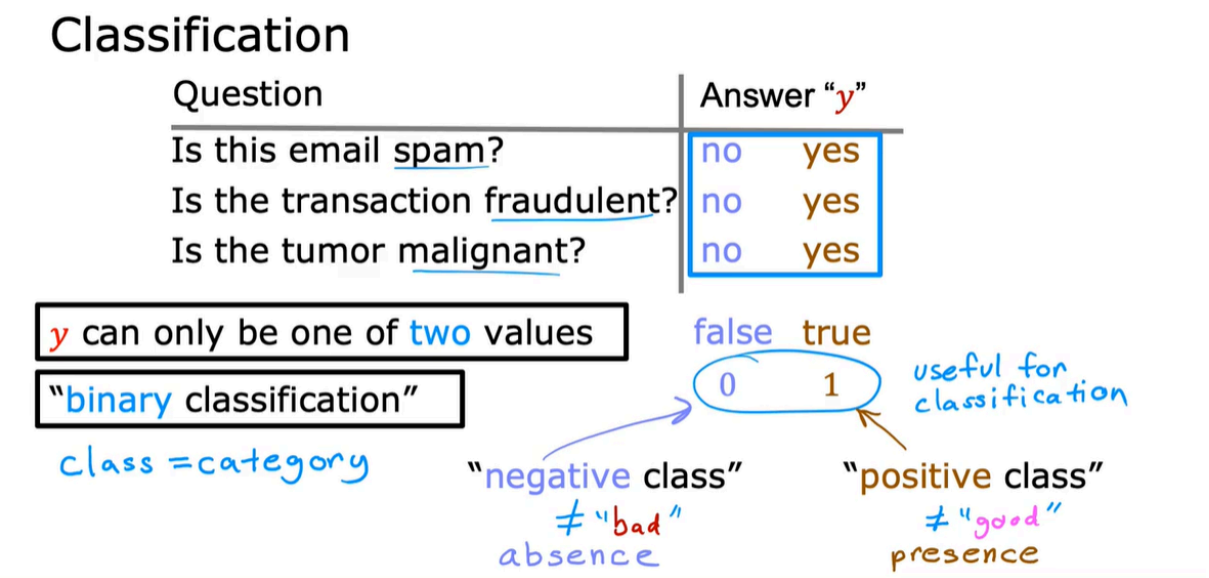
Classification with logistic regression

This week, you learn about classification where your output variable y can take on only one of a small handful of possible values instead of any number in an infinite range of numbers. It turns out that linear regression is not a good algorithm for classification problems. Let's take a look at why and this will lead us into a different algorithm called **logistic regression.**

Motivations

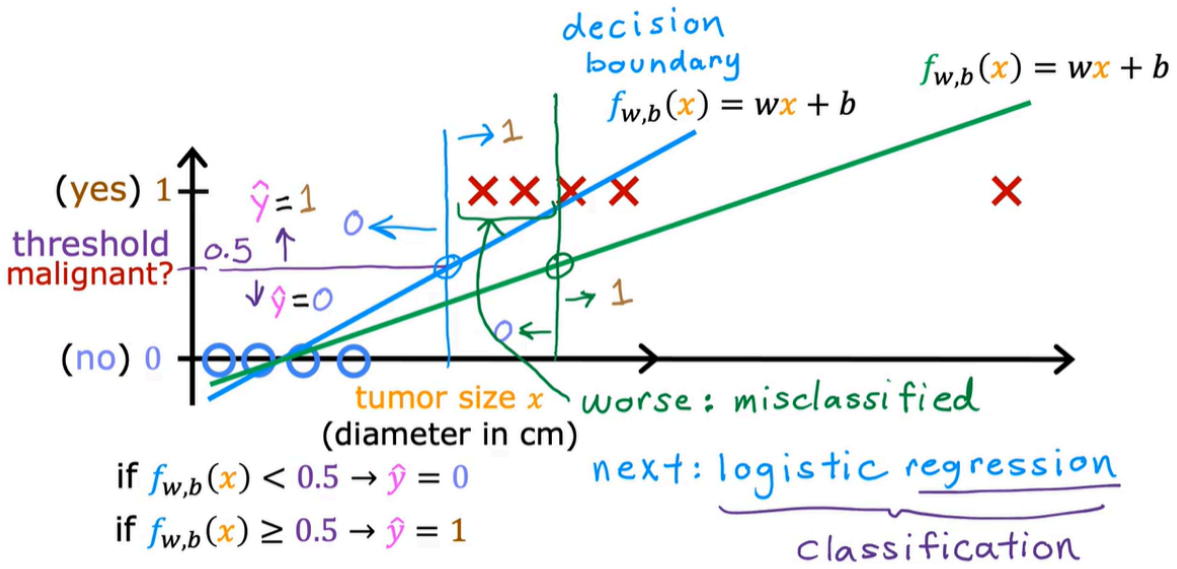


- Which is one of the most popular and most widely used learning algorithms today. Here are some examples of classification problems recall the example of trying to figure out whether an email is spam. So the answer you want to output is going to be either a no or a yes.
- Another example would be figuring out if an online financial transaction is fraudulent. Fighting online financial fraud is something I once worked on and it was strangely exhilarating. Because I knew there were forces out there trying to steal money and my team's job was to stop them. So the problem is given a financial transaction.

- Can your learning algorithm figure out is this transaction fraudulent, such as what this credit card stolen? Another example we've touched on before was trying to classify a tumor as malignant versus not.

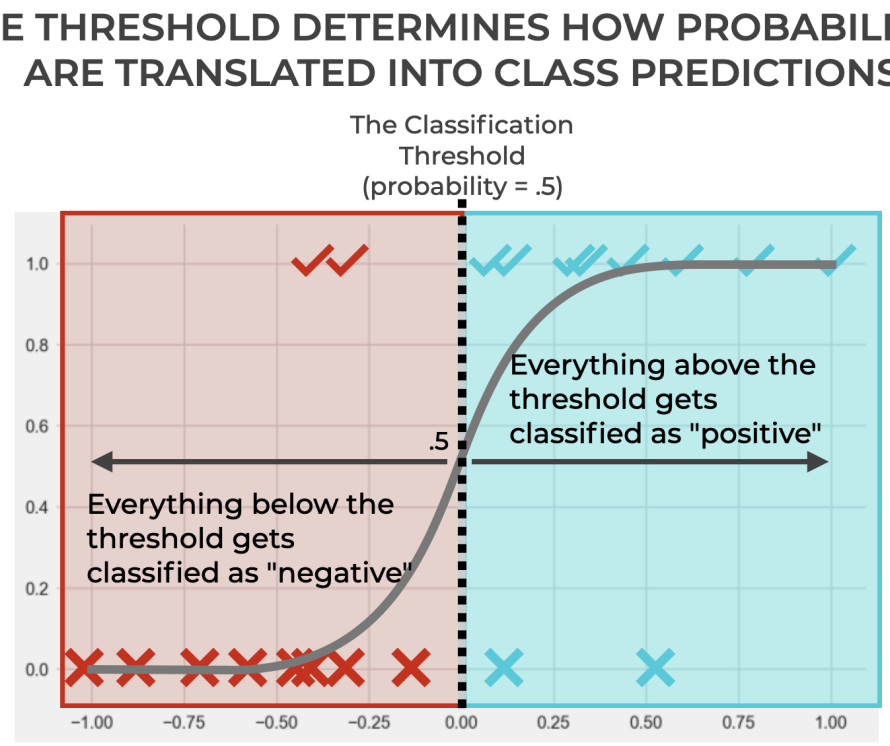
How to build a classification algorithm?

- Tumor: Khối u
- Malignant: ác tính



- The blue line is linear regression model trained on dataset without the most right point (red X)
- The green line is linear regression model trained on dataset with the most right point
- We could clearly see that the different decision boundaries of 2 models that could lead to misclassified
- To solve this: use logistic regression

Threshold value

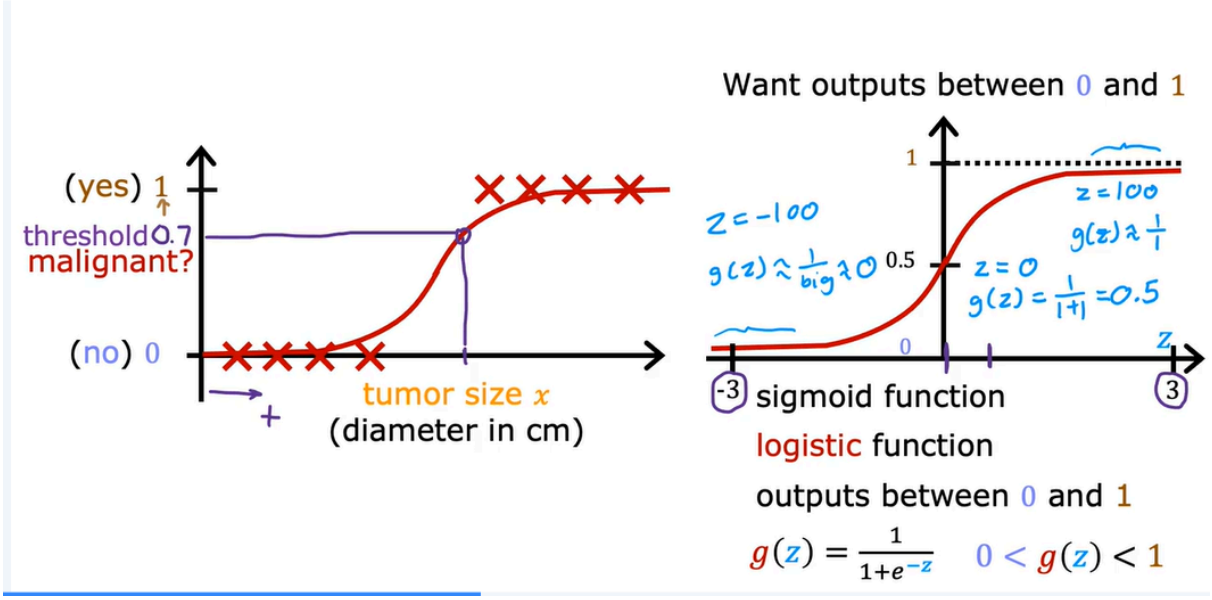


- In machine learning, a threshold value (or classification threshold, decision threshold or cutoff) is **a point that determines how a model's output is interpreted to make a prediction**. It's used to convert probability scores or other model outputs into binary (yes/no, true/false) or categorical classifications.
- **Default threshold:**

  A common default threshold is 0.5, but it's often necessary to adjust it based on the specific task and the desired balance between precision and recall.
- **Adjusting the threshold:**The choice of threshold can impact the trade-off between different metrics like precision and recall. A higher threshold generally increases precision (fewer false positives) but may decrease recall (more false negatives), while a lower threshold increases recall but may decrease precision.
- **Example:** In a spam filter, a model might predict the probability of an email being spam. A threshold of 0.5 would mean that if the predicted probability is greater than 0.5, the email is classified as spam, and if it's less than 0.5, it's classified as not spam.

- **How it works:** Models typically provide probability scores or other numerical outputs that represent the likelihood of an input belonging to a certain class. A threshold value is then used to classify the input. If the output score is above the threshold, it's assigned to one class; otherwise, it's assigned to the other.
- **Purpose:**Thresholds are crucial in classification tasks, where models need to predict the class (e.g., spam/not spam, positive/negative) for a given input.

Logistic Regression



- Let's talk about logistic regression, which is probably the single most widely used classification algorithm in the world. This is something that I use all the time in my work.
- Let's continue with the example of classifying whether a tumor is malignant. Whereas before we're going to use the label 1 or yes to the positive class to represent malignant tumors, and zero or no and negative examples to represent benign tumors. Here's a graph of the dataset (the left one) where the horizontal axis is the tumor size and the vertical axis takes on only values of 0 and 1, because is a classification problem.
- You saw in the last video that linear regression is not a good algorithm for this problem. In contrast, what logistic regression we end up doing is fit a

curve that looks like this, S-shaped curve to this dataset (the red curve in the left graph.

- For this example, if a patient comes in with a tumor of this size, which I'm showing on the x-axis, then the algorithm will output 0.7 suggesting that is closer or maybe more likely to be malignant and benign. Will say more later what 0.7 actually means in this context. But the output label y is never 0.7 is only ever 0 or 1.
- To build out to the logistic regression algorithm, there's an important mathematical function I like to describe which is called the Sigmoid function, sometimes also referred to as the logistic function. The Sigmoid function looks like this.
- Notice that the x-axis of the graph on the left and right are different. In the graph to the left on the x-axis is the tumor size, so is all positive numbers. Whereas in the graph on the right, you have 0 down here, and the horizontal axis takes on both negative and positive values and have label the horizontal axis Z. I'm showing here just a range of negative 3 to plus 3. **So the Sigmoid function outputs value is between 0 and 1**.
- If I use $g(z)$ to denote this function, then the formula:

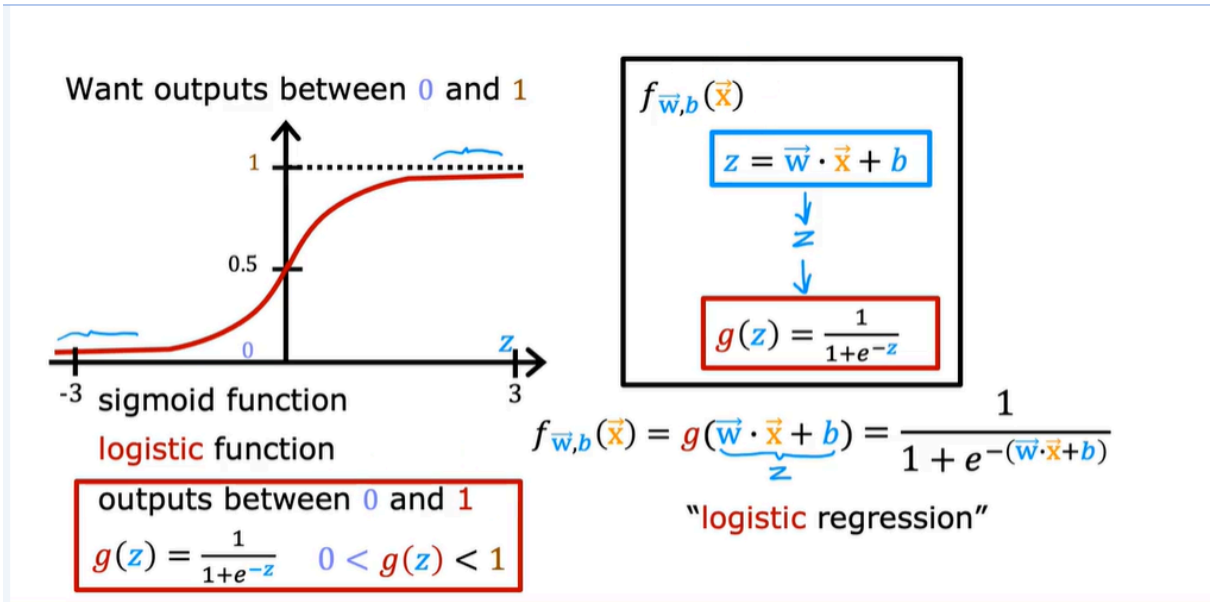
$$g(z) = \frac{1}{1 + e^{-z}}$$

- Where here e is a mathematical constant that takes on a value of about 2.7, and so e^{-z} is that mathematical constant to the power of -z.
    - Notice if z where really be, say a 100, e to the negative z is e to the negative 100 which is a tiny number. So this ends up being 1 over 1 plus a tiny little number, and so the denominator will be basically very close to 1. Which is why when z is large, $g(z)$ that is a Sigmoid function of z is going to be very close to 1.
    - Conversely, you can also check for yourself that when z is a very large negative number, then $g(z)$ becomes 1 / (a giant number), which is why $g(z)$ is very close to 0. That's why the sigmoid function has this shape where it starts very close to zero and slowly builds up or grows to the value of one. Also, in the Sigmoid function when z is equal to 0, then e to the negative z is e to the negative 0 which is equal to 1, and so $g(z)$

is equal to 1 over 1 plus 1 which is 0.5, so that's why it passes the vertical axis at 0.5.

Build up a logistic regression algorithm

Now, let's use this to build up to the logistic regression algorithm. We're going to do this in two steps.



Interpretation of logistic regression output

$f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-(\vec{w} \cdot \vec{x}+b)}}$

"probability" that class is 1

Example:
x is "tumor size"
y is 0 (not malignant)
or 1 (malignant)

$f_{\vec{w},b}(\vec{x}) = 0.7$
70% chance that y is 1

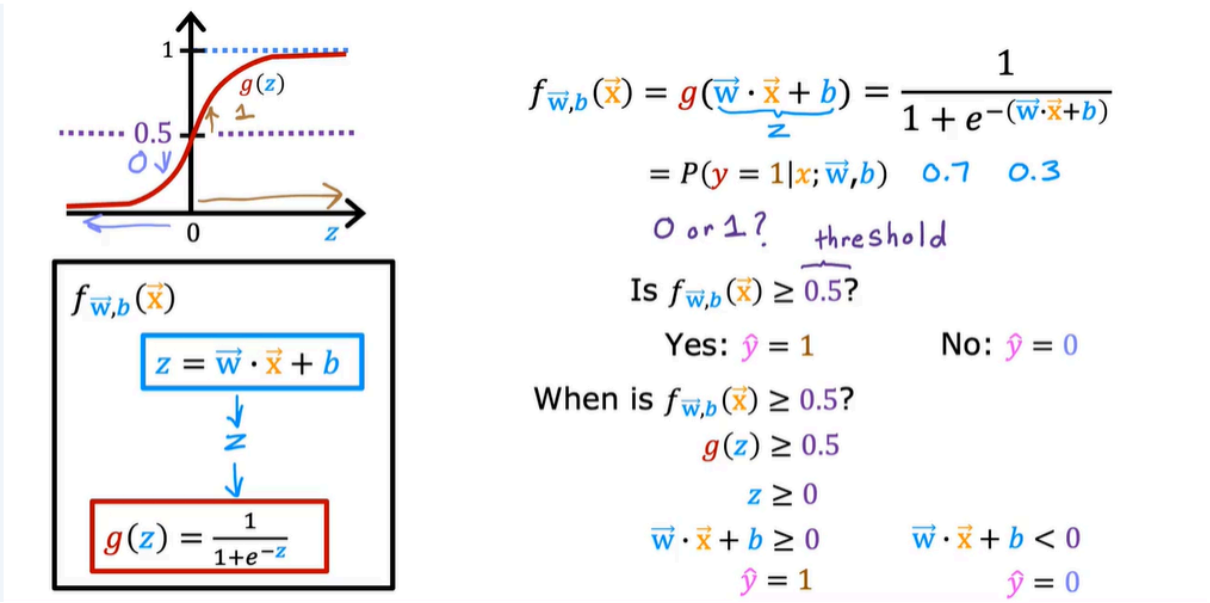
$P(1)$
$f_{\vec{w},b}(\vec{x}) = P(y = 1|\vec{x}; \vec{w}, b)$

Probability that y is 1,
given input $\vec{x}$, parameters $\vec{w}, b$

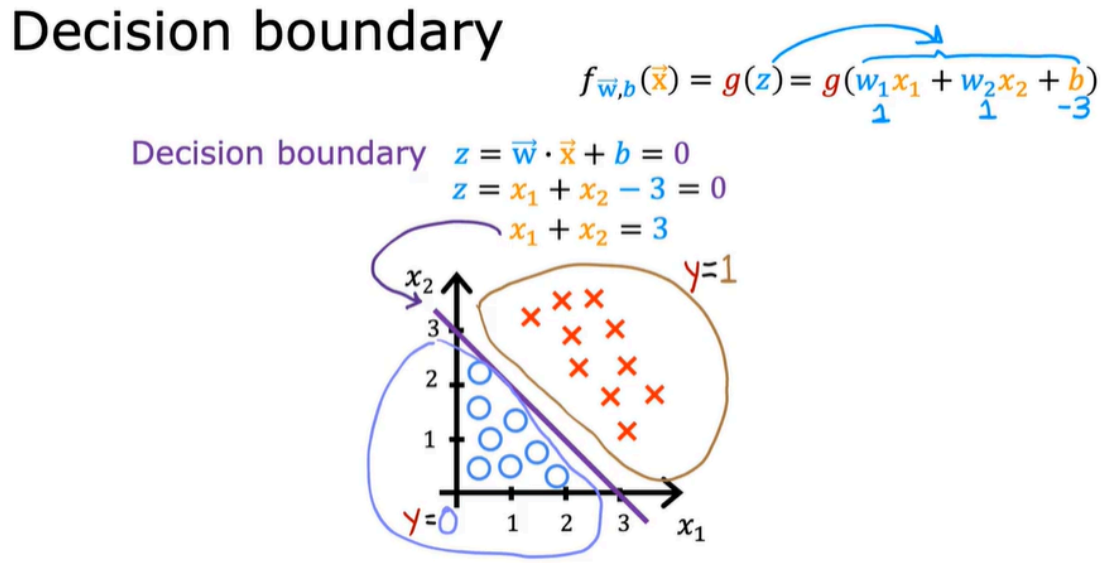
$P(y = 0) + P(y = 1) = 1$

Decision boundary

Recall:



Decision boundary

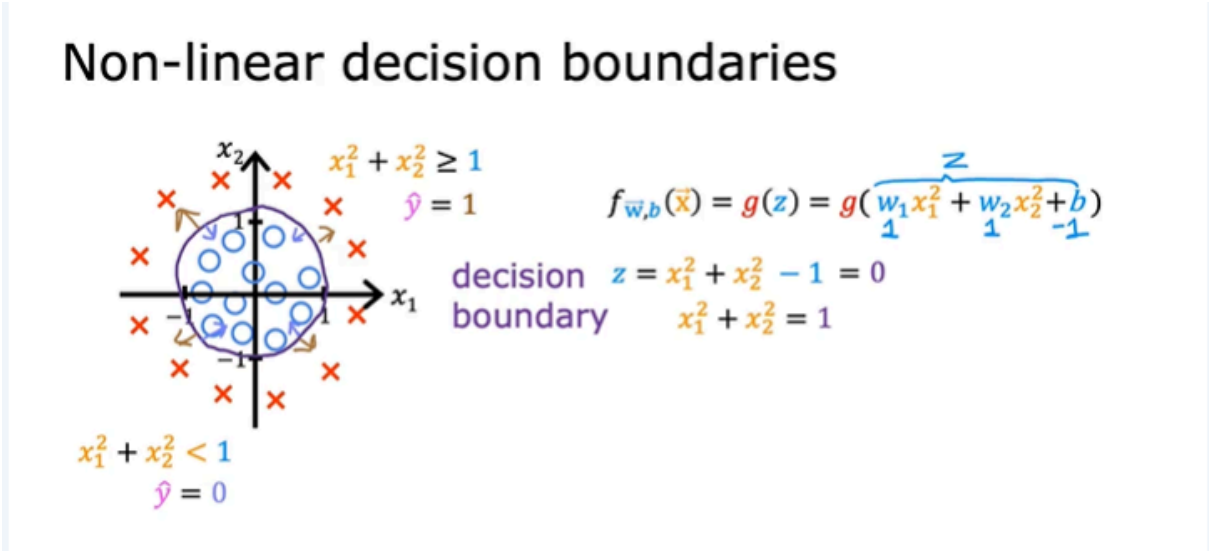


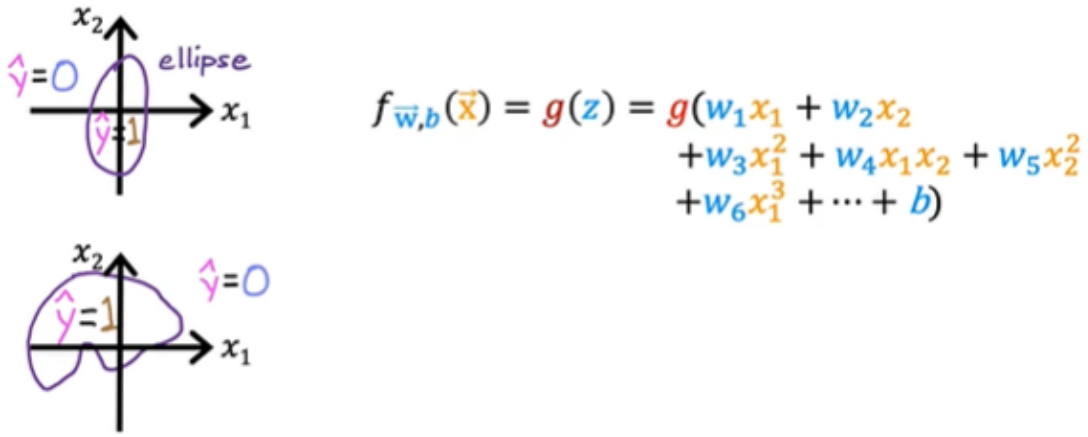
Decision boundary of this data set is the line: $x_1 + x_2 = 3$

And we have the Decision boundary when:

$$z = \vec{w} \cdot \vec{x} + b = 0$$

Non-linear decision boundaries





You can even get more complex decision boundaries, which can look like functions that maybe looks like that. So this is an example of an even more complex decision boundary than the ones we've seen previously.

- This implementation of logistic regression will predict y equals 1 inside this shape and outside the shape will predict y equals 0. With these polynomial features, you can get very complex decision boundaries.
- In other words, logistic regression can learn to fit pretty complex data. Although if you were to not include any of these higher-order polynomials, so if the only features you use are x_1, x_2, x_3, and so on, then the decision boundary for logistic regression will always be linear, will always be a straight line.