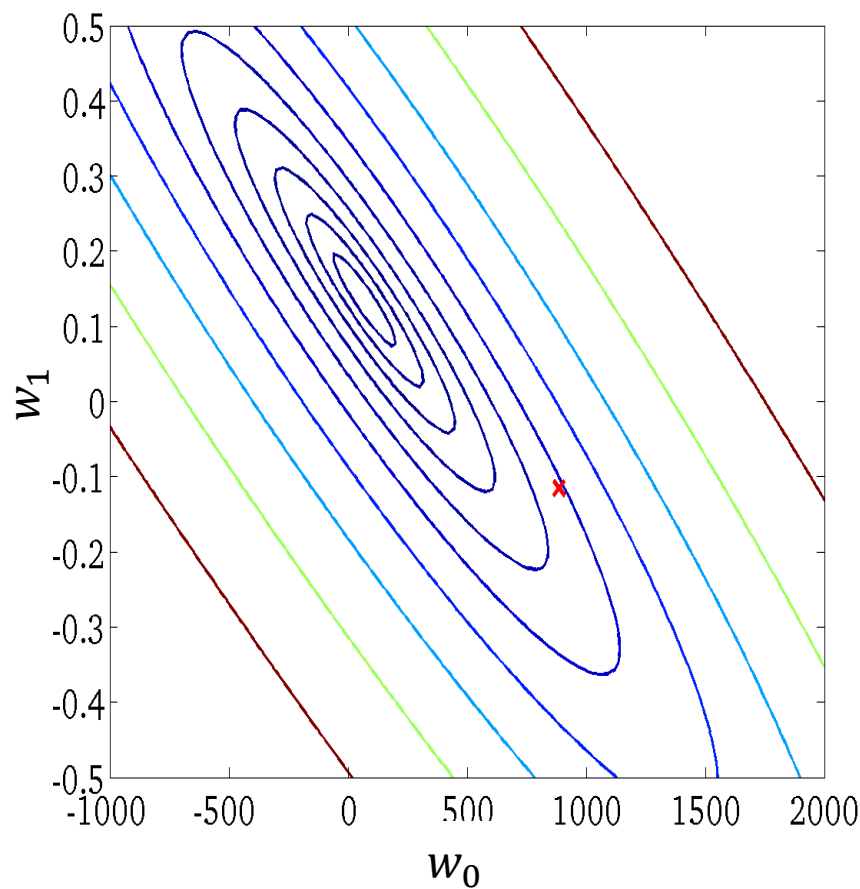
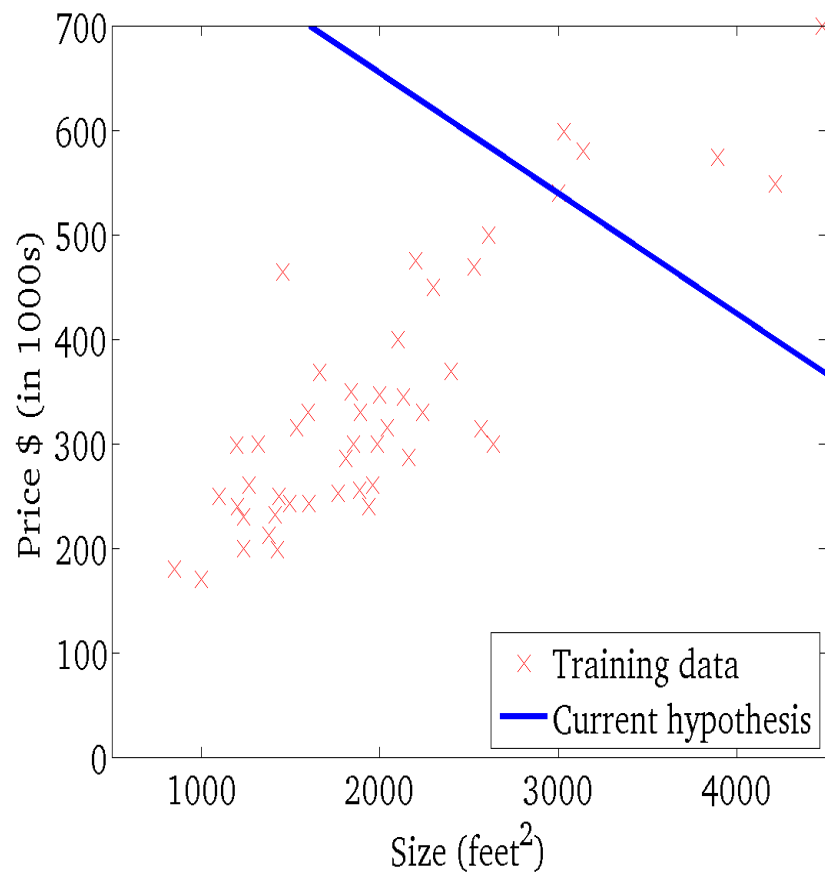
Batch mode: each step
considers all training data

- ▶ η : too small \rightarrow gradient descent can be slow.
- ▶ η : too large \rightarrow gradient descent can overshoot the minimum. It may fail to converge, or even diverge.

$$J(w_0, w_1)$$

(function of the parameters w_0, w_1)

$$h_w(x) = w_0 + w_1 x$$

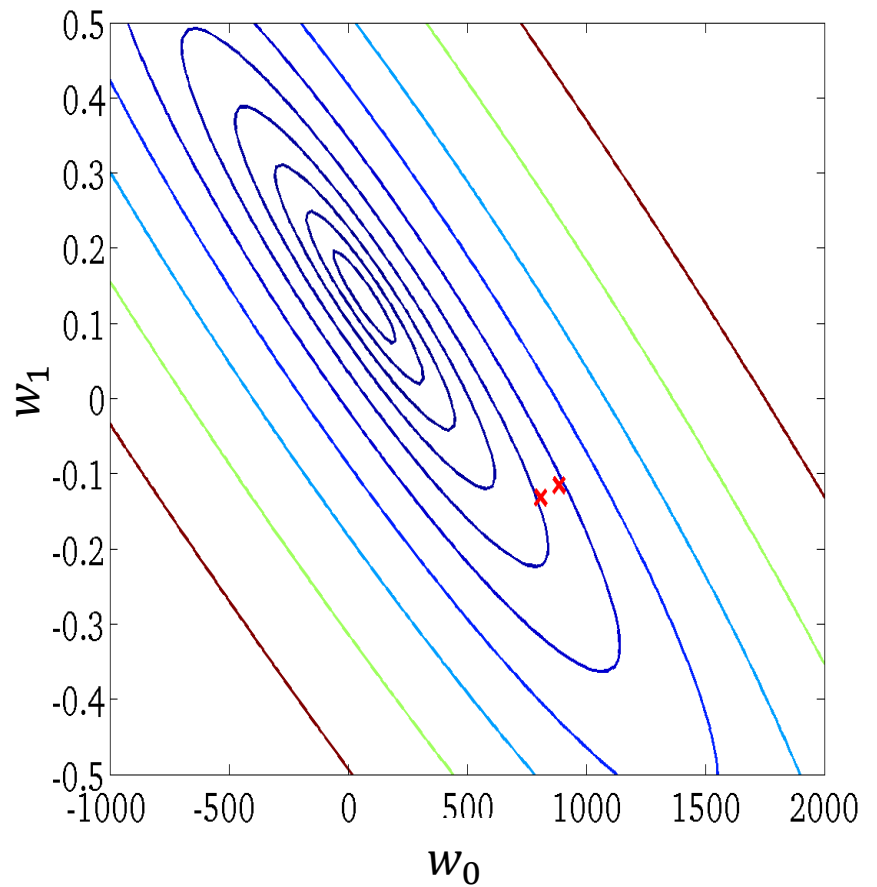
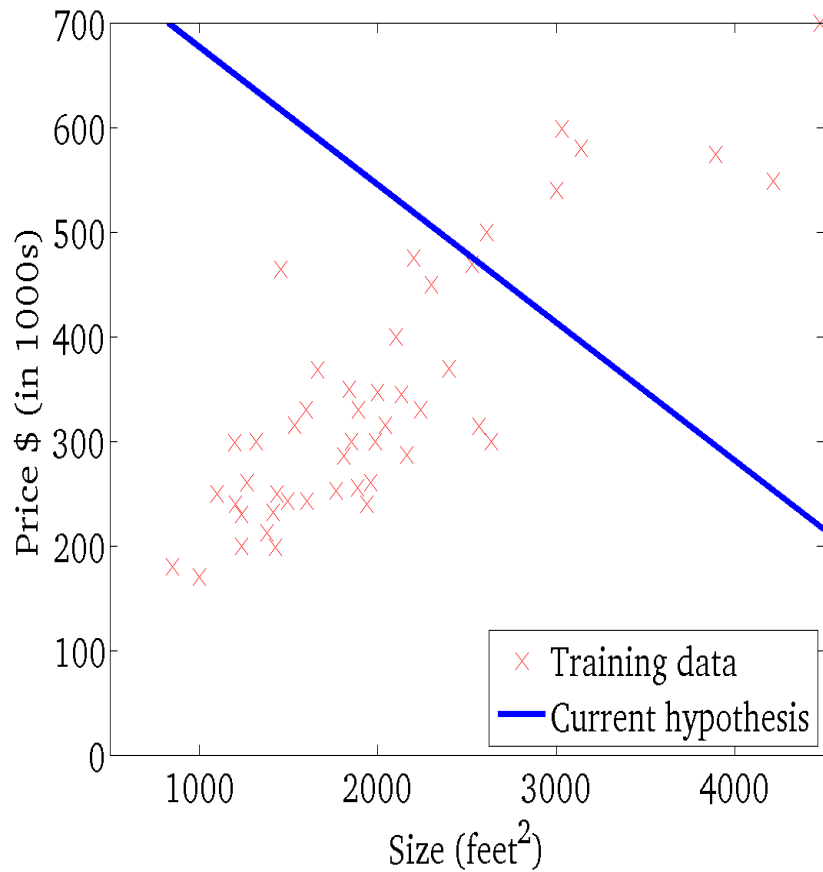


Ref. [2]

$$J(w_0, w_1)$$

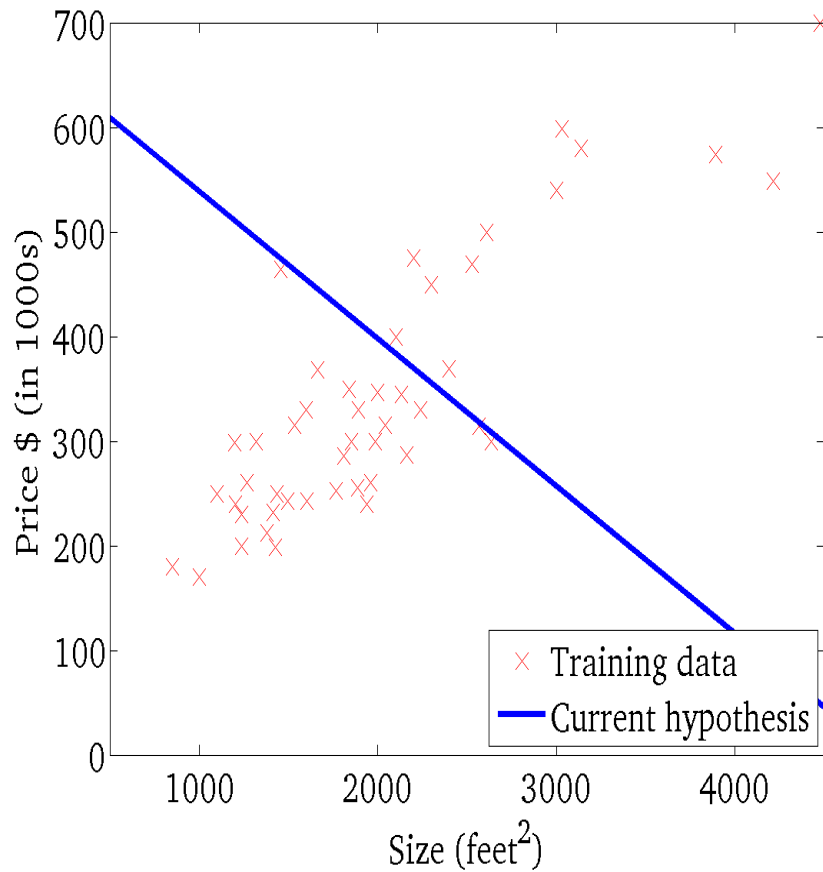
(function of the parameters w_0, w_1)

$$h_w(x) = w_0 + w_1 x$$

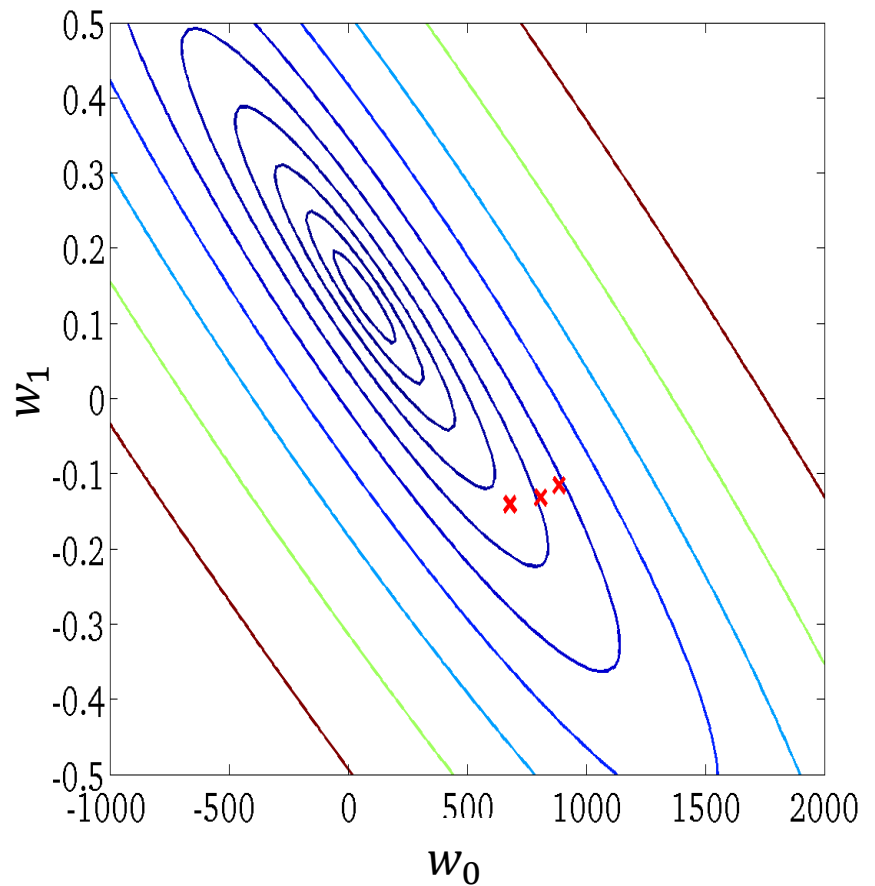


Ref. [2]

$$h_w(x) = w_0 + w_1 x$$



$J(w_0, w_1)$
(function of the parameters w_0, w_1)

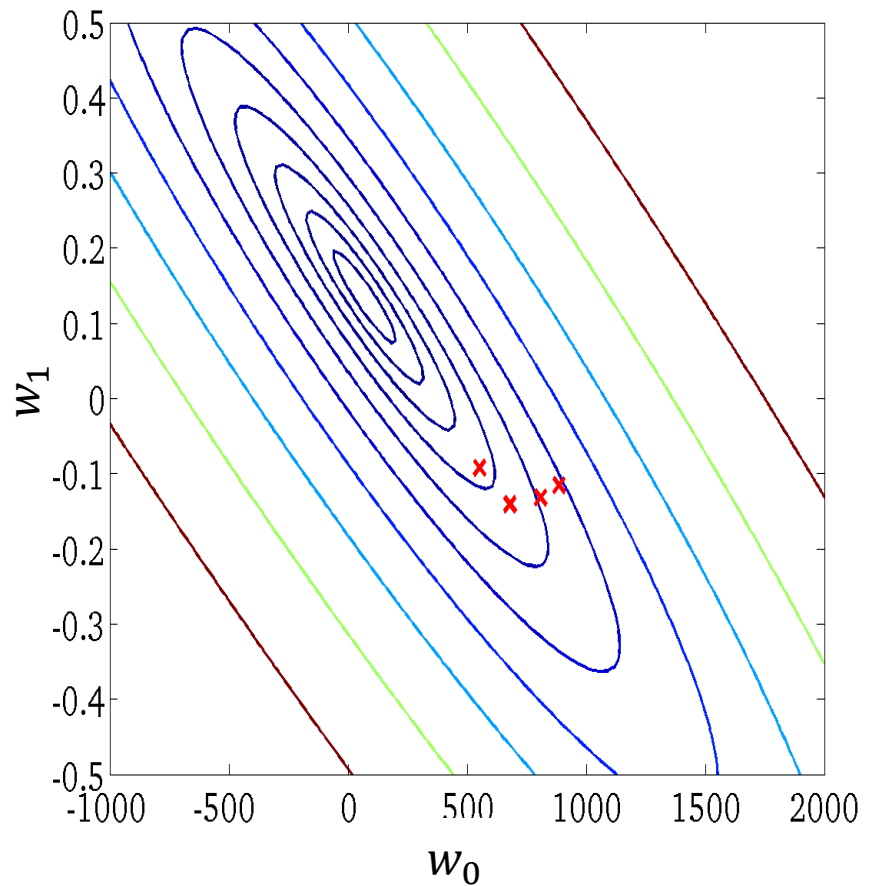
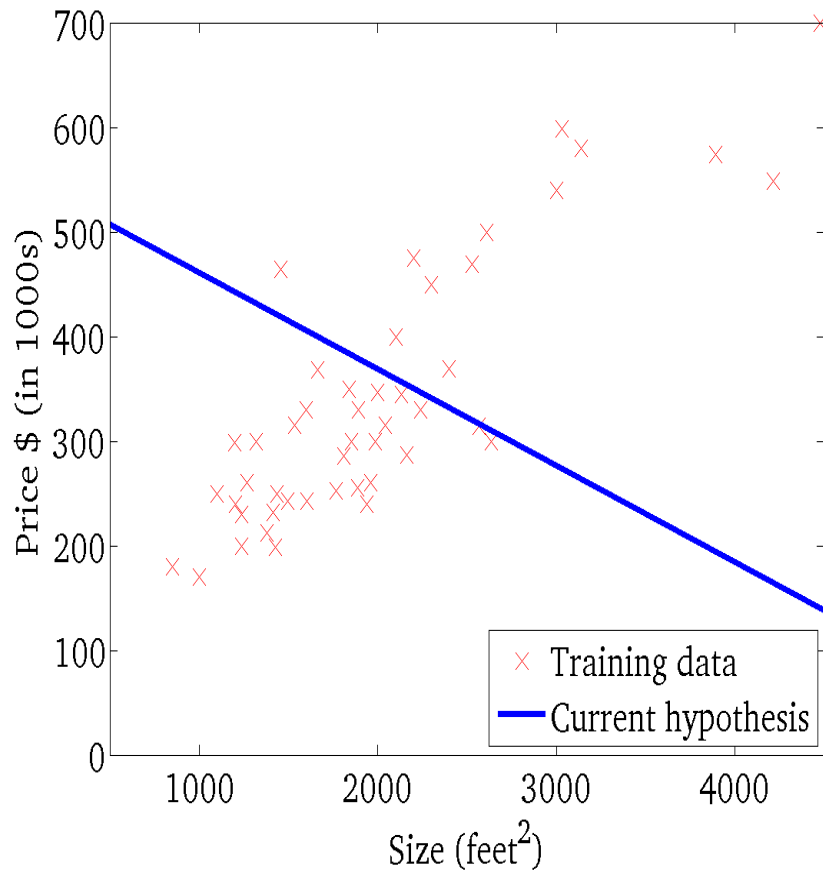


Ref. [2]

$$J(w_0, w_1)$$

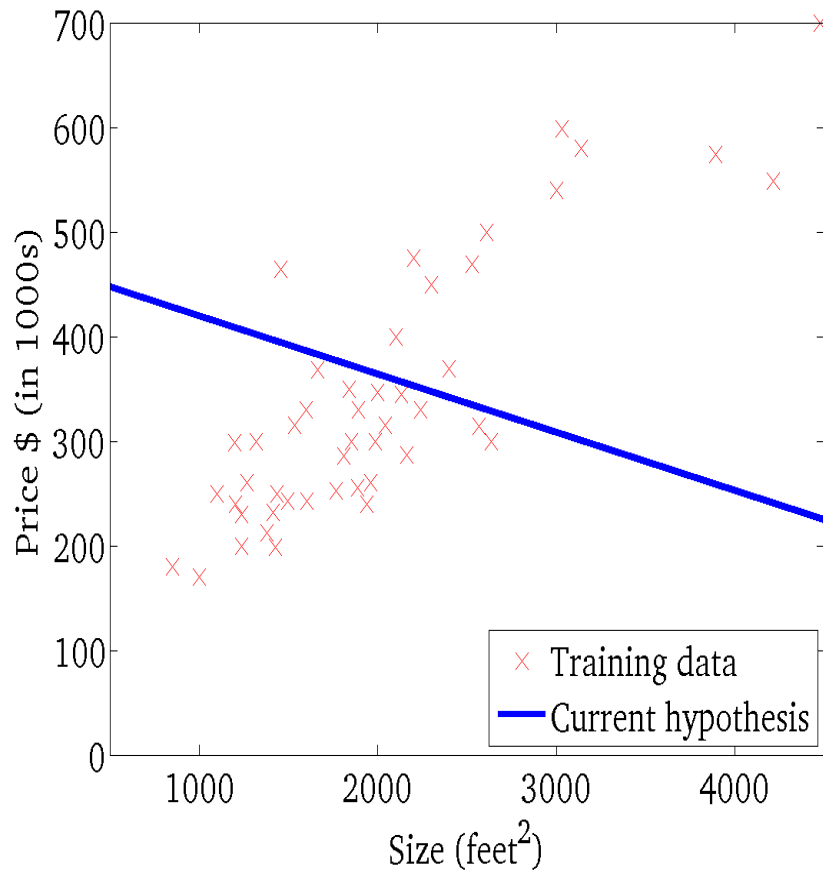
(function of the parameters w_0, w_1)

$$h_w(x) = w_0 + w_1 x$$

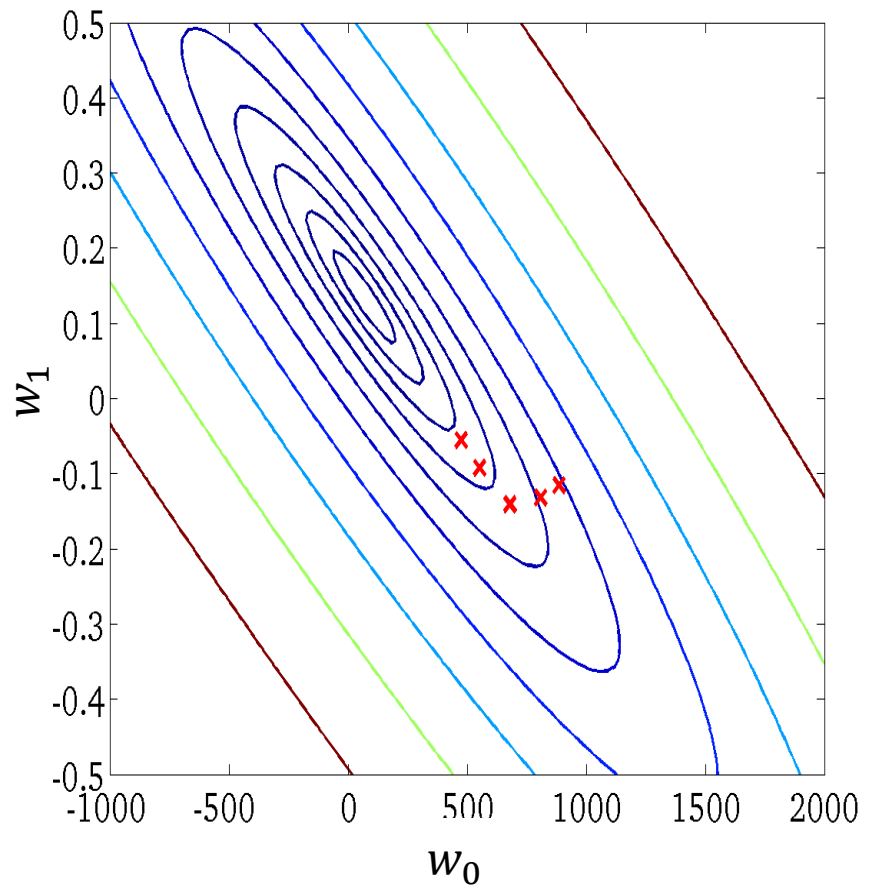


Ref. [2]

$$h_w(x) = w_0 + w_1 x$$



$J(w_0, w_1)$
(function of the parameters w_0, w_1)

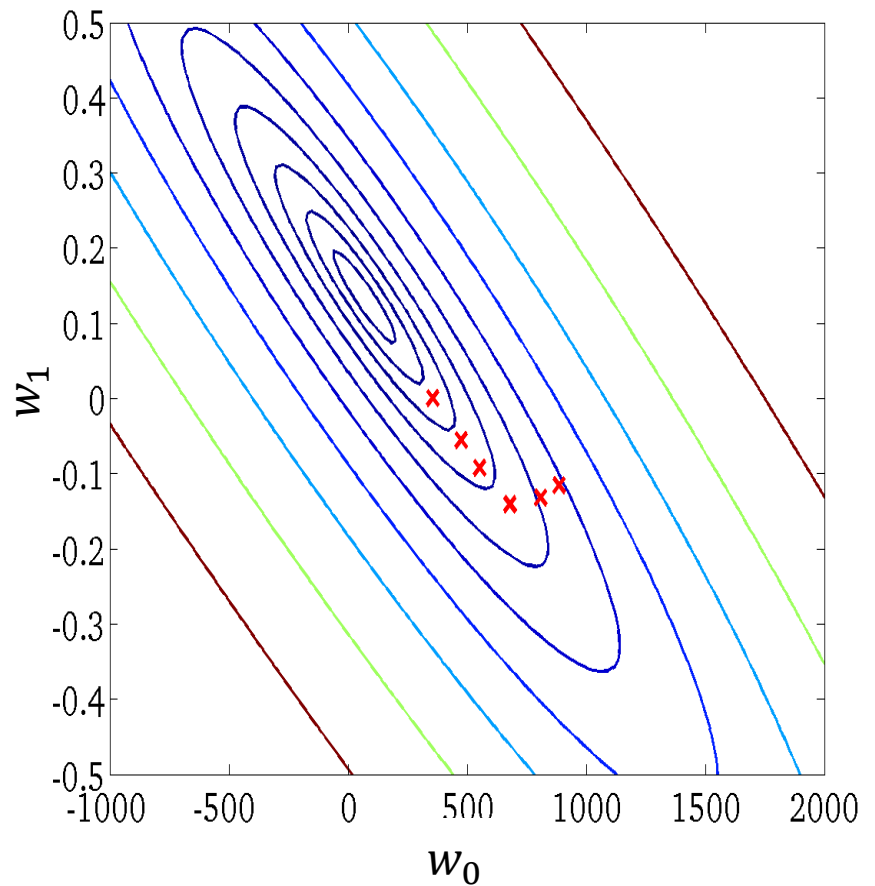
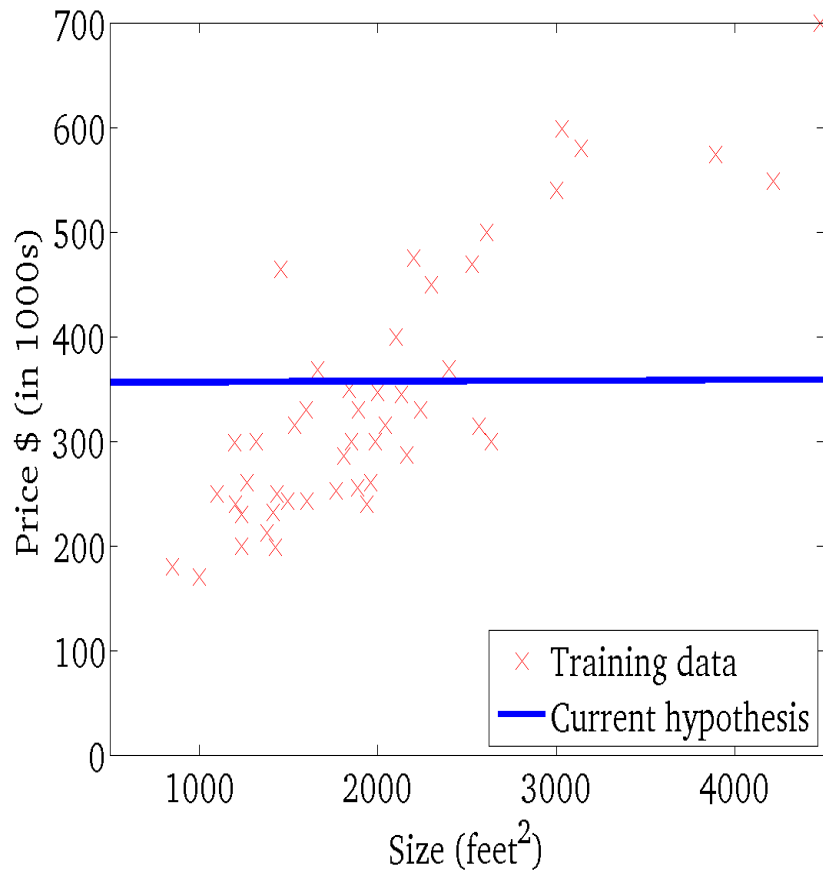


Ref. [2]

$$h_w(x) = w_0 + w_1 x$$

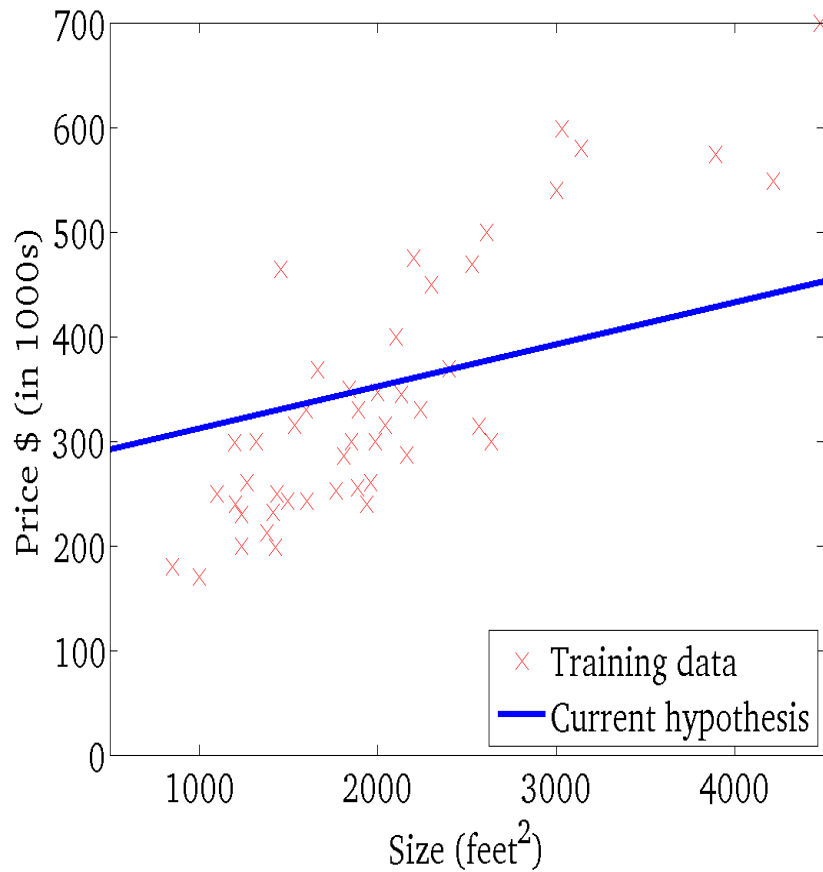
$$J(w_0, w_1)$$

(function of the parameters w_0, w_1)

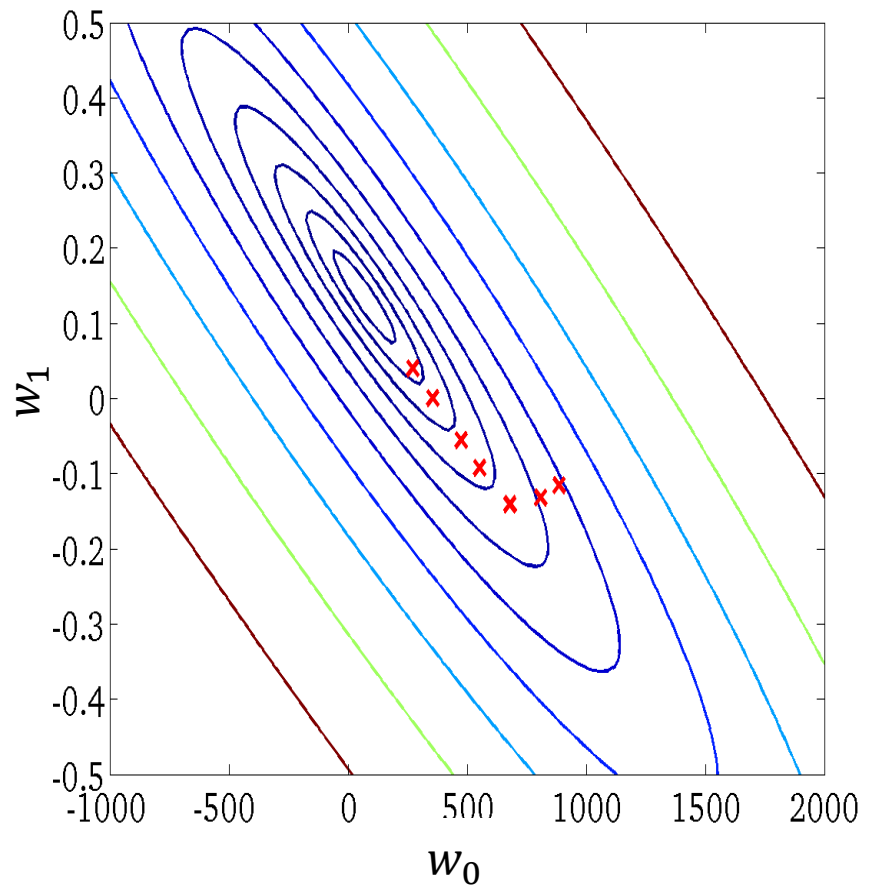


Ref. [2]

$$h_w(x) = w_0 + w_1 x$$



$J(w_0, w_1)$
(function of the parameters w_0, w_1)

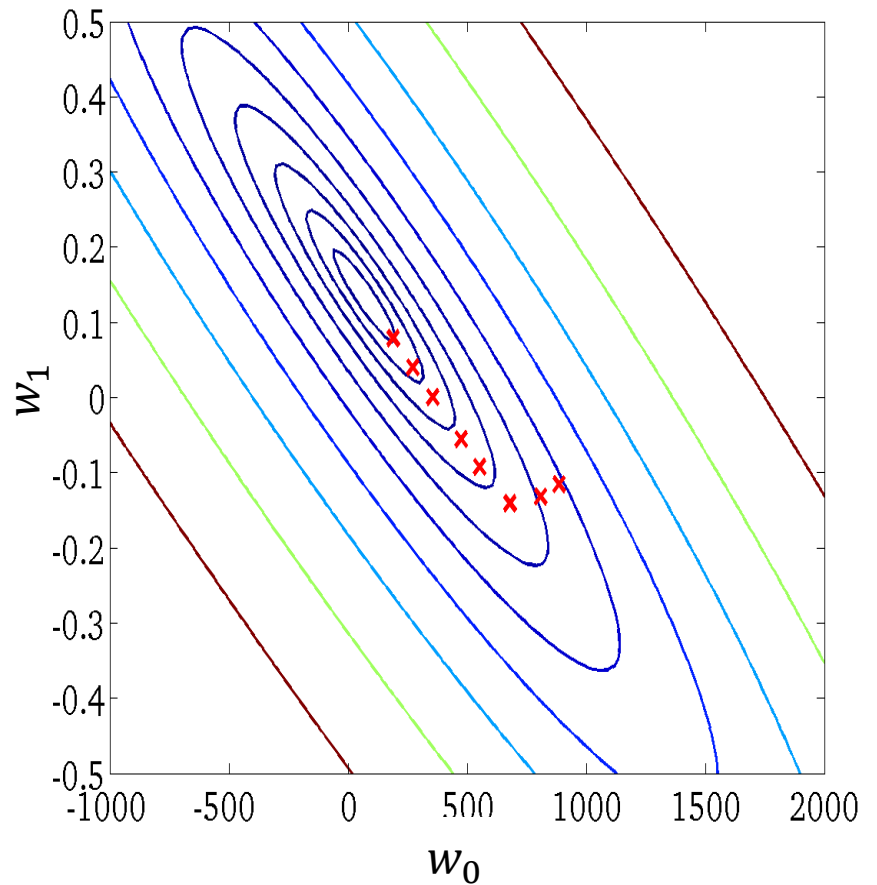
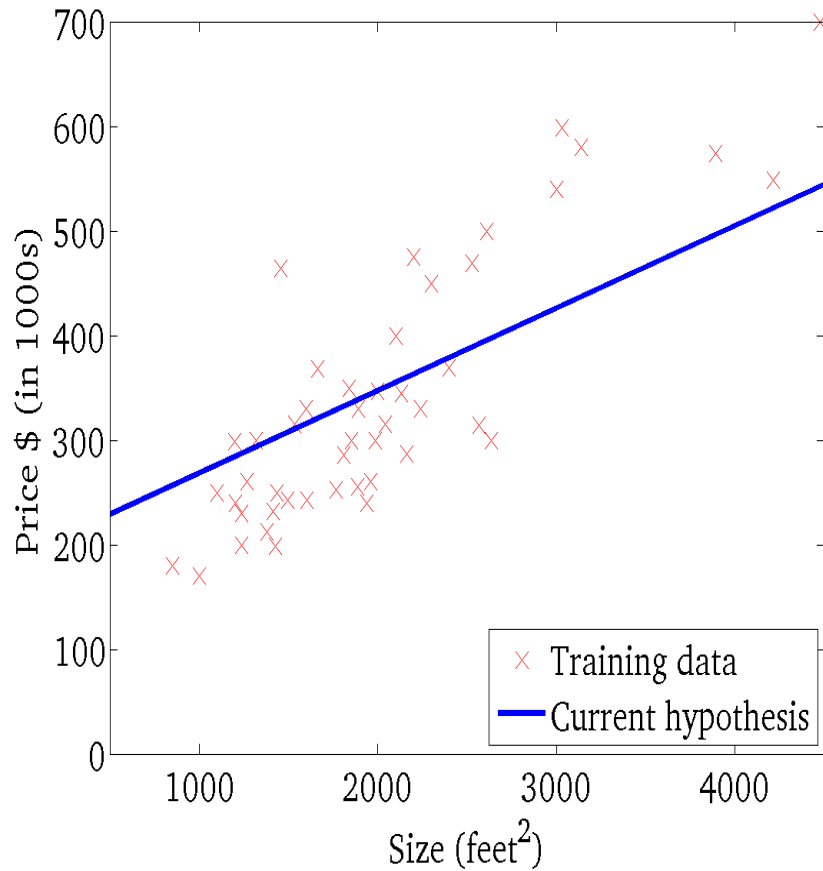


Ref. [2]

$$h_w(x) = w_0 + w_1 x$$

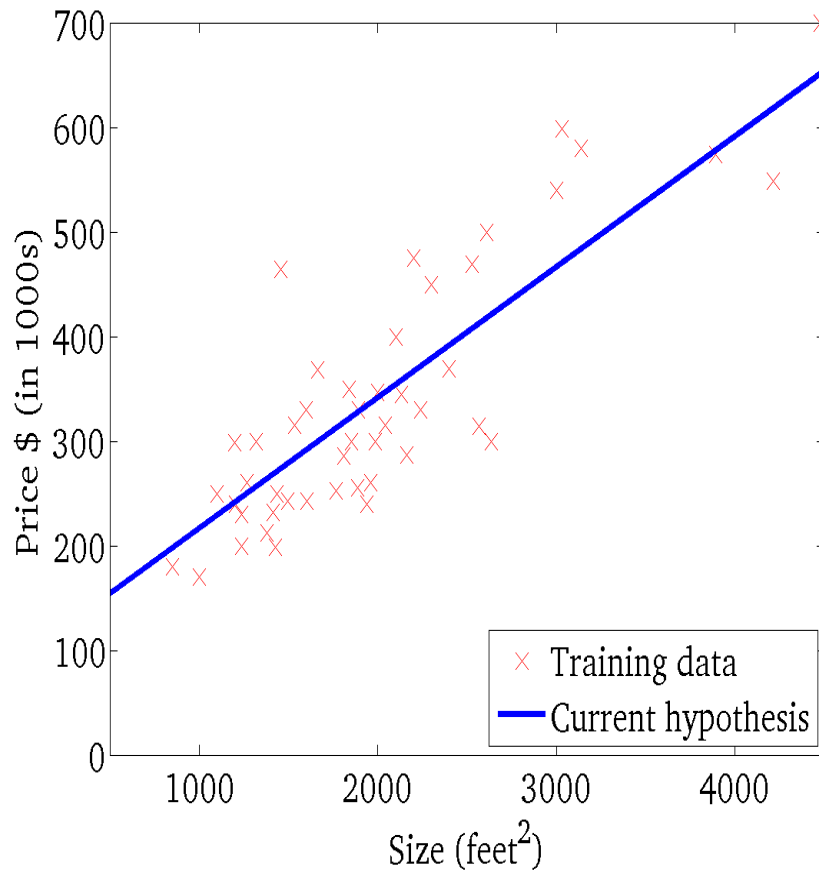
$$J(w_0, w_1)$$

(function of the parameters w_0, w_1)



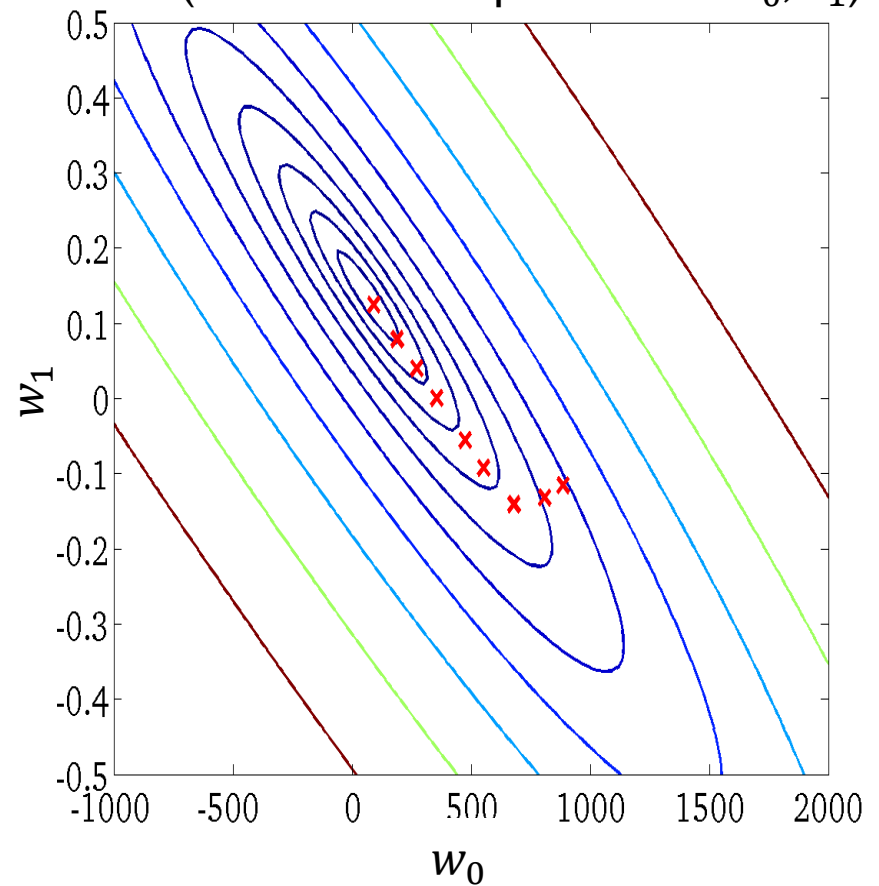
Ref. [2]

$$h_w(x) = w_0 + w_1 x$$



$$J(w_0, w_1)$$

(function of the parameters w_0, w_1)



Ref. [2]