Linear Regression.

A number of slides and screenshots from : Andrew Ng's course on machine learning and Sebastian Raschka's course on deep learning
Both can be found for free on youtube !

What does regression mean ?

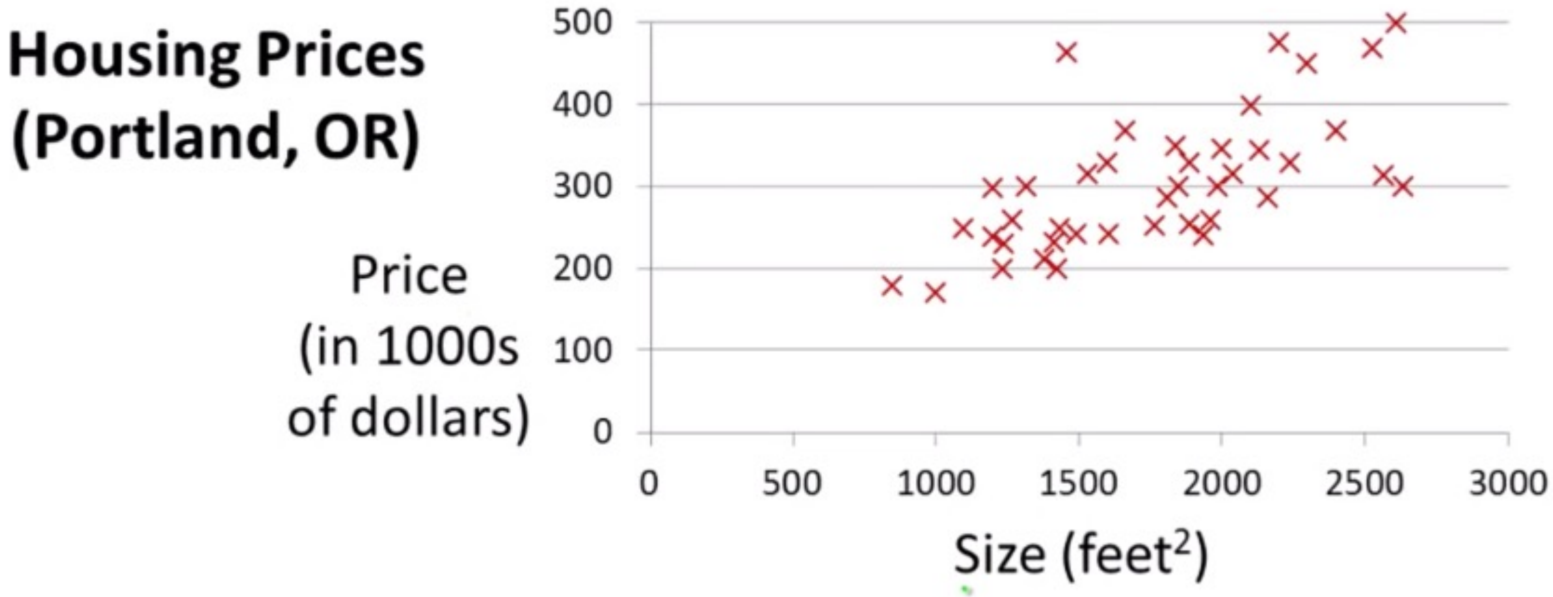
- Seen in intro, but :
- Regression means predictiong **real-valued** outputs.

- An essential type of supervised machine learning task : for each example in the data, we want to get as close as possible to the real-valued label.
- Often contrasted with classification (**discrete** labels).

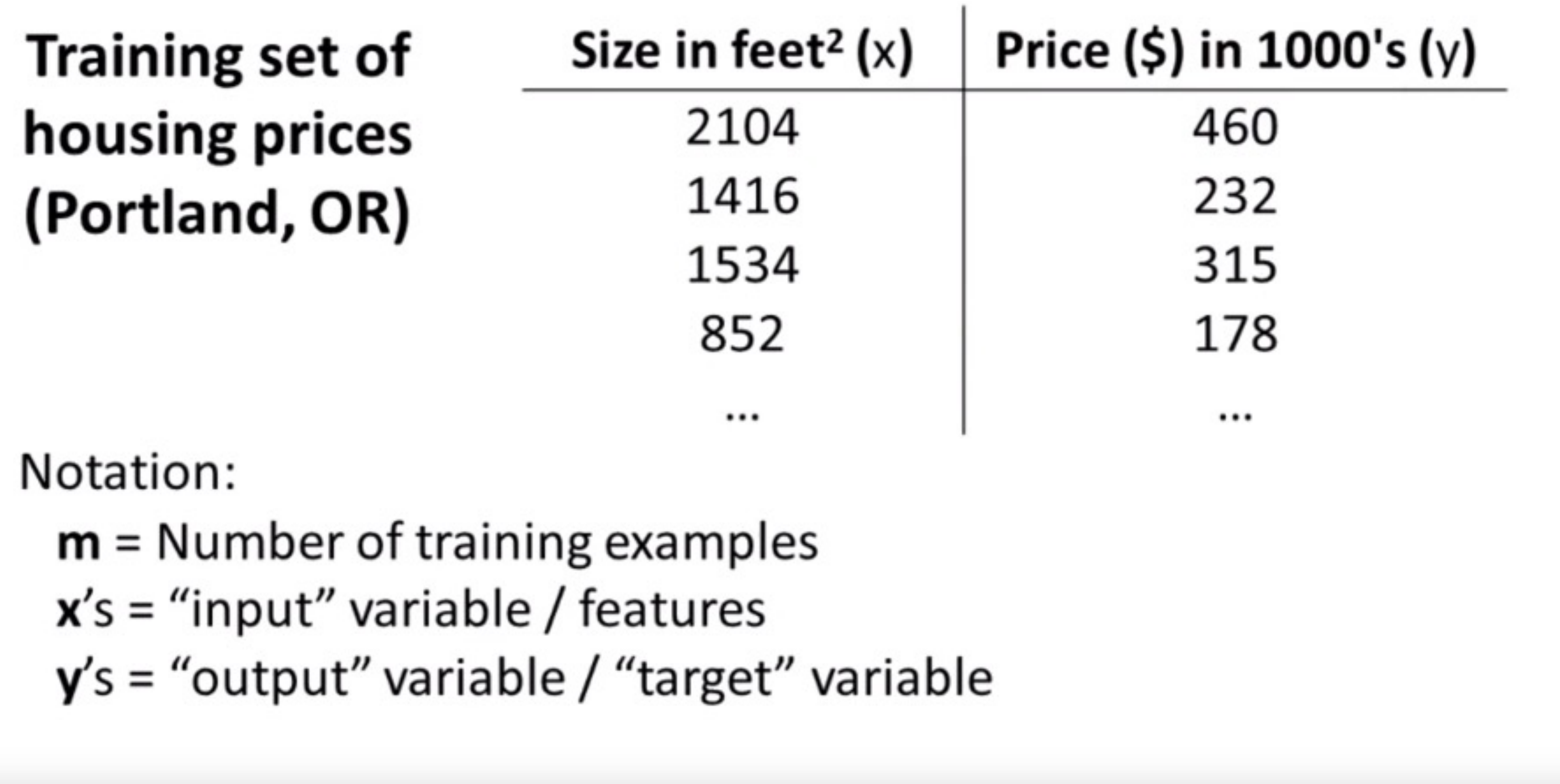
- Example :
  - Predicting height => many many real-valued outputs are possible…
  - Vs. Predicting a « height class » : short | medium-height | tall

Dataset and problem example

- Imagine we want to create an ML algorithm that predicts the price of a house using collected data, which only contains information about the size of the house.



Training Set and Notation



Training set of housing prices (Portland, OR)

Size in feet^2 (x)	**Price ($) in 1000's** (y)
2104	460
1416	232
1534	315
852	178
…	…

Notation:

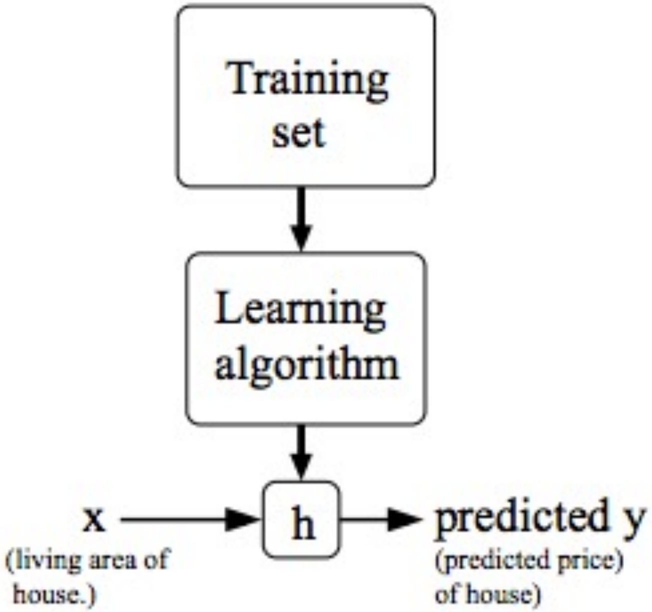
m = Number of training examples
x's = "input" variable / features
y's = "output" variable / "target" variable

The supervised learning workflow

- **h**: hypothesis
- h is a function which **maps** x’s to y’s
- Our goal will be to find the function which takes

x as input and predicts the correct y for that

x.

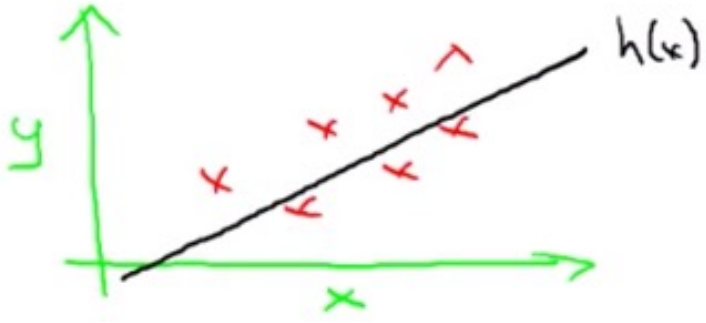


How to model *h*

- To start with, we will use a simple model, a function which corresponds to the equation of a line (maybe you remember y = ax + b?)

$$h(x) = \theta_0 + \theta_1 x$$

- This model will predict that y is some linear function (straight line) :



If this seems a bit odd to you…

- Remember we want our function to predict the examples we have in our training set correctly, which our simple model will probably not do very well….
- What if we can’t get to all the points using a straight line ?
- Don’t worry for now, this is still a very decent starting point in practice !

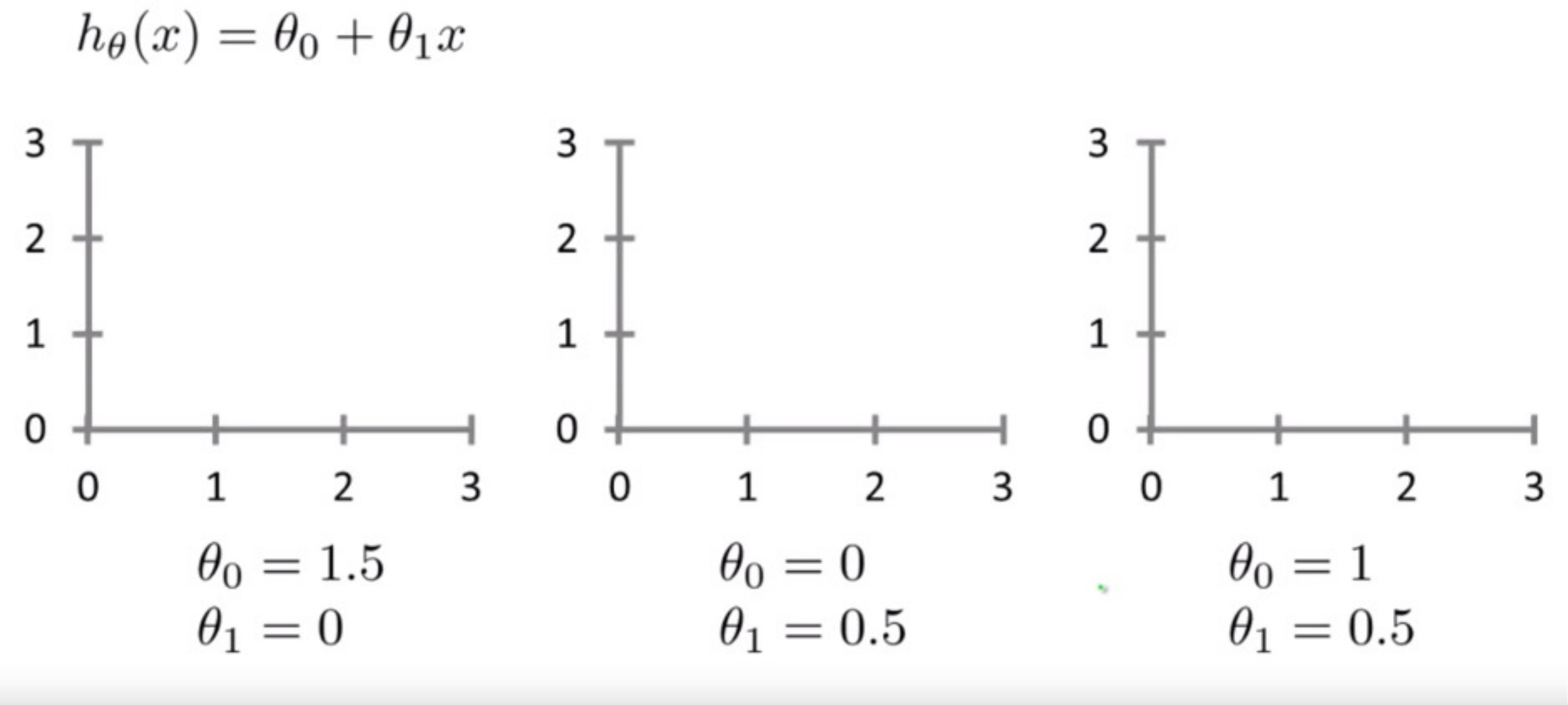
Cost Function

- This is a **second** function we will use to judge **how well** our straight **line fits** the data.
- In other words, this function will help us **find the best possible straight line.**

Motivating the Cost Function…

- To recap:
- $h(x) = \theta_0 + \theta_1 x$ is our **model**
- θ_i are what we call **parameters**
- We want to find the right combination of those parameters to get the best line.
- So **how do we choose the right parameters** ?

Visualizing different parameter choices/hypotheses



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

$\theta_0 = 1.5$
$\theta_1 = 0$

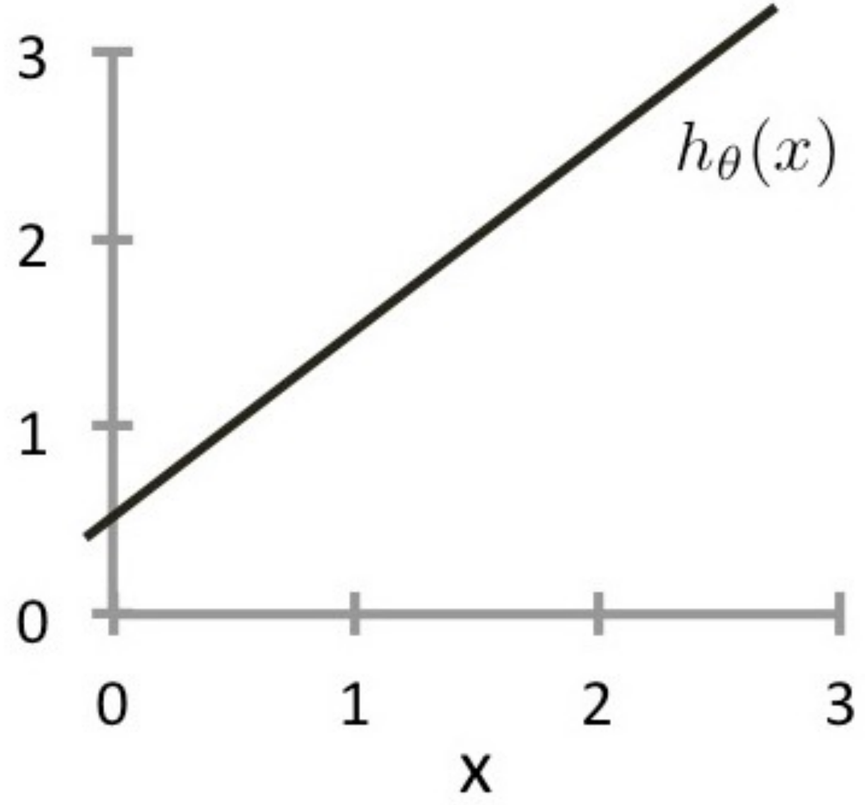
$\theta_0 = 0$
$\theta_1 = 0.5$

$\theta_0 = 1$
$\theta_1 = 0.5$

Exercise

- Look at the plot of $h(x) = \theta_0 + \theta_1 x$

- Just by eyeballing the plot, what seem to be the values of θ_0 and θ_1 ?



Finding the Cost as a Minimization Problem

- We want to choose θ_0 and θ_1 so that

- $h(x)$ is close to y for our training examples (x, y)…

- This actually comes down to a **minimization problem**,

- where we want to **minimize** $(\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{y})^{\mathbf{2}}$ for example, by tweaking our parameters θ_0 and θ_1

Cost function = Quantifying the model’s error

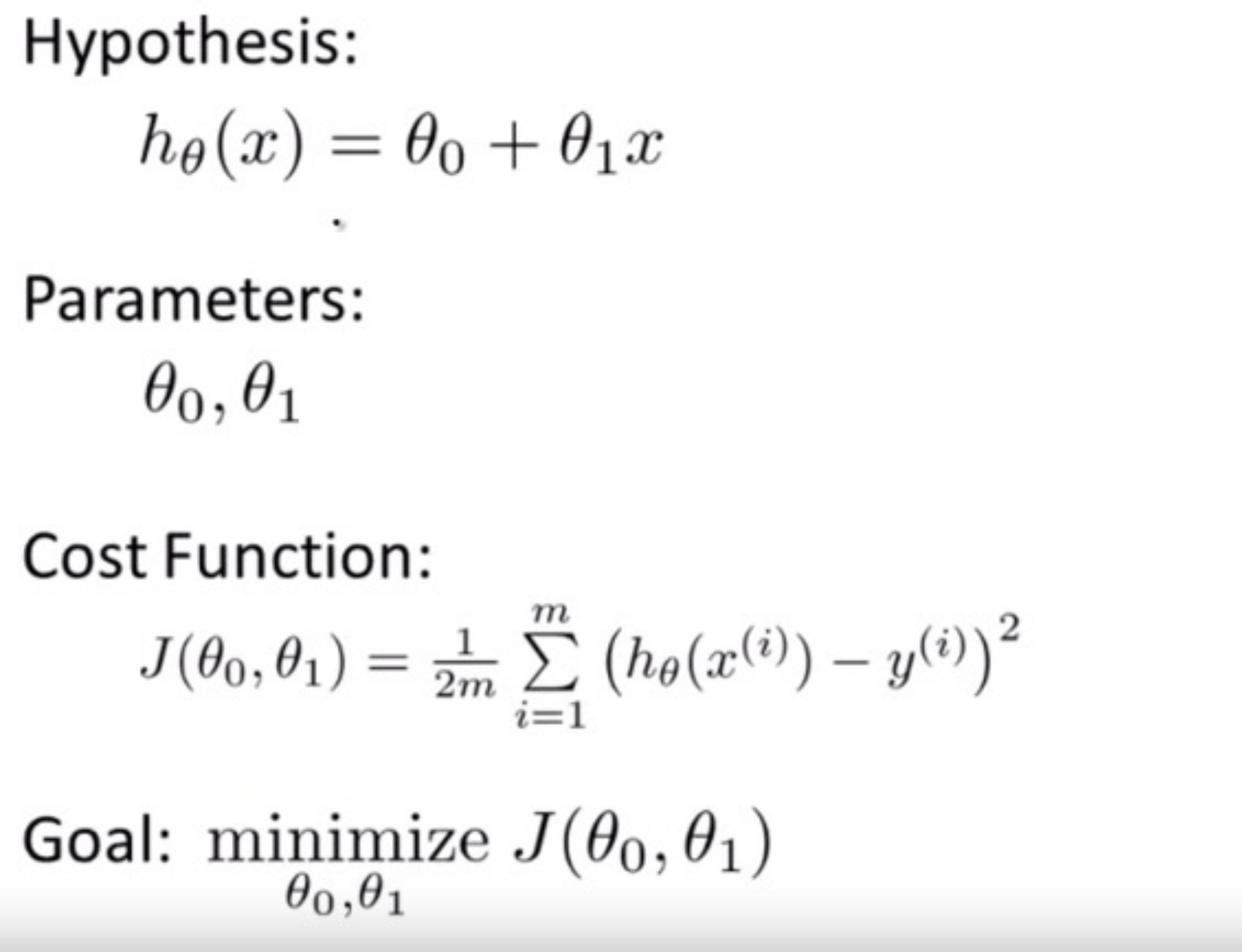
- For all of our examples m the **average error** is :

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

Picking $\frac{1}{2m}$ makes the math easier later on, but you can regard this as just an averaging constant.

- This function is known as the **Mean Squared Error** (we’ll see how it works in a few slides) and is the most **commonly** used

To recap



Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$

Cost Function:

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

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

https://www.coursera.org/learn/machine-learning/home/welcome

Cost Function Intuition

- Let’s use a **simplified model hypothesis** to understand what’s going on a bit better:

$$h(x) \;= \theta_1 x$$

- Our objective is now to minimize

$$J(\theta_1)$$

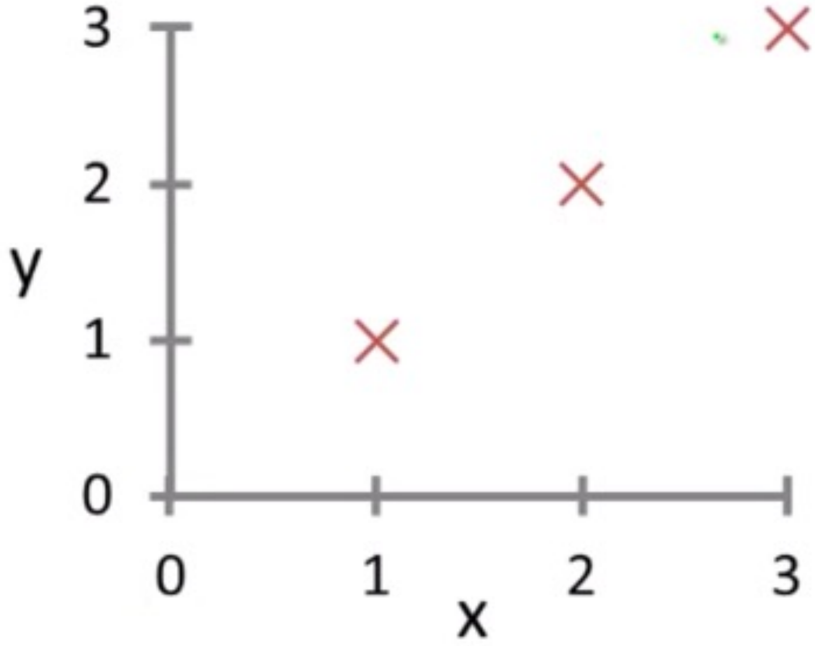
- Which is equal to

$$\frac{1}{2m}\sum_{i=1}^{m}(\,\theta_1 x^i - y^i)^2$$

Hypothesis function vs. Cost function

- If the points on the graph represent our training data and $\theta_1 = 1,$ what does our **hypothesis** (line) look like ?

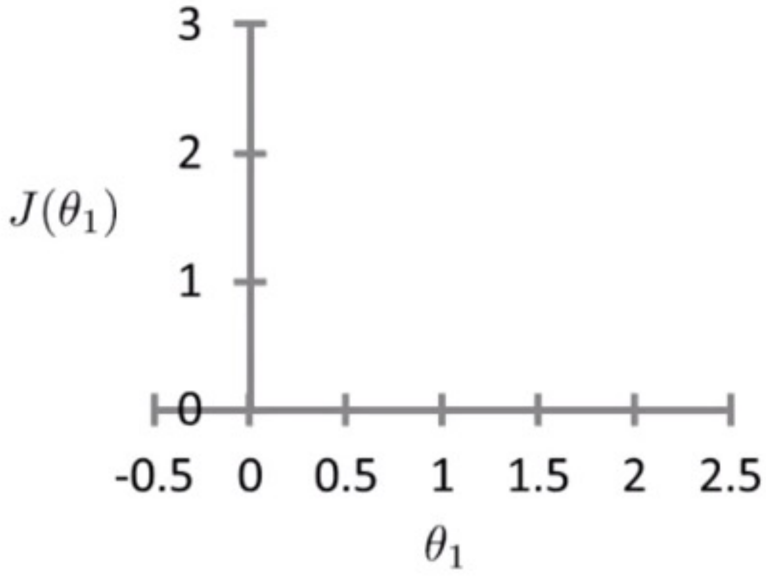
- What is the cost ?
- Remember : $\frac{1}{2m}\sum_{i=1}^{m}(\theta_1 x^i - y^i)^2$



Hypothesis function vs. Cost function

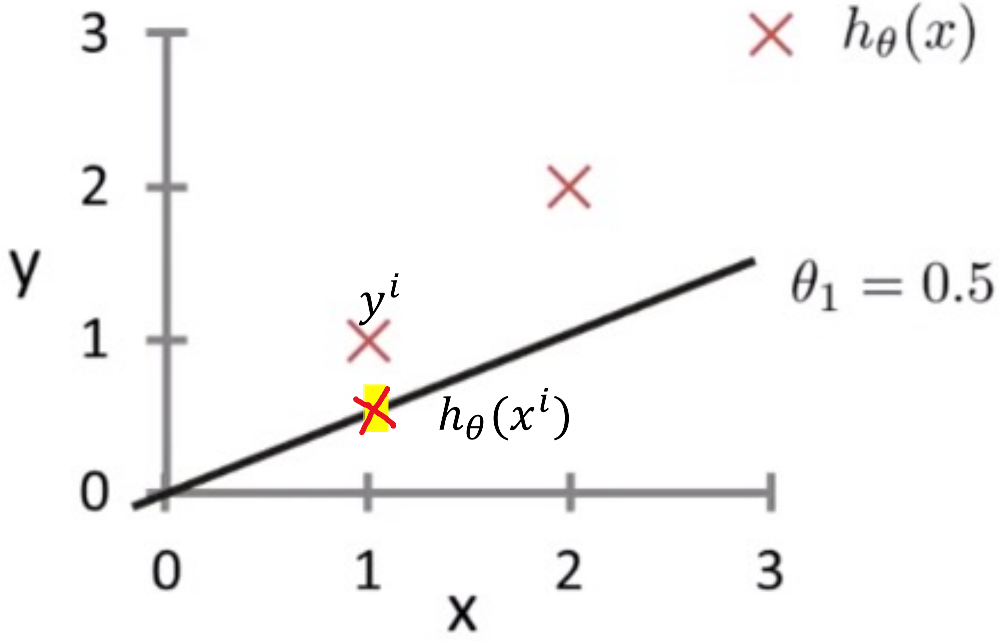
- $J(\theta_1 = 1) = \mathbf{0}$
- We can now **plot** our error rate
- Notice that the values for $\boldsymbol{\theta_1}$ **are on the horizontal axis**. This is **not the same** plot as before !!

- This is a plot for the **cost function** :



Hypothesis function vs. Cost function

- Now let’s look at $\theta_1 = 0.5$
- And compute $J(\theta_1 = 0.5)$ (approx. 0.58)
- The error for each point is actually the **height** which **seperates** the **data point from the line** for a giver

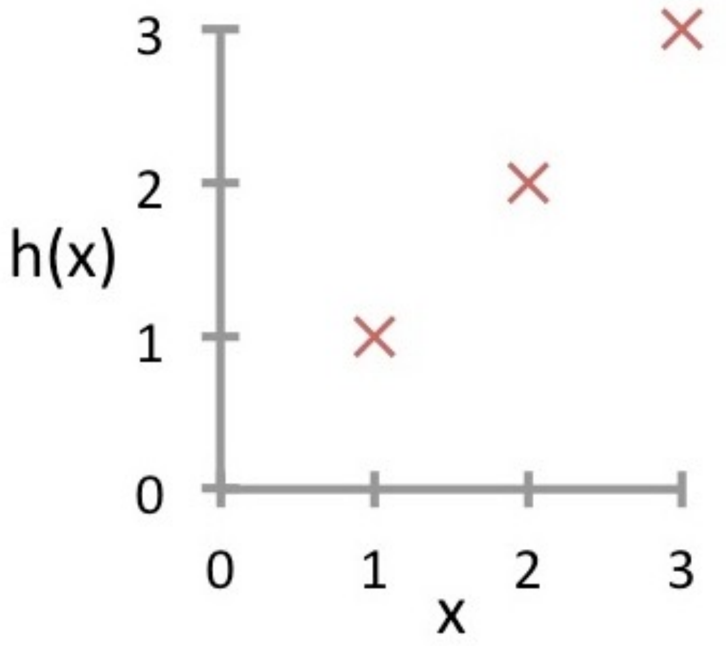


Your turn !

- Suppose this is our training set. $m = 3$
- Given the same hypothesis and cost

functions as before, what is $J(0)$?

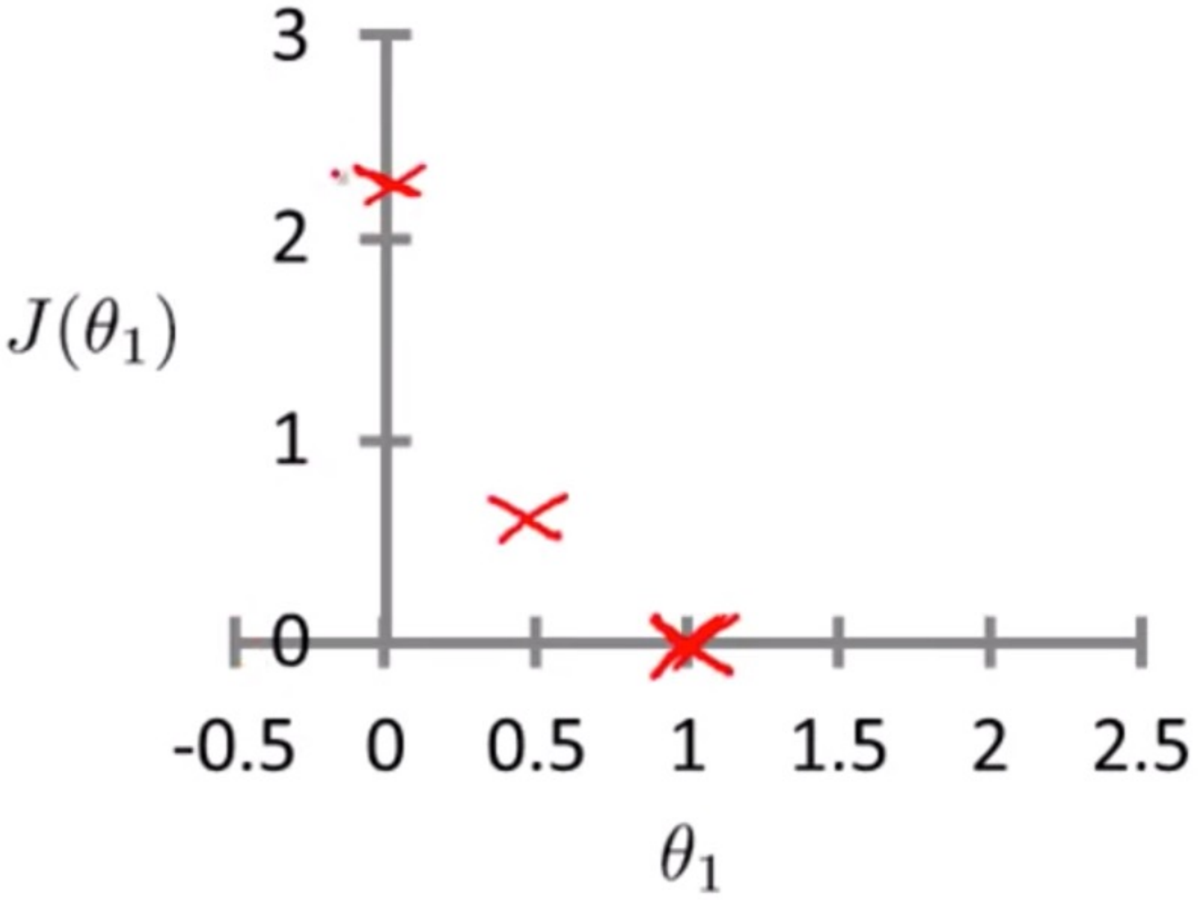
ie. $\theta_1 = 0$



- Should be approx. 2.3

Hypothesis function vs. Cost function

- We could continue plotting points but we'll stop here.
- With the error calculated for the different values of θ_1, we start to see part of the **general shape** of the function.
- It turns out the function is convex/looks like a parabola.



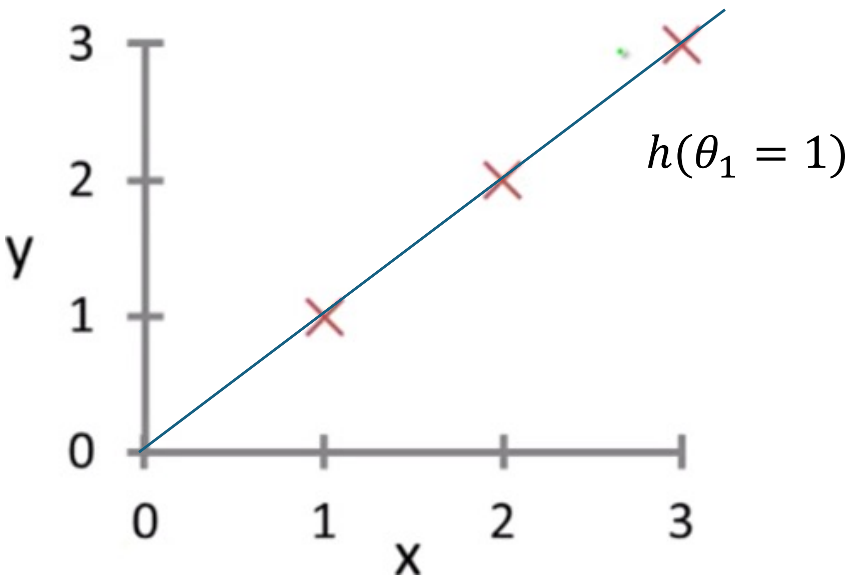
Quick cost function recap

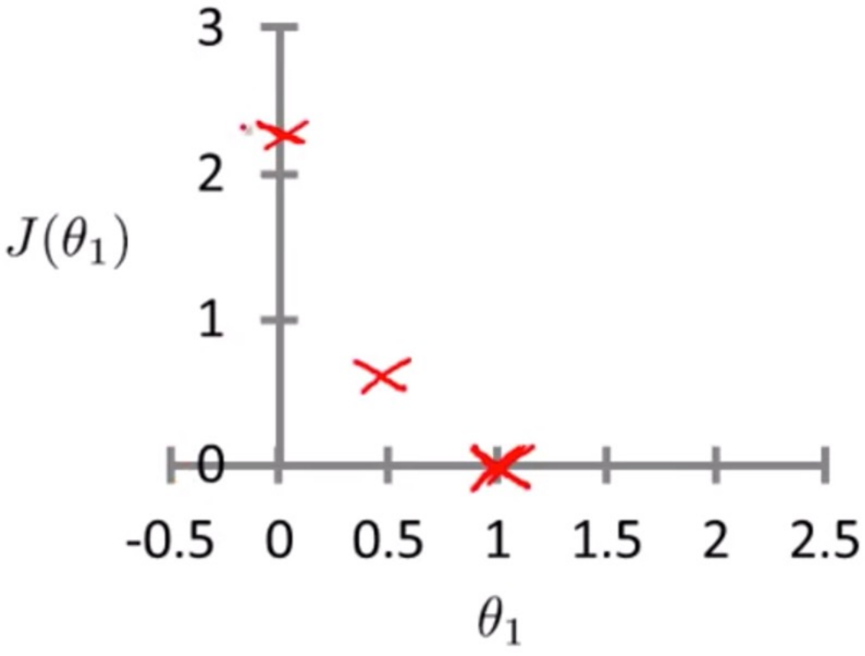
- Each **value** of $\theta_1$plotted corresponds to a **different hypothesis / model**

- For each value of θ_1 we can compute a value $J(\theta_1)$ to trace out the cost function.

- Now remember, we wanted to find the value of θ_1 which **minimized** $J(\theta_1)$… Looking at the graph we can now do so !

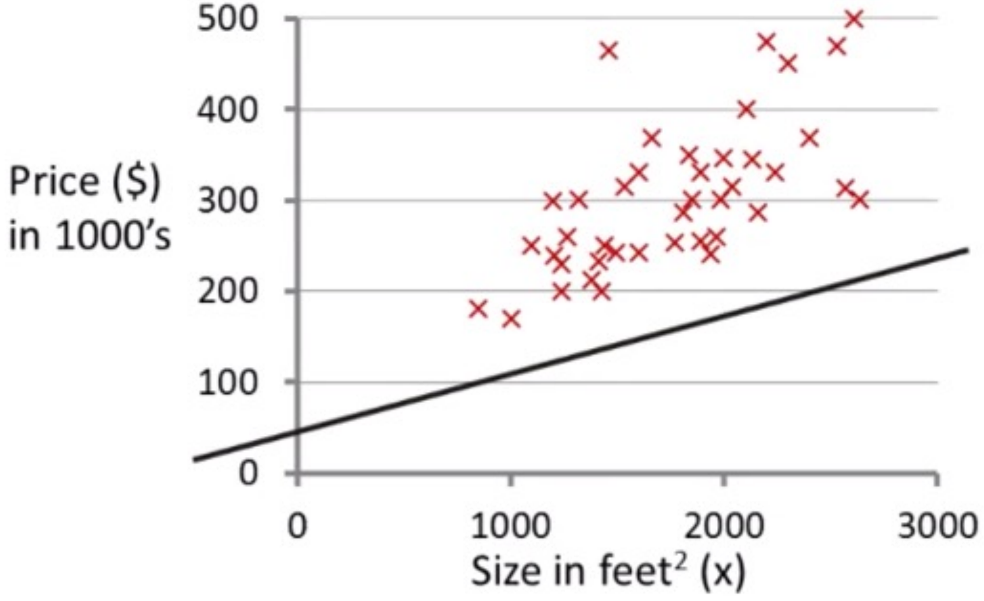
- No surprise, the value of θ_1 which minimizes the error is associated with the **model which fits the data perfectly**





Back to 2 parameters

- Now, going back to our original data and model, we use a 2 parameter hypothesis to draw our line
- For :
- $\theta_0 = 50$
- $\theta_1 = 0.06$
- We get this straight line as our model



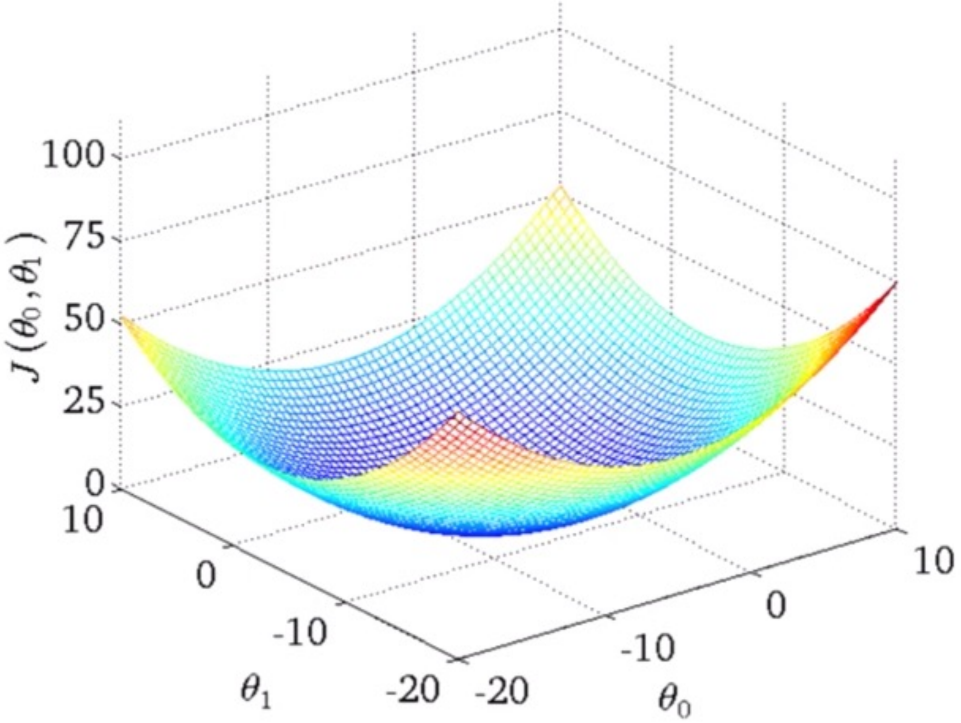
$$h_\theta(x) = 50 + 0.06x$$

Corresponding Cost function

- Now we have **two parameters**, the error graph will be slightly harder to plot as it has **3 dimensions**:

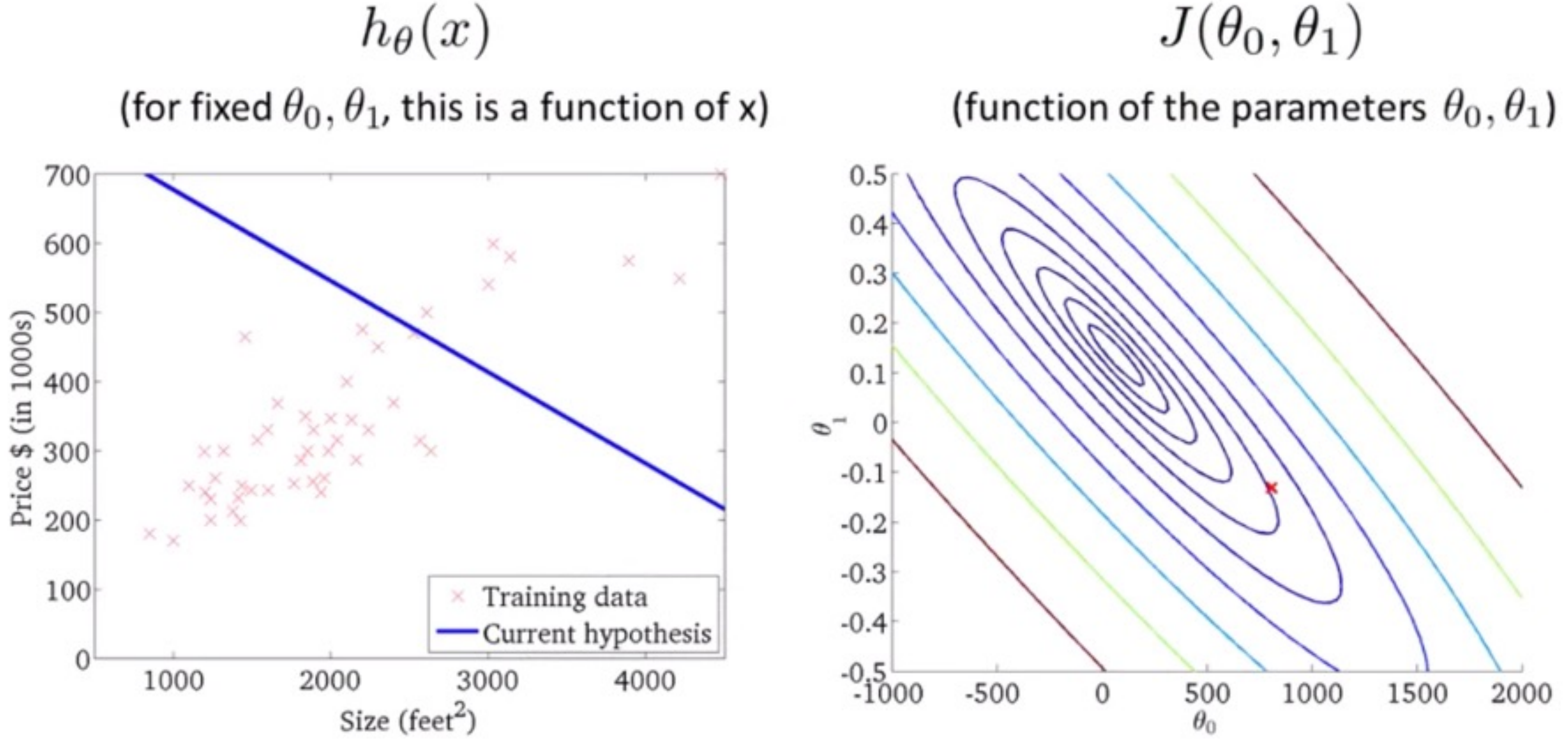
$$\theta_1, \theta_2, cost$$

- Indeed , $J(\theta_1, \theta_2)$ now has 2 inputs,
- So it will look like this in 3D:



Contour Plots

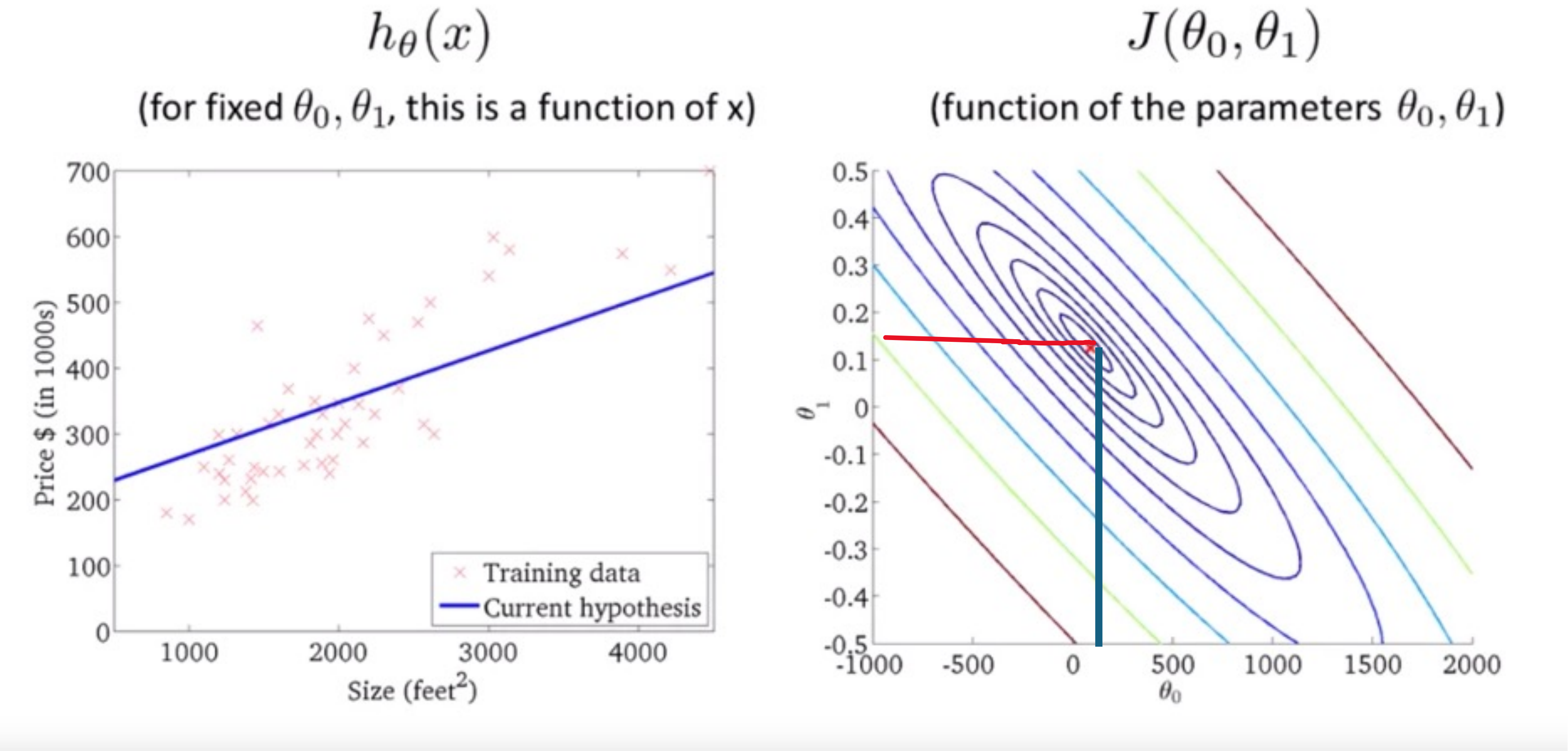
- You will sometimes see the cost function represented by a contour plot ·



The ovals/ellipses show the set of points which take on the same value for given values of θ_0, θ_1

Countour Plots

- The **minimum** is at the **center** of all the « ellipses ».
- This plot shows a model very close to the minimum.



$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)