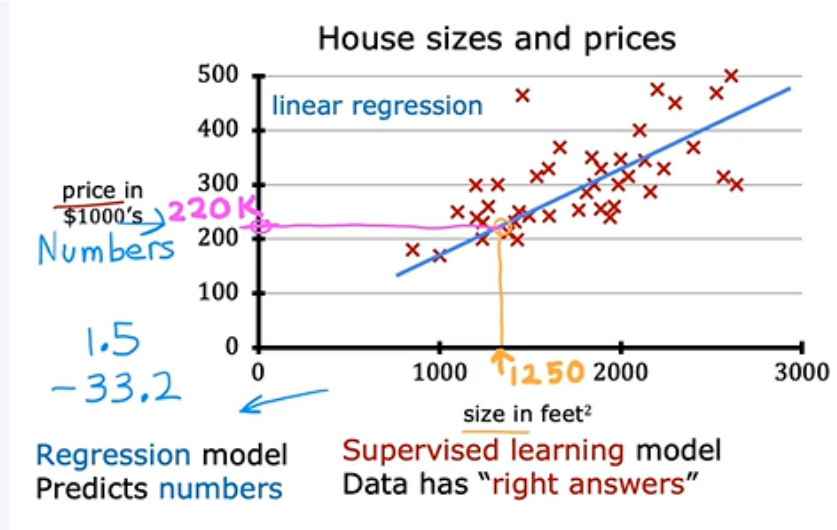
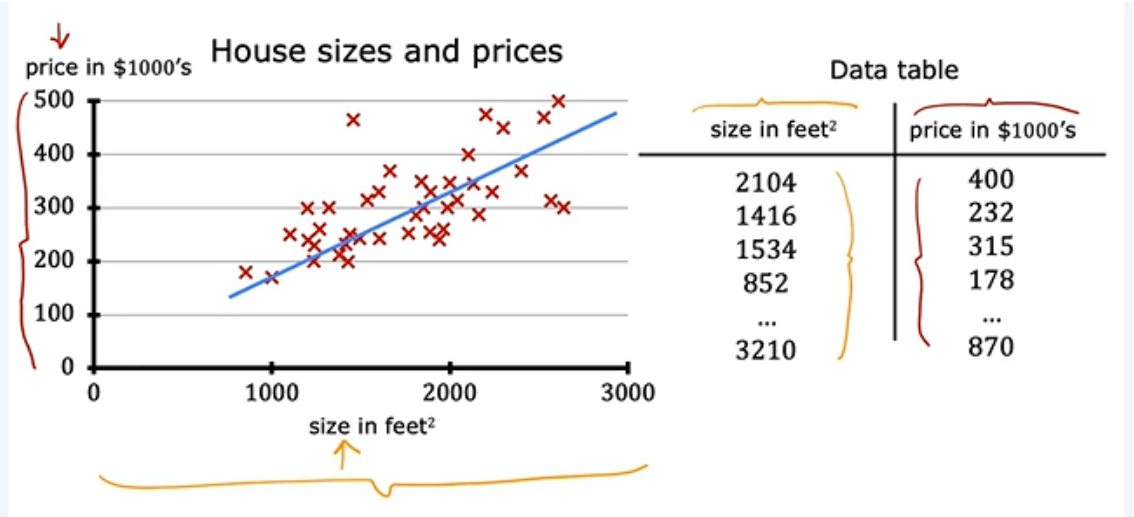
Regression model

Linear regression model



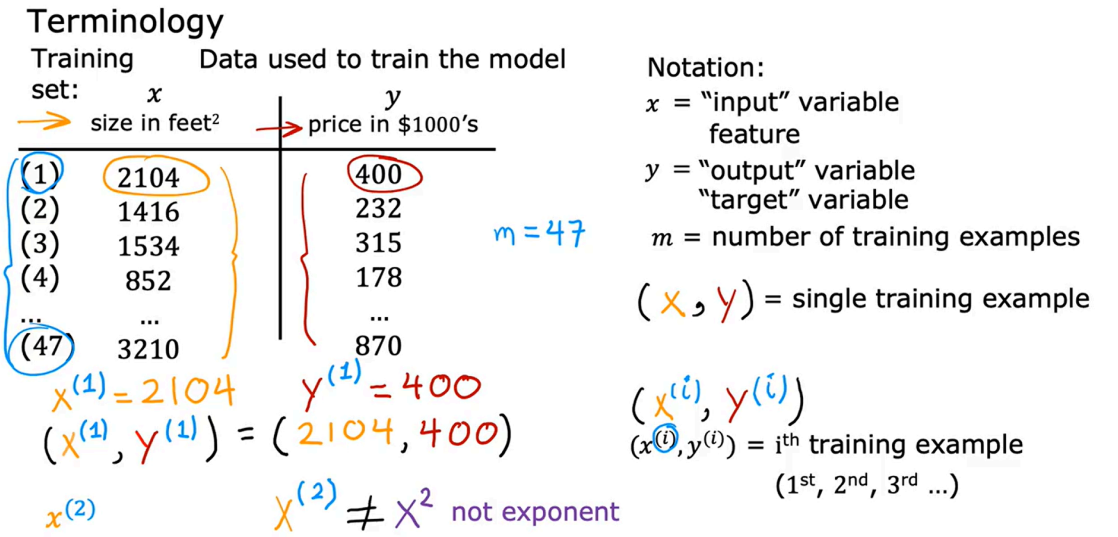
- Any supervised learning model that predicts a number such as 220,000 or 1.5 or negative 33.2 is addressing what's called a regression problem.
- Linear regression is one example of a regression model. But there are other models for addressing regression problems too.
- Just to remind you, in contrast with the regression model, the other most common type of supervised learning model is called a classification model.
    - Classification model predicts categories or discrete categories, such as predicting if a picture is of a cat, meow or a dog, woof, or if given medical record, it has to predict if a patient has a particular disease.



- In addition to visualizing this data as a plot here on the left, there's one other way of looking at the data that would be useful, and that's a data table here on the right.
- The data comprises a set of inputs. This would be the size of the house, which is this column here. It also has outputs. You're trying to predict the price, which is this column here. Notice that the horizontal and vertical axes correspond to these two columns, the size and the price.
- If you have, say, 47 rows in this data table, then there are 47 of these little crosses on the plot of the left, each cross corresponding to one row of the table.
- For example, the first row of the table is a house with size, 2,104 square feet, so that's around here, and this house is sold for $400,000 which is around here.

The dataset that you just saw and that is used to train the model is called a training set.

Note that your client's house is not in this dataset because it's not yet sold, so no one knows what the price is. To predict the price of your client's house, you first train your model to learn from the training set and that model can then predict your client's houses price.



In Machine Learning, the standard notation to denote the input here is lowercase x, and we call this the input variable, is also called a feature or an input feature.

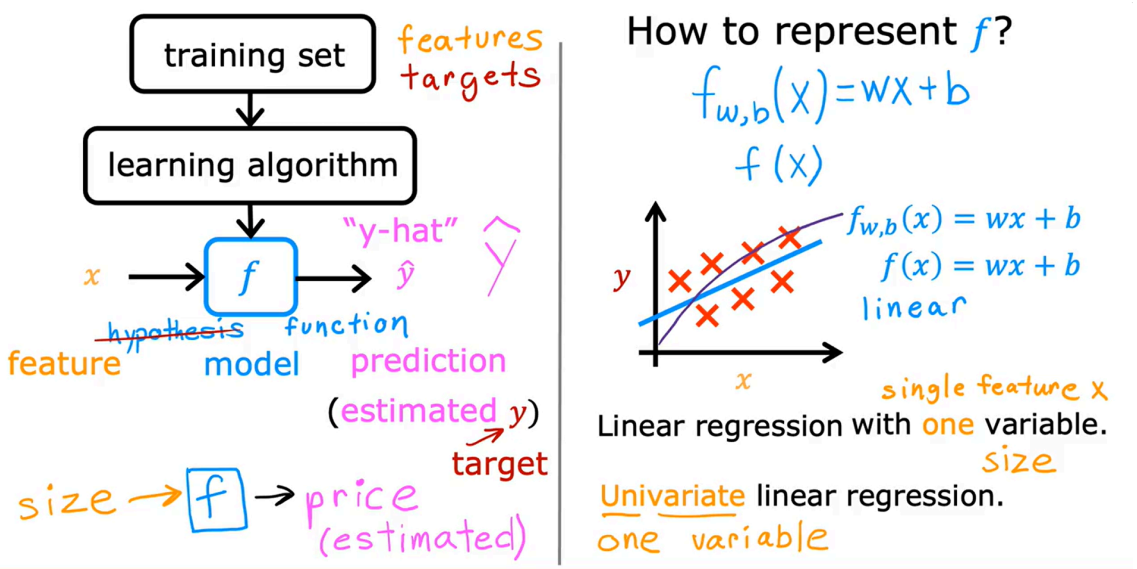
For example, for the first house in your training set, x is the size of the house, so x equals 2,104. The standard notation to denote the output variable which you're trying to predict, which is also sometimes called the target variable, is lowercase y. Here, y is the price of the house, and for the first training example, this is equal to 400, so y equals 400.

The dataset has one row for each house and in this training set, there are 47 rows with each row representing a different training example. We're going to use lowercase m to refer it to the total number of training examples, and so here m is equal to 47.

To indicate the single training example, we're going to use the notation parentheses x, y. For the first training example, (x, y), this pair of numbers is (2104, 400). Now we have a lot of different training examples. We have 47 of them in fact. To refer to a specific training example, this will correspond to a specific row in this table on the left,

I'm going to use the notation x superscript in parenthesis, i, y superscript in parentheses i. The superscript tells us that this is the ith training example, such as the first, second, or third up to the 47th training example. i here, refers to a specific row in the table. For instance, here is the first example, when i equals 1 in the training set, and so x superscript 1 is equal to 2,104 and y superscript 1 is equal to 400 and let's add this superscript 1 here as well.

Just to note, this superscript i in parentheses is not exponentiation. When I write this, this is not x squared. This is not x to the power 2. It just refers to the second training example. This i, is just an index into the training set and refers to row i in the table. In this video, you saw what a training set is like, as well as a standard notation for describing this training set.



- Supervised learning algorithm will input a dataset and then what exactly does it do and what does it output? Let's find out in this video.
- Recall that a training set in supervised learning includes both the input features, such as the size of the house and also the output targets, such as the price of the house. The output targets are the right answers to the model we'll learn from.
- To train the model, you feed the training set, both the input features and the output targets to your learning algorithm. Then your supervised learning algorithm will produce some function. We'll write this function as lowercase f, where f stands for function. Historically, this function used to be called a hypothesis, but I'm just going to call it a function f in this class.
- The job with f is to take a new input x and output and estimate or a prediction, which I'm going to call y-hat (y mũ), and it's written like the variable y with this little hat symbol on top. In machine learning, the convention is that y-hat is the estimate or the prediction for y. The function f is called the model. X is called the input or the input feature, and the output of the model is the prediction, y-hat. The model's prediction is the estimated value of y.

- When the symbol is just the letter y, then that refers to the target, which is the actual true value in the training set. In contrast, y-hat is an estimate. It may or may not be the actual true value. Well, if you're helping your client to sell the house, well, the true price of the house is unknown until they sell it. Your model f, given the size, outputs the price which is the estimator, that is the prediction of what the true price will be.
- Now, when we design a learning algorithm, a key question is, how are we going to represent the function f? Or in other words, what is the math formula we're going to use to compute f? For now, let's stick with f being a straight line. You're function can be written as f_{w, b} of x equals, I'm going to use w times x plus b ($f_{w,b}(X) = w.X + b$). I'll define w and b soon. But for now, just know that w and b are numbers, and the values chosen for w and b will determine the prediction y-hat based on the input feature x. This f_w b of x means f is a function that takes x as input, and depending on the values of w and b, f will output some value of a prediction y-hat.
- As an alternative to writing this, f_w, b of x, I'll sometimes just write f of x without explicitly including w and b into subscript. Is just a simpler notation that means exactly the same thing as f_w b of x.
- Let's plot the training set on the graph where the input feature x is on the horizontal axis and the output target y is on the vertical axis. Remember, the algorithm learns from this data and generates the best-fit line like maybe this one here. This straight line is the linear function f_w b of x equals w times x plus b. Or more simply, we can drop w and b and just write f of x equals wx plus b. Here's what this function is doing, it's making predictions for the value of y using a streamline function of x.
- You may ask, why are we choosing a linear function, where linear function is just a fancy term for a straight line instead of some non-linear function like a curve or a parabola? Well, sometimes you want to fit more complex non-linear functions as well, like a curve like this. But since this linear function is relatively simple and easy to work with, let's use a line as a foundation that will eventually help you to get to more complex models that are non-linear. This particular model has a name, it's called linear regression.
- More specifically, this is linear regression with one variable, where the phrase one variable means that there's a single input variable or feature x,

namely the size of the house. Another name for a linear model with one input variable is univariate linear regression, where uni means one in Latin, and where variate means variable. Univariate is just a fancy way of saying one variable.

Cost function formula

- The cost funtion tell us how well the model is doing so that we can try to get it to do better:
- We already have that

$$\hat{y}^{(i)} = f_{w,b}\left(x^{(i)}\right)$$

$$f_{w,b}\left(x^{(i)}\right) = wx^{(i)} + b$$

And we need to find w, b:

$\hat{y}^{(i)}$ is close to $y^{(i)}$ for all $(x^{(i)}, y^{(i)})$

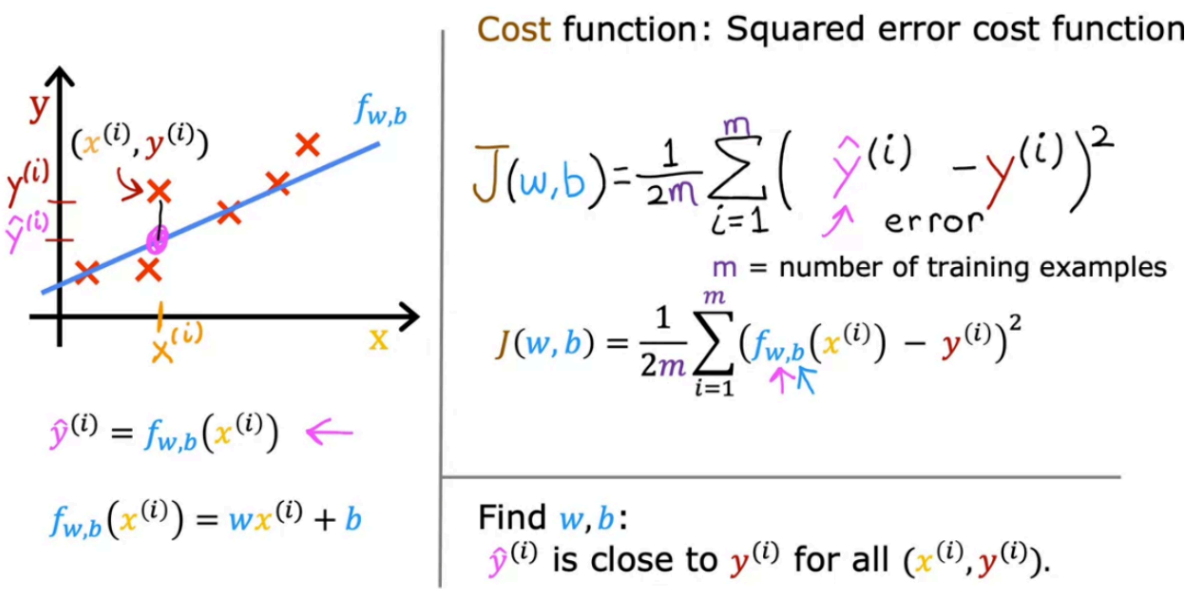
Note:

- $y^{(i)}$ is the true value
- $\hat{y}^{(i)}$ is the predict value

How the question is how do you find values for w and b so that the prediction y hat i is close to the true target?

- To answer that question, let's first take a look at how to measure how well a line fits the training data. To do that, we're going to construct a cost function.

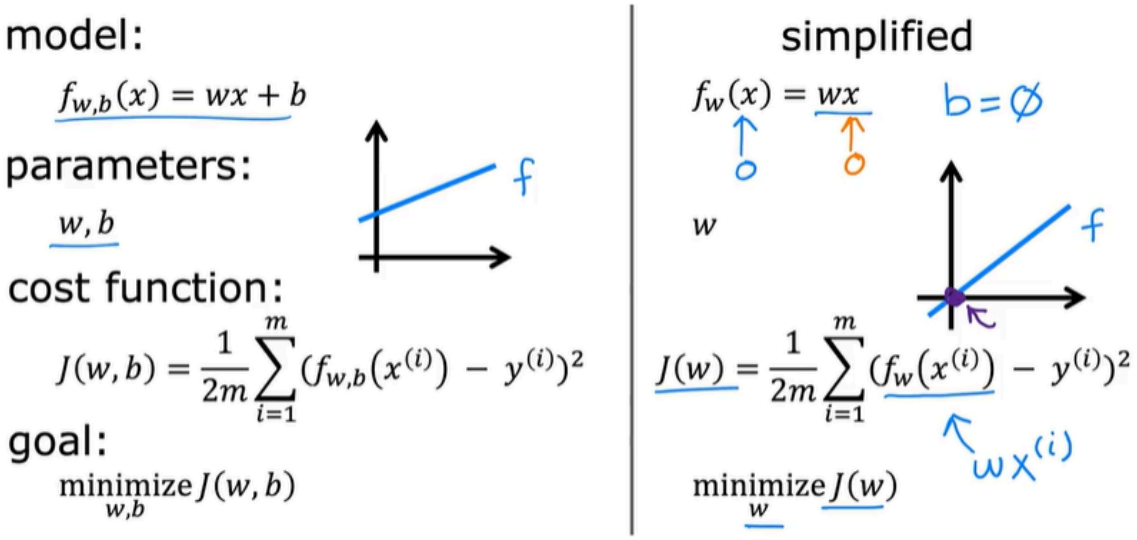
Cost function



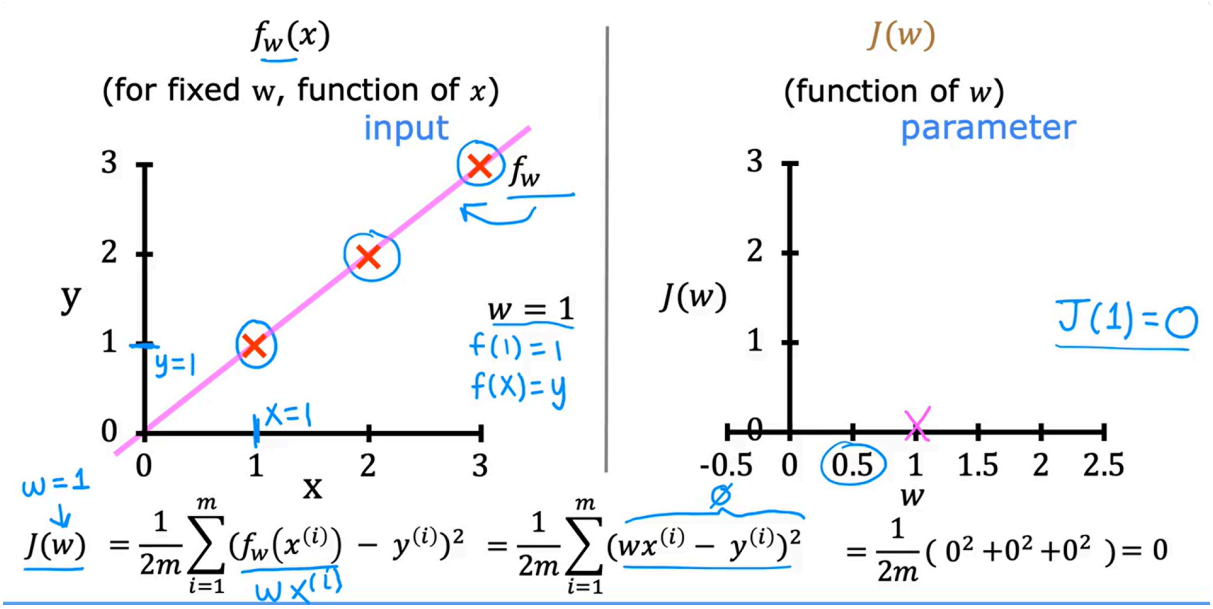
- In machine learning different people will use different cost functions for different applications, but the squared error cost function is by far the most commonly used one for linear regression and for that matter, for all regression problems where it seems to give good results for many applications.

Cost function intuition

Try simplifying the model and see what change

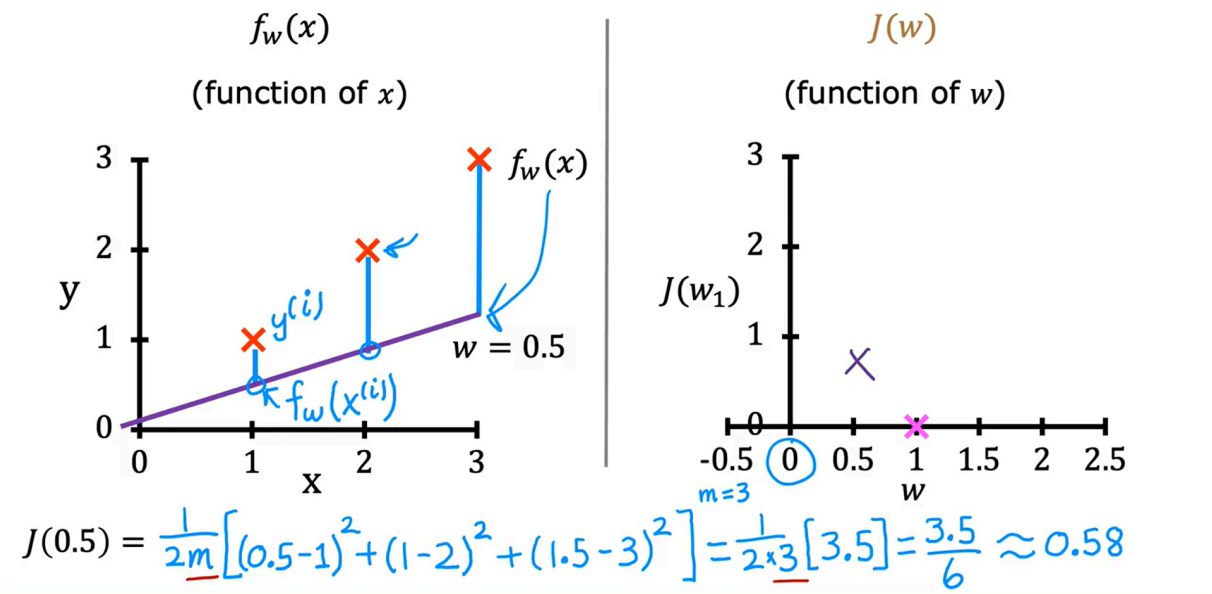


Let's assume that w is 1 and b is 0 (simplified):

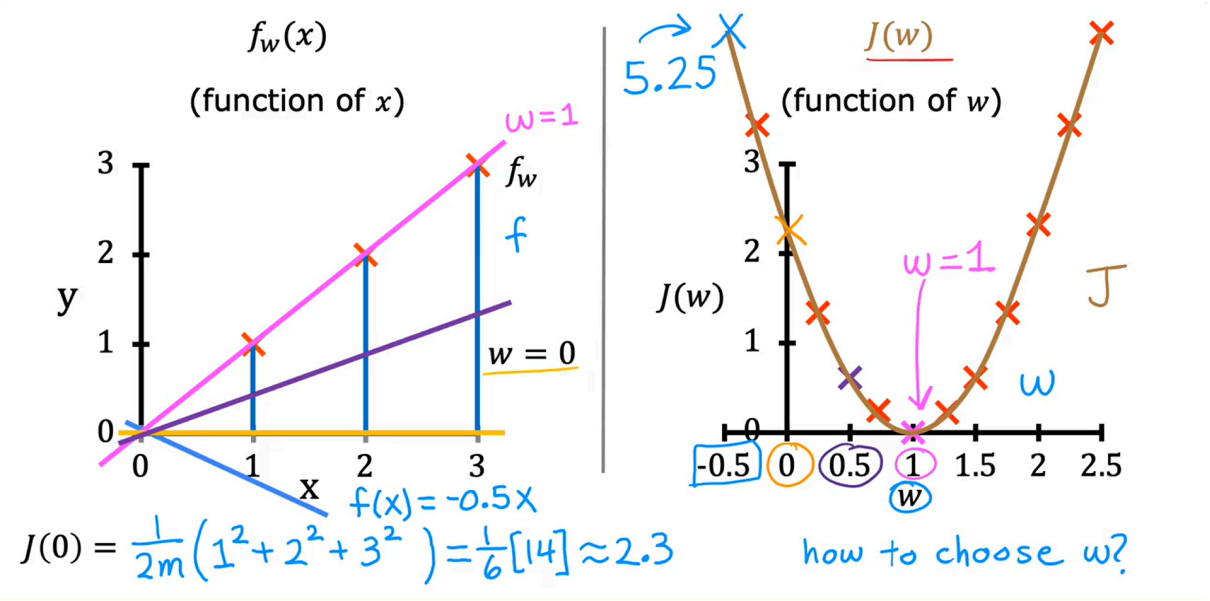


- We can see that with w = 1, the value of cost funtion J(1) = 0

Let's assume that w is 0.5 and b is 0 (simplified):



How about w = 0?



- The cost J when w equals 0 is 1 over 2m times the quantity, 1^2 plus 2^2 plus 3^2, and that's equal to 1 over 6 times 14, which is about 2.33.
- Let's plot this point where w is 0 and J of 0 is 2.33 over here. You can keep doing this for other values of w.
- Since w can be any number, it can also be a negative value. If w is - 0.5, then the line f is a downward-sloping line like this (the blue line goes through the origin). It turns out that when w is negative 0.5 then you end up with an even higher cost, around 5.25, which is this point up here (blue X).
- You can continue computing the cost function for different values of w and so on and plot these. It turns out that by computing a range of values, you can slowly trace out what the cost function J looks like and that's what J is.
- To recap, each value of parameter w corresponds to different straight line fit, f of x, on the graph to the left. For the given training set, that choice for a value of w corresponds to a single point on the graph on the right because for each value of w, you can calculate the cost J of w.
- For example, when w equals 1, this corresponds to this straight line fit through the data and it also corresponds to this point on the graph of J, where w equals 1 and the cost J of 1 equals 0. Whereas when w equals 0.5, this gives you this line which has a smaller slope. This line in combination with the training set corresponds to this point on the cost function graph at w equals 0.5. For each value of w you wind up with a different line and its corresponding costs, J of w, and you can use these points to trace out this plot on the right.

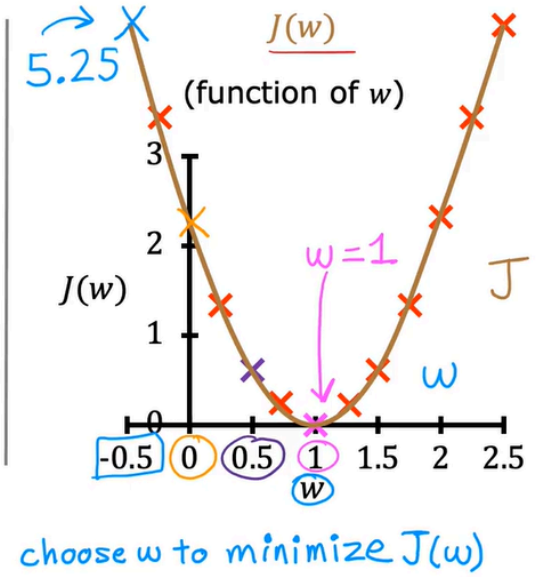
- Given this, how can you choose the value of w that results in the function f, fitting the data well?

goal of linear regression:

$$\underset{w}{\text{minimize}}\, J(w)$$

general case:

$$\underset{w,b}{\text{minimize}}\, J(w,b)$$



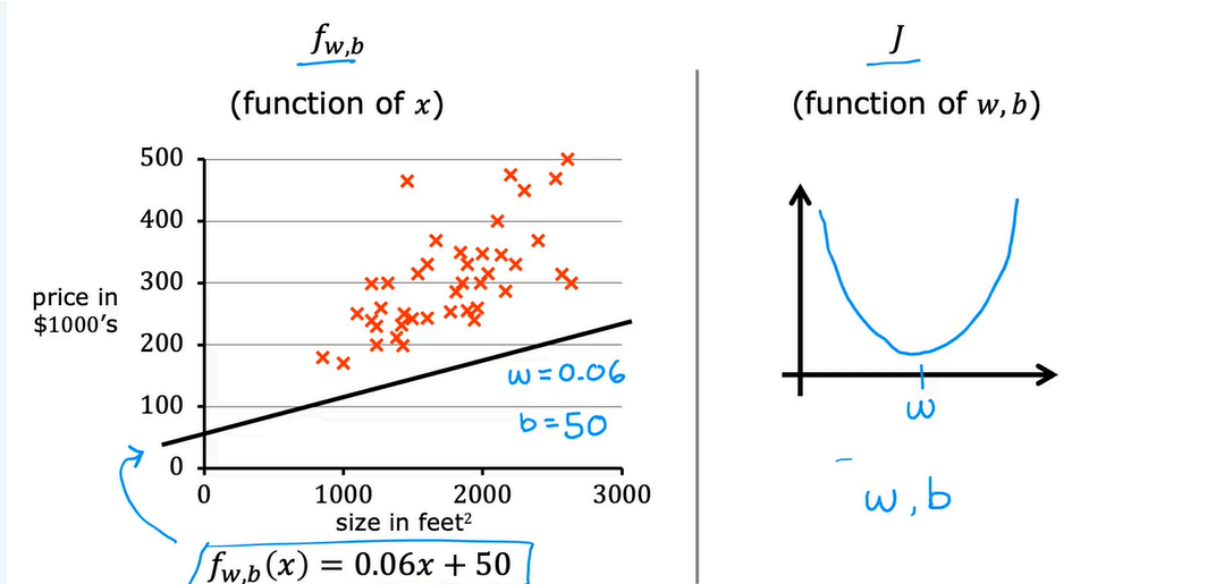
The answer is that: Find w to minimize J(w), could also use image to find

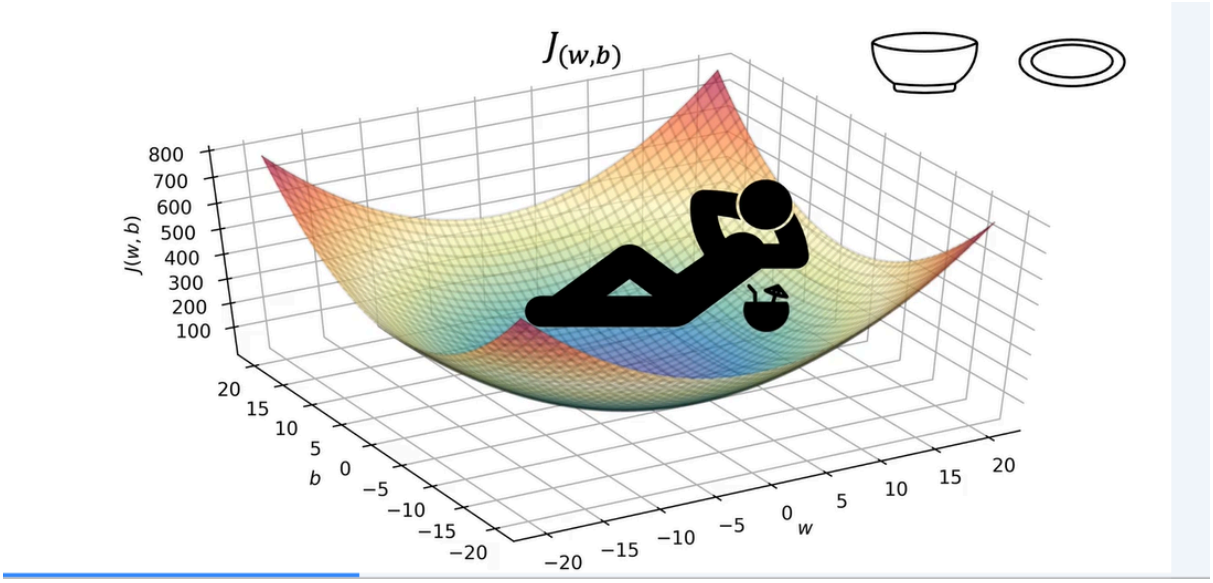
Visualizing the cost function

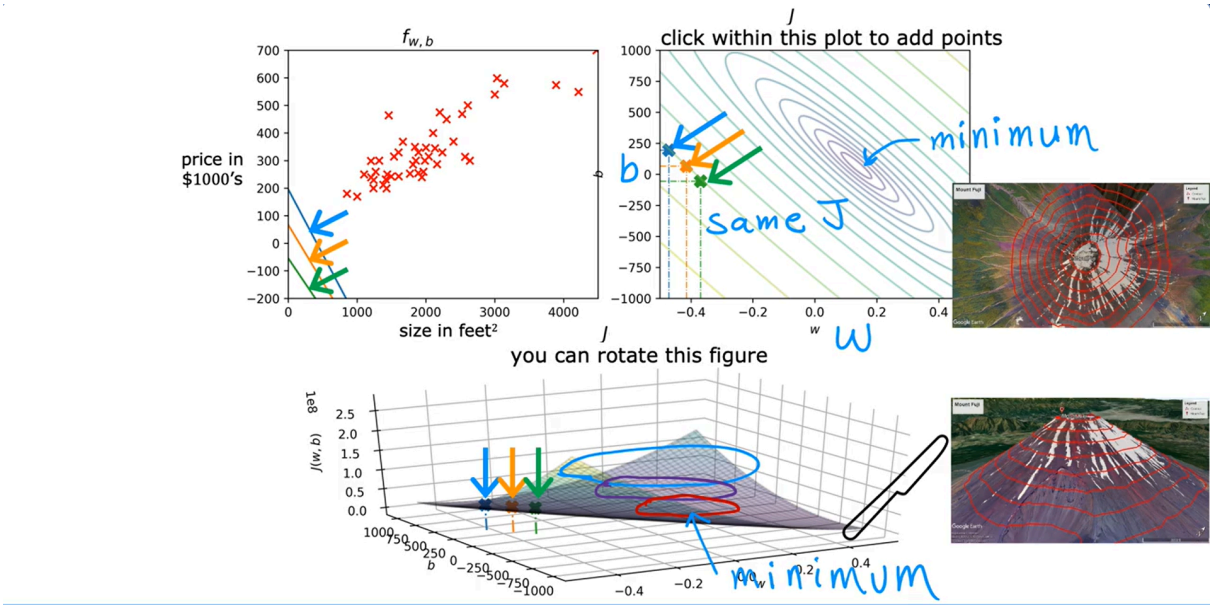
Model	$f_{w,b}(x) = wx + b$
Parameters	w, b ~~before: b=0~~
Cost Function	$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$
Objective	$\underset{w,b}{\text{minimize}}\, J(w,b)$

- In the last video, we had temporarily set b to zero in order to simplify the visualizations.
- Now, let's go back to the original model with both parameters w and b without setting b to be equal to 0

Housing example:






$f_{w,b}$
price in $1000's
size in feet²
J
click within this plot to add points
b
w
minimum
same J
J
you can rotate this figure
1e8
J(w, b)
b
w
minimum