Classification with logistic regression

Motivation

Classification Overview

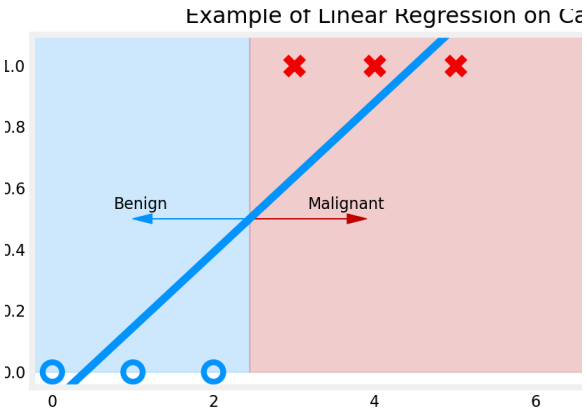
- Classification predicts discrete outcomes, where the output variable can take on a limited number of values (e.g., yes/no, spam/not spam).
- Linear regression is not suitable for classification tasks because it predicts continuous values rather than discrete categories.

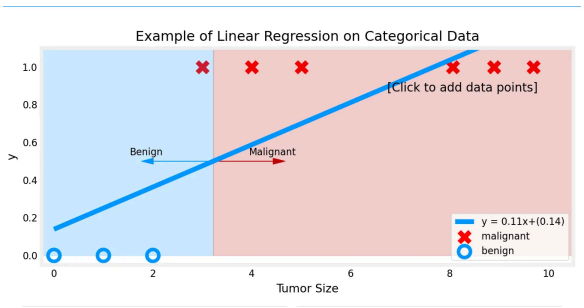
Binary Classification

- Binary classification involves two possible outputs, often represented as 0 (negative class) and 1 (positive class).
- Common examples include spam detection in emails and identifying fraudulent financial transactions.

Limitations of Linear Regression

- Using linear regression for classification can lead to inaccurate predictions, especially when new data points shift the decision boundary.
- The decision boundary is the threshold that separates the two classes, and linear regression can misclassify data if it shifts due to new examples.





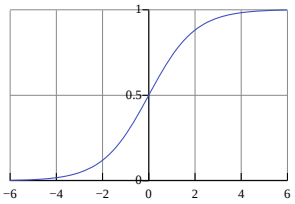
Logistic Regression

Logistic regression is a **classification algorithm** designed for binary classification problems. It predicts the probability of an instance belonging to the positive class.

Model

Sigmoid Function

The **sigmoid function** is the main function in logistic regression. It maps input values (which can be both + and - after normalization around zero) to outputs between **0 and 1**, forming an **S-shaped curve**. This makes it ideal for **classification tasks**, as it represents the **probability** that an input belongs to a certain class. A **threshold** (typically 0.5) is used to decide whether the output should be classified as **0 or 1**.



source: https://en.wikipedia.org/wiki/Sigr

Formula:

$$g(z) = \frac{1}{1 + e^{-z}} = 1 - P(-z)$$
$$0 < g(z) < 1$$

or probability can also be represented as:

$$g(z) = log_e(\frac{P}{1 - P})$$

- Takes a linear combination $z = \vec{w} \cdot \vec{x} + b$ and maps it to a value between 0 and 1, representing probabilities.
- Behavior:
  - Large positive $z \rightarrow g(z) \approx 1$
  - Large negative $z \rightarrow g(z) \approx 0$
  - $z = 0 \rightarrow g(z) = 0.5$

Mathematical Form:

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

- $\vec{w}$: Weights (parameters learned from data)
- b: Bias (intercept term)
- $\vec{x}$: Input features (e.g., tumor size, shape)

Interpretation

- The output $f(\vec{x})$ is the **probability** $P(y = 1 \mid \vec{x}; \vec{w}, b)$, where:
  - $y = 1$: Positive class (e.g., *malignant* tumor)
  - $y = 0$: Negative class (e.g., *benign* tumor)
  - Given input $\vec{x}$, parameters $\vec{w}, b$
- If the model predicts a $f_{\vec{w},b}(\vec{x}) = 0.7$ probability, it suggests a 70% chance of malignancy, leaving a 30% chance for benignity.
- We always have:

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

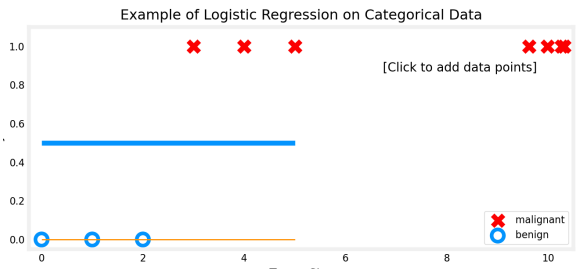
Example

Suppose we classify tumors based on size ($\vec{x} = \text{tumor size in cm}$):

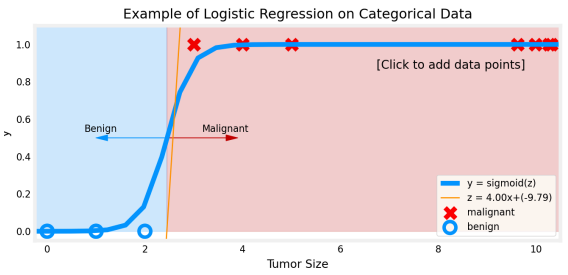
- After training, the model might learn $\vec{w} = 2$ and $b = -4$.
- For a tumor of size 2 cm:
  - $z = (2 \cdot 2) + (-4) = 0$
  - $f(\vec{x}) = g(0) = \frac{1}{1+e^0} = 0.5$
  - Probability of malignancy $= 50\%$
- For a tumor of size 3 cm:
  - $z = (2 \cdot 3) + (-4) = 2$
  - $f(\vec{x}) = g(2) = \frac{1}{1+e^{-2}} \approx 0.88$
  - Probability of malignancy $\approx 88\%$

Let's apply logistic regression to the categorical data example of tumor classification.

For many data points that far from initial points (X point)



The Logistic Regression works well to toggle 0.5 threshold (after regression)



Note

Unlike linear regression, which predicts **continuous** values, logistic regression outputs probabilities between 0 and 1, making it suitable for classification.

The Decision Boundary

Introduction

In logistic regression, the **decision boundary** is the **threshold** that separates data points into two classes (e.g., positive class $y = 1$ or negative class $y = 0$). It represents the set of points where the model's predicted probability is exactly 0.5, indicating an equal likelihood of belonging to either class.

$$g(z) \geq \text{threshold}$$

Defining the Decision Boundary

Logistic regression computes predictions in two steps:

1. Calculate the linear combination $z = \vec{w} \cdot \vec{x} + b$, where $\vec{w}$ is the weight vector, $\vec{x}$ is the input feature vector, and b is the bias.
2. Apply the sigmoid function $g(z) = \frac{1}{1+e^{-z}}$, which outputs a probability between 0 and 1.

The decision boundary occurs where $g(z) = 0.5$, which corresponds to $z = 0$. Thus, the boundary is defined by:

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

Next, the decision itself is represented as $\hat{y}$. It can be more formally written as:

$$\begin{gathered}\text{Is } f_{\vec{w},b}(\vec{x}) \geq \text{threshold?} \\ \text{Yes: } \hat{y} = 1 \\ \text{No: } \hat{y} = 0\end{gathered}$$

So, the next question is:

$$\begin{gathered}\text{When is } f_{\vec{w},b}(\vec{x}) \geq \text{threshold} (= 0.5) \\ g(z) \geq \text{threshold} \\ z \geq 0\end{gathered}$$

- If $z \geq 0$, the predicted output $\hat{y} = 1$ (positive class).
- If $z < 0$, the predicted output $\hat{y} = 0$ (negative class).

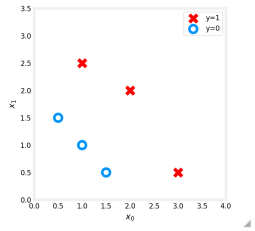
When $z = 0$, it lies exactly on the decision boundary, but in binary classification, it must still be assigned to either the positive or negative class. This rule also applies to non-linear (e.g., polynomial) decision functions—it's simply about deciding how to handle the boundary point.

Example:

source: https://www.coursera.org/learn/machine-learning/ungradedLab/ULi82/optional-lab-decision-boundary/lab?path=%2Fnotebooks%2FC1_W3_Lab03_Decision_Boundary_Soln.ipynb

Dataset

$X = \{x^{(i)}\} = \{(0.5, 1.5), (1, 1), (1.5, 0.5), (3, 0.5), (2, 2), (1, 2.5)\}$
$y = \{0, 0, 0, 1, 1, 1\}$



The plotting for above dataset.

source: *w3_lab03*

To train a logistic regression model:

$$f(x) = g(w_0 x_0 + w_1 x_1 + b)$$

where:

- $g(z) = \frac{1}{1+e^{-z}}$
- $b = -3, w_0 = 1, w_1 = 1$

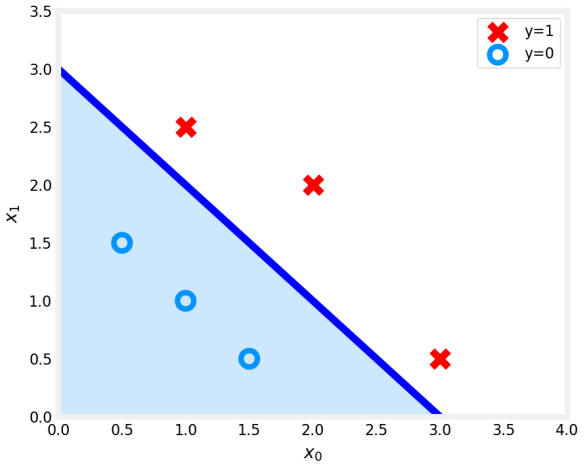
$$\Rightarrow f(x) = g(x_0 + x_1 - 3)$$

Plotting decision boundary

Now, let's go back to our example to understand how the logistic regression model is making predictions.

- Our logistic regression model has the form: $f(\mathbf{x}) = g(-3 + x_0 + x_1)$
- From what you've learnt above, you can see that this model predicts $y = 1$ if $-3 + x_0 + x_1 >= 0$

Let's see what this looks like graphically. We'll start by plotting $-3 + x_0 + x_1 = 0$, which is equivalent to $x_1 = 3 - x_0$.



- In the plot above, the blue line represents the line $x_0 + x_1 - 3 = 0$ and it should intersect the x_1 axis at 3 (if we set $x_1 = 3, x_0 = 0$) and the x_0 axis at 3 (if we set $x_1 = 0, x_0 = 3$).

- The shaded region represents $-3 + x_0 + x_1 < 0$.
- The region above the line is $-3 + x_0 + x_1 > 0$.
- Any point in the shaded region (under the line) is classified as $y = 0$. Any point on or above the line is classified as $y = 1$. This line is known as the "decision boundary".

As we've seen in the lectures, by using higher order polynomial terms (eg: $f(x) = g(x_0^2 + x_1 - 1)$, we can come up with more complex non-linear boundaries.

Refresher on logistic regression and decision boundary

- Recall that for logistic regression, the model is represented as: $f_{w,b}(x^{(i)}) = g(w \cdot x^{(i)} + b)$ where $g(z)$ is known as the sigmoid function and it maps all input values to values between 0 and 1:

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

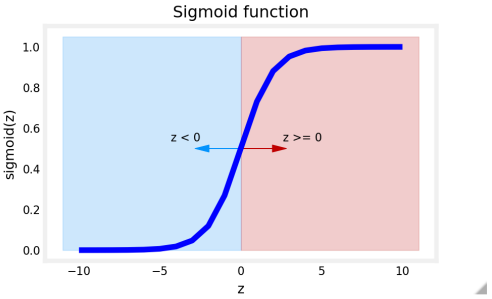
and $\mathbf{w} \cdot \mathbf{x}$ is the vector dot product: $\mathbf{w} \cdot \mathbf{x} = w_0 x_0 + w_1 x_1$

- We interpret the output of the model $(f_{\mathbf{w},b}(x))$ as the probability that $y = 1$ given $\mathbf{x}$ and parameterized by $\mathbf{w}$ and b.
  - Therefore, to get a final prediction ($y = 0$ or $y = 1$) from the logistic regression model, we can use the following heuristic -

    if $f_{\mathbf{w},b}(x) >= 0.5$, predict $\hat{y} = 1$

    if $f_{\mathbf{w},b}(x) < 0.5$, predict $\hat{y} = 0$

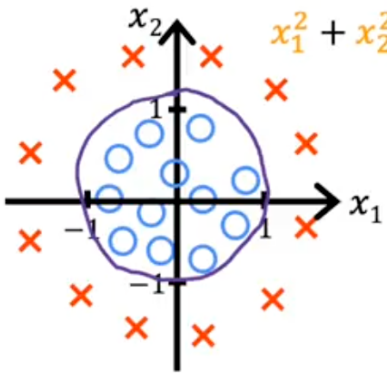
- Let's plot the sigmoid function to see where $g(z) >= 0.5$



Linear Decision Boundary: With two features (e.g., x_1 and x_2), the equation $\vec{w} \cdot \vec{x} + b = 0$ forms a straight line in the feature space. This line divides the plane into regions where the model predicts $y = 1$ or $y = 0$.

Complex Decision Boundaries

By adding polynomial features (e.g., x_1^2, x_1x_2, x_2^2), z becomes a polynomial function. This allows the decision boundary to take on non-linear shapes, such as curves, circles, or ellipses, enabling the model to capture more complex data patterns.



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(\overbrace{\underbrace{w_1}_{1}x_1^2 + \underbrace{w_2}_{1}x_2^2 + \underbrace{b}_{1}}^{z})$$

$$\Rightarrow z = x_1^2 + x_2^2 - 1 = 0$$
$$\Rightarrow x_1^2 + x_2^2 = 1 \text{ (a circle)}$$

Then:

- $x_1^2 + x_2^2 \geq 1 \Rightarrow \hat{y} = 0$
- $x_1^2 + x_2^2 < 1 \Rightarrow \hat{y} = 1$

Adjusting the Threshold

While 0.5 is the standard threshold, it can be adjusted based on the application. For instance, in a tumor detection scenario, a lower threshold (e.g., 0.3) might be used to prioritize detecting more potential tumors, reducing false negatives at the cost of more false positives. The document, however, emphasizes the default threshold of 0.5 for simplicity.

💡 **QUES:** Let's say you are creating a tumor detection algorithm. Your algorithm will be used to flag potential tumors for future inspection by a specialist. What value should you use for a threshold?

- High, say a threshold of 0.9?
- Low, say a threshold of 0.2? 👍🏻

⇒ You would not want to **miss a potential tumor,** so you will want a **low threshold**. A specialist will review the output of the algorithm which reduces the possibility of a 'false positive'. The key point of this question is to note that the threshold value **does not** need to be 0.5.