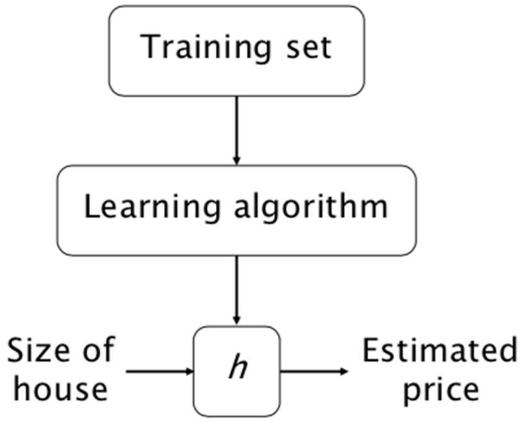
02 - MODEL AND COST FUNCTION

Model representation

Size in feet^2 (x)	Price ($) in 1000's (y)	
2104	460	
$x^{(2)}$ $\boxed{1416}$	$y^{(2)}$ $\boxed{232}$	2-nd training example
1534	315	
852	178	
...	...	

Notazione

- x: variabili di input (indipendente) → features
- y: variabile di output → variabile di target
- (x, y): un training example
- $(x^{(i)}, y^{(i)})$: l'i-esimo **training example** (riga i)
- m: numero di training examples (numero di righe) → il numero di $(x^{(i)}, y^{(i)})$



$h : x \to y$

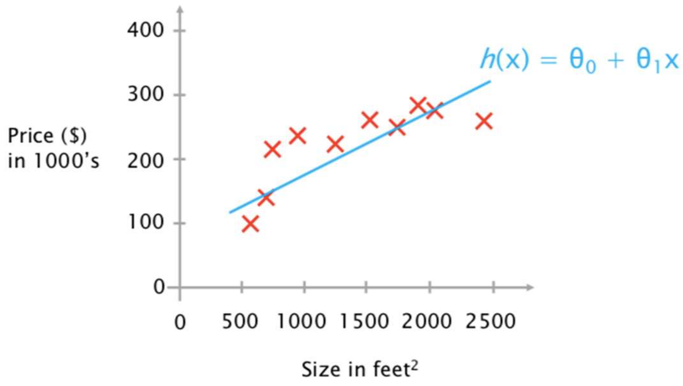
Quindi l'algoritmo, dato un training set, andrà a trovare una funzione chiamata **ipotesi** "h" *(la curva di fitting)* che mappi un input x ad un output y nel modo migliore possibile.
Dati gli esempi x io vorrei avere come output dei valori molto simili a quelle che sono le etichette delle x.
Di conseguenza vorrei che l'algoritmo si comporti bene anche con valori nuovi.

Se si parla di **regressione lineare univariata**, la h è una funzione lineare: $h_\theta(x) = \theta_0 + \theta_1 x$

dove θ_0 : intercetta b, θ_1 : pendenza w

(Univariate) Linear regression

- regressione: y continuo
- lineare: uso una retta
- univariata: abbiamo una sola variabile indipendente (x)



Cost function (linear regression)

Il nostro obiettivo è trovare la retta che fitti nel modo migliore possibile i dati. Per farlo mi serve una funzione di costo.
Quindi avremo:

- un'ipotesi: $h_\theta(x) = \theta_0 + \theta_1 x$
- θ_i: i parametri

Variando i parametri cambia la retta.
Devo minimizzare la seguente **funzione di costo** (square cost function):

$$\underset{\theta_0\,\theta_1}{\text{minimize}}\ \frac{1}{m}\sum_{i=1}^{m}\overbrace{\frac{1}{2}\big(\underbrace{h_\theta(x^{(i)})}_{\text{Predicted value}} - \underbrace{y^{(i)}}_{\text{Actual value}}\big)^2}^{\text{Loss}(h_\theta(x),y)\ \text{or Cost}(h_\theta(x),y)}$$

Faccio la differenza tra il valore predetto e il valore vero. Prendo il quadrato di queste differenze dividendolo per due e poi lo faccio per tutti i training examples.

Consideriamo un attimo una **versione semplificata** del problema togliendo il parametro θ_0 :

$$\underset{\theta_0\,\theta_1}{\text{minimize}}\ \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

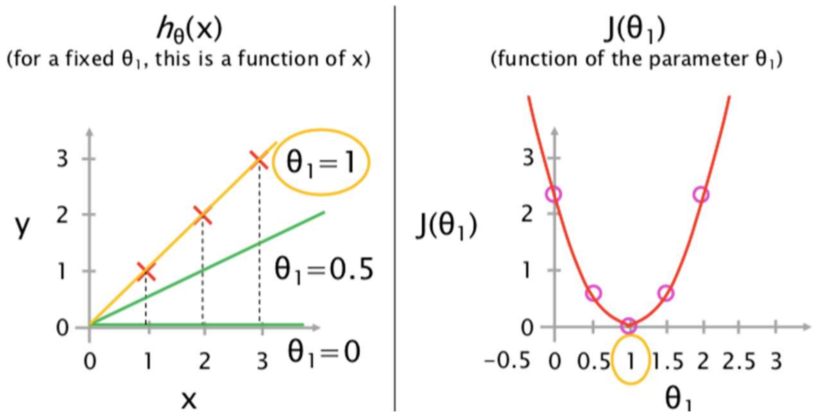
$$\Downarrow$$

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

$$\underset{\theta_0\,\theta_1}{\text{minimize}}\ J(\theta_0,\theta_1)$$

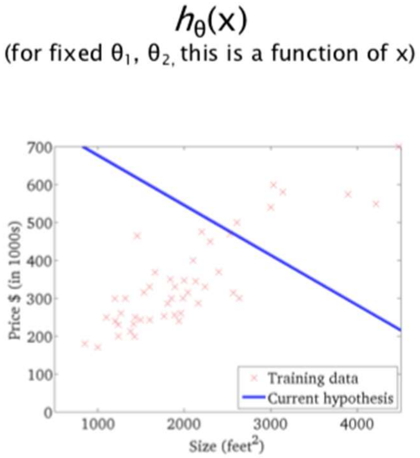
- Hypothesis
  $h(x) = \theta_0 + \theta_1 x$
- Parameters
  θ_0, θ_1
- Cost function
  $J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$
- Goal
  $\underset{\theta_0\,\theta_1}{\min}\ J(\theta_0,\theta_1)$

Provando diversi valori di θ_1 posso andare a capire come l'ipotesi riesca a intercettare i dati e di conseguenza come varia il valore della funzione di costo.

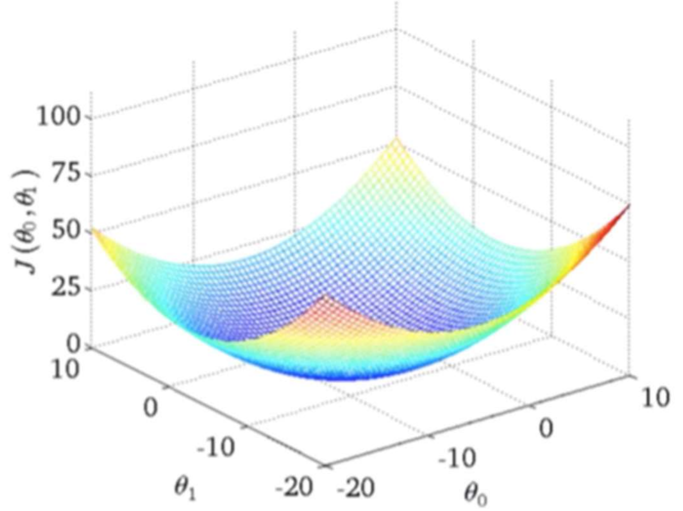


Il nostro obiettivo è trovare il minimo della funzione di costo

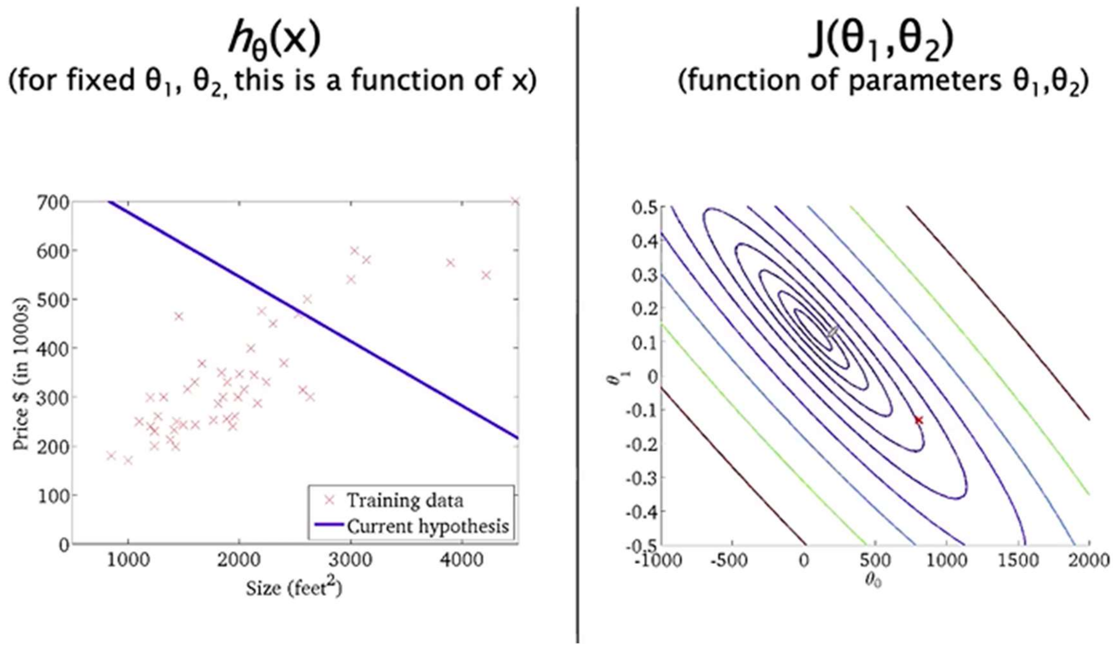
Ritornando al problema originale con i **due parametri**



Quella che prima era funzione di un solo parametro, ora lo è di due e quindi la funziona di costo diventa una superficie parabolica



Molto spesso si vanno ad utilizzare i **contour plot**: curve altimetriche, tutti i punti sullo stesso ovale hanno lo stesso valore. Più sono piccoli gli ovali, più sto andando sul fondo.



Quello che vorrei è che la macchina facesse questo per me: trovare θ_0 e θ_1 che portino la mia curva a configurarsi in maniera ideale -> porta la funzione di costo ad avere il valore minimo. *Problema di* ***ottimizzazione****.*