Machine Learning
ICT-4261

By-

Dr. Jesmin Akhter
Professor
Institute of Information Technology
Jahangirnagar University

Contents

The course will mainly cover the following topics:

- ✔ A Gentle Introduction to Machine Learning
- ✔ Linear Regression
- ✔ Logistic Regression
- ✔ Naive Bayes
- ✔ Support Vector Machines
- ✔ Decision Trees and Ensemble Learning
- ✔ Clustering Fundamentals
- ✔ Hierarchical Clustering
- ✔ Neural Networks and Deep Learning
- ✔ Unsupervised Learning

Outline

- ✔ Logistic Regression
  - – Logistic regression
  - – Sigmoid Function
  - – Logistic Regression Equation and Assumptions
  - – Loss Function
  - – Derivation of Cost Function

Logistic regression

- ✔ Logistic regression is a supervised machine learning algorithm that is commonly used in binary classification problems where the outcome variable reveals either of the two categories: yes/no, 0/1, or true/false.
- ✔ Logical regression analyzes the relationship between one or more independent variables and classifies data into discrete classes. It is extensively used in predictive modeling, where the model estimates the mathematical probability of whether an instance belongs to a specific category or not.
  - – For example,
    - 0 – represents a negative class;
    - 1 – represents a positive class.
- ✔ Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
- ✔ So despite its name, it's a classification algorithm

Sigmoid Function

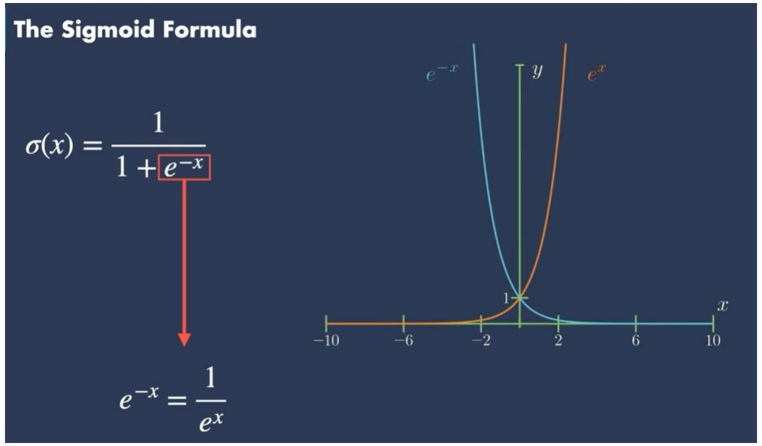
✔ The sigmoid function is used to map the predicted values to probabilities.

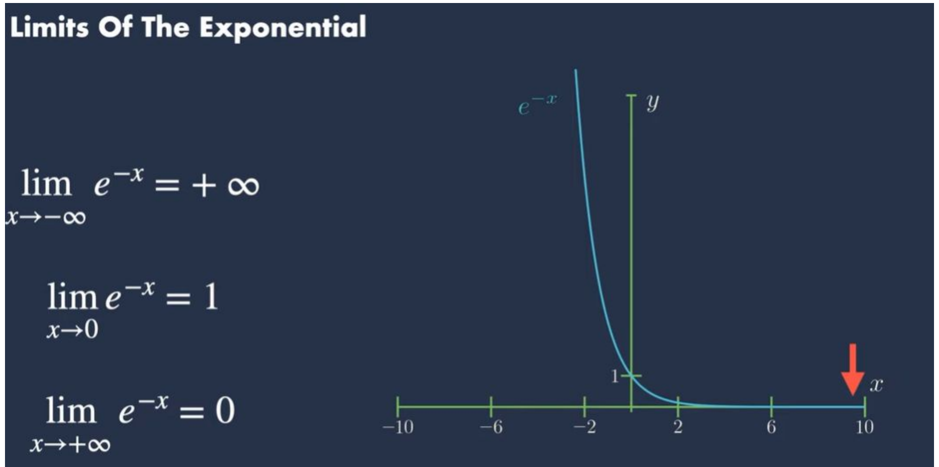
Sigmoid function: $\sigma(\boldsymbol{x}) = \frac{\mathbf{1}}{\mathbf{1}+\boldsymbol{e}^{-\boldsymbol{x}}}$

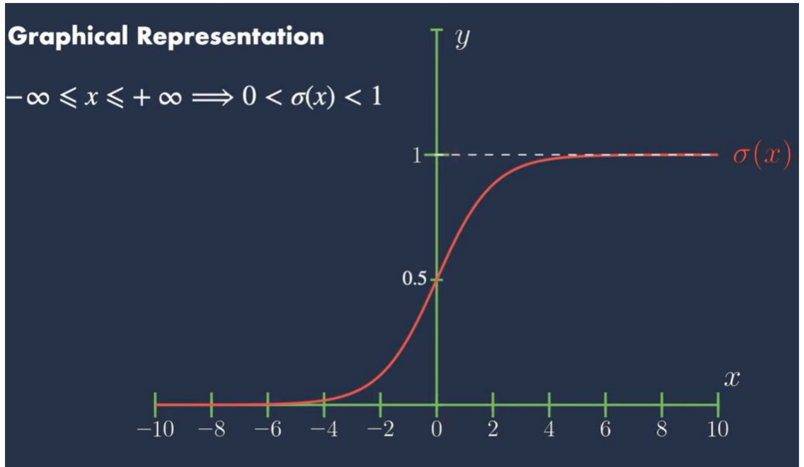
✓ The sigmoid function takes any real number as input and outputs a value between 0 and 1. It approaches 0 as the input becomes negative and 1 as the input becomes positive. When the input is 0, the sigmoid function returns 0.5.

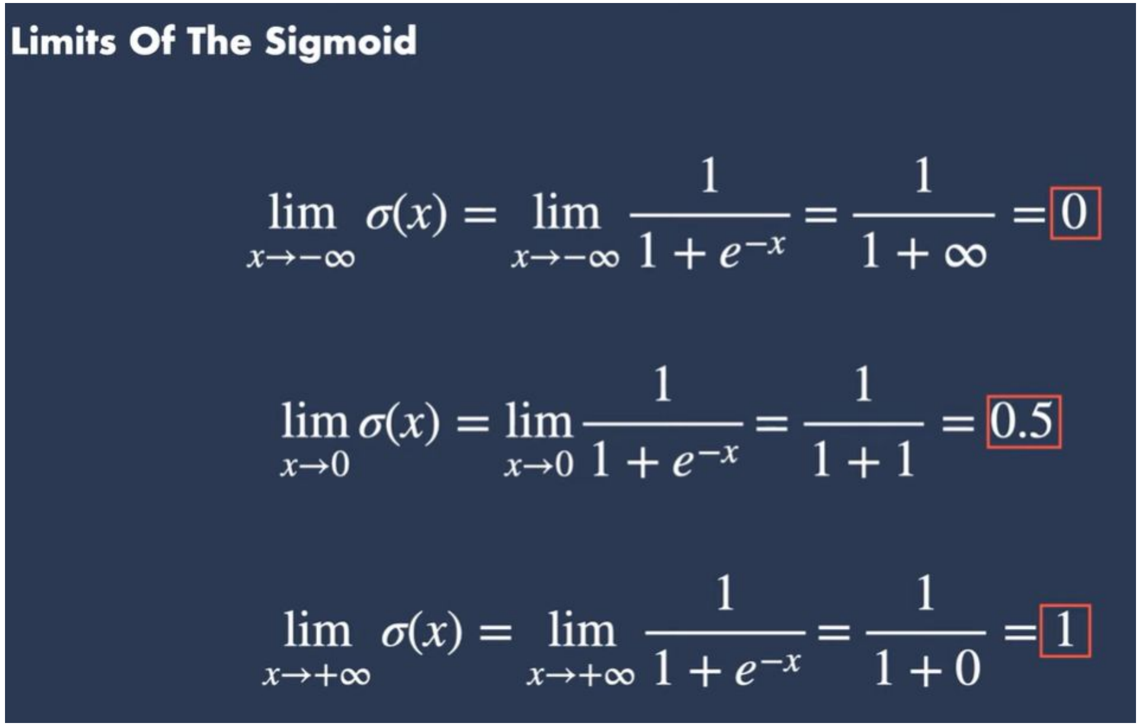
✓ The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.

✓ In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.







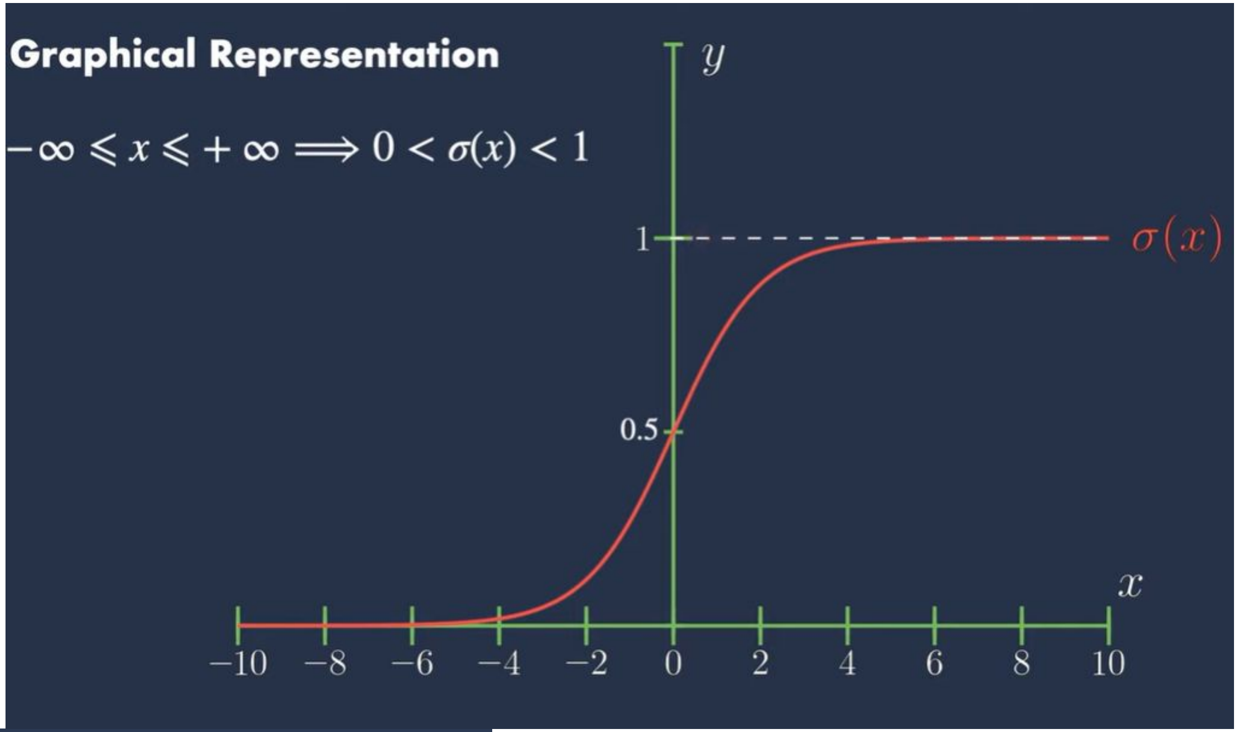


Limits Of The Sigmoid

$$\lim_{x\to-\infty} \sigma(x) = \lim_{x\to-\infty} \frac{1}{1+e^{-x}} = \frac{1}{1+\infty} = \boxed{0}$$

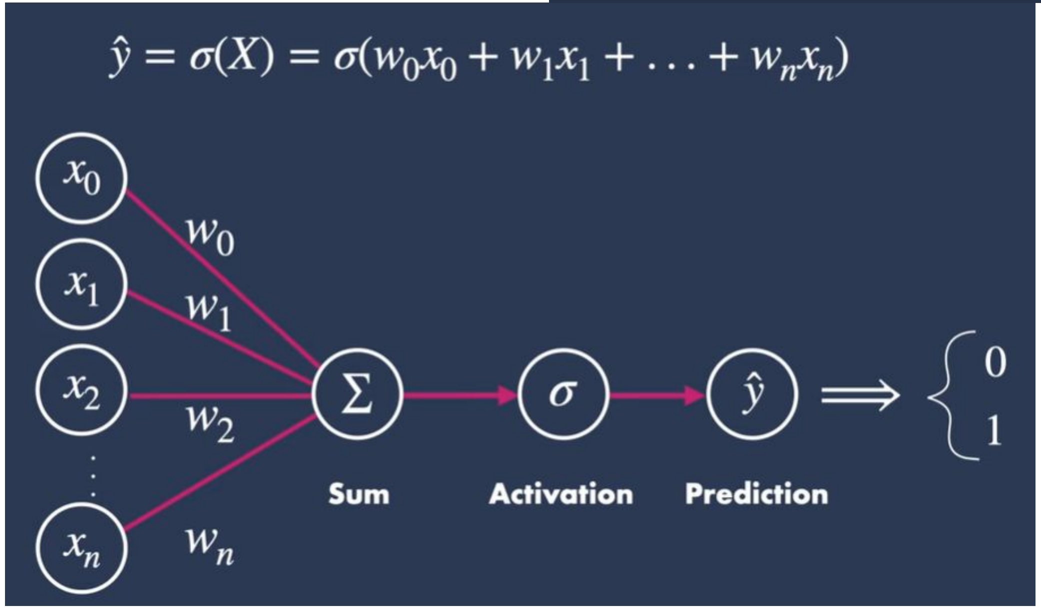
$$\lim_{x\to 0} \sigma(x) = \lim_{x\to 0} \frac{1}{1+e^{-x}} = \frac{1}{1+1} = \boxed{0.5}$$

$$\lim_{x\to+\infty} \sigma(x) = \lim_{x\to+\infty} \frac{1}{1+e^{-x}} = \frac{1}{1+0} = \boxed{1}$$



Graphical Representation

$$-\infty \leqslant x \leqslant +\infty \Longrightarrow 0 < \sigma(x) < 1$$



$$\hat{y} = \sigma(X) = \sigma(w_0x_0 + w_1x_1 + \ldots + w_nx_n)$$

Logistic Regression Equation and Assumptions

✔ How logistic regression squeezes the output of linear regression between 0 and 1?

✔ Let’s start by mentioning the formula of logistic function:

$$P = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

✔ The equation of the best fit line in linear regression is:

$$y = \beta_0 + \beta_1 x$$

✔ Let’s say instead of y we are taking probabilities (P). $P = \beta_0 + \beta_1 x$

✔ But there is an issue here, the value of (P) will exceed 1 or go below 0 and we know that range of Probability is (0-1). To overcome this issue we take ***“odds”*** of P:

$$\frac{P}{1 - P} = \beta_0 + \beta_1 x$$

✔ Odds can always be positive which means the range will always be (0,+∞). Odds are nothing but the ratio of the probability of success and probability of failure. Now the question comes out of so many other options to transform this why did we only take ***‘odds’***? Because odds are probably the easiest way to do this.

Logistic Regression Equation and Assumptions

- ✔ The problem here is that the range is restricted and we don’t want a restricted range because if we do so then our correlation will decrease. By restricting the range we are actually decreasing the number of data points and of course, if we decrease our data points, our correlation will decrease. It is difficult to model a variable that has a restricted range. To control this we take the ***log of odds*** which has a range from (-∞,+∞).

$$\log\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 x$$

- ✔ Now we just want a function of P because we want to predict probability, not log of odds. To do so we will multiply by ***exponent*** on both sides and then solve for P.

$$\exp[\log(\frac{p}{1-p})] = \exp(\beta_0 + \beta_1 x)$$

$$e^{\ln[\frac{p}{1-p}]} = e^{(\beta_0 + \beta_1 x)}$$

$$\frac{p}{1-p} = e^{(\beta_0 + \beta_1 x)}$$

$$p = e^{(\beta_0 + \beta_1 x)} - pe^{(\beta_0 + \beta_1 x)}$$

$$p = p[\frac{e^{(\beta_0 + \beta_1 x)}}{p} - e^{(\beta_0 + \beta_1 x)}]$$

$$1 = \frac{e^{(\beta_0 + \beta_1 x)}}{p} - e^{(\beta_0 + \beta_1 x)}$$

$$p[1 + e^{(\beta_0 + \beta_1 x)}] = e^{(\beta_0 + \beta_1 x)}$$

$$p = \frac{e^{(\beta_0 + \beta_1 x)}}{1 + e^{(\beta_0 + \beta_1 x)}}$$

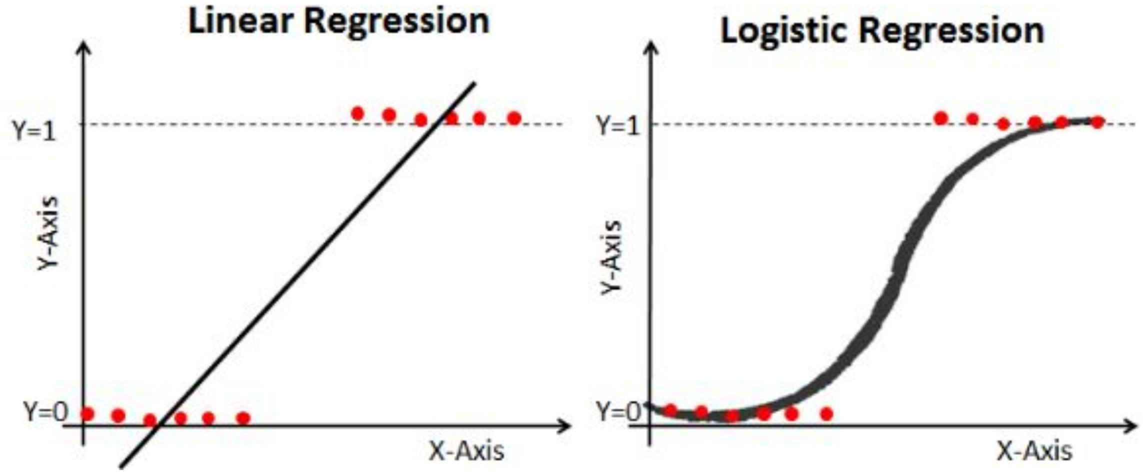
Logistic Regression Equation and Assumptions

Now dividing by $e^{\left(\beta_0+\beta_1 x\right)}$, *we will get*

$$p = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$$

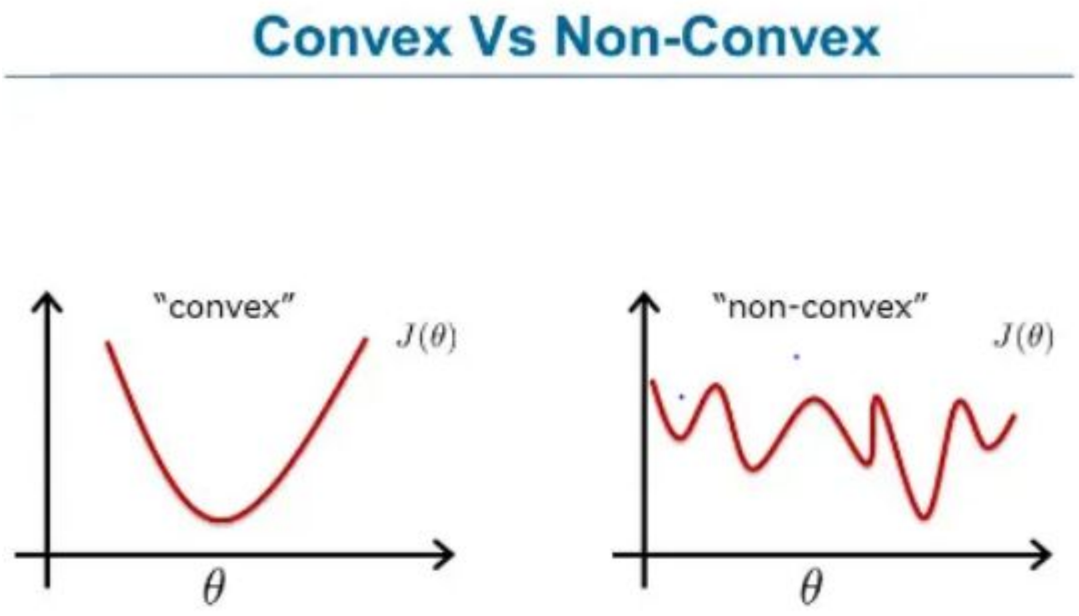
here,
x = input value
P = predicted output
⍰0 = bias or intercept term
⍰$_1$ = coefficient for input (x)

- ✔ The above equation represents logistic regression. This equation is similar to linear regression, where the input values are combined linearly to predict an output value using weights or coefficient values. However, unlike linear regression, the output value modeled here is a binary value (0 or 1) rather than a numeric value
- ✔ Thus Logistic regression squeezes a straight line into an S-curve with the help of a sigmoid function is as shown below.



Loss Function

✔ For Logistic Regression we can't use the same loss function as for Linear Regression because the Logistic Function (Sigmoid Function) will cause the output to be non-convex, which will cause many local optima.



Convex Vs Non-Convex

Loss Function

✔ Instead, we will use the following loss function for logistic regression:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\begin{aligned} &\mathrm{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) && \text{if y} = 1 \\ &\mathrm{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x)) && \text{if y} = 0 \end{aligned}$$

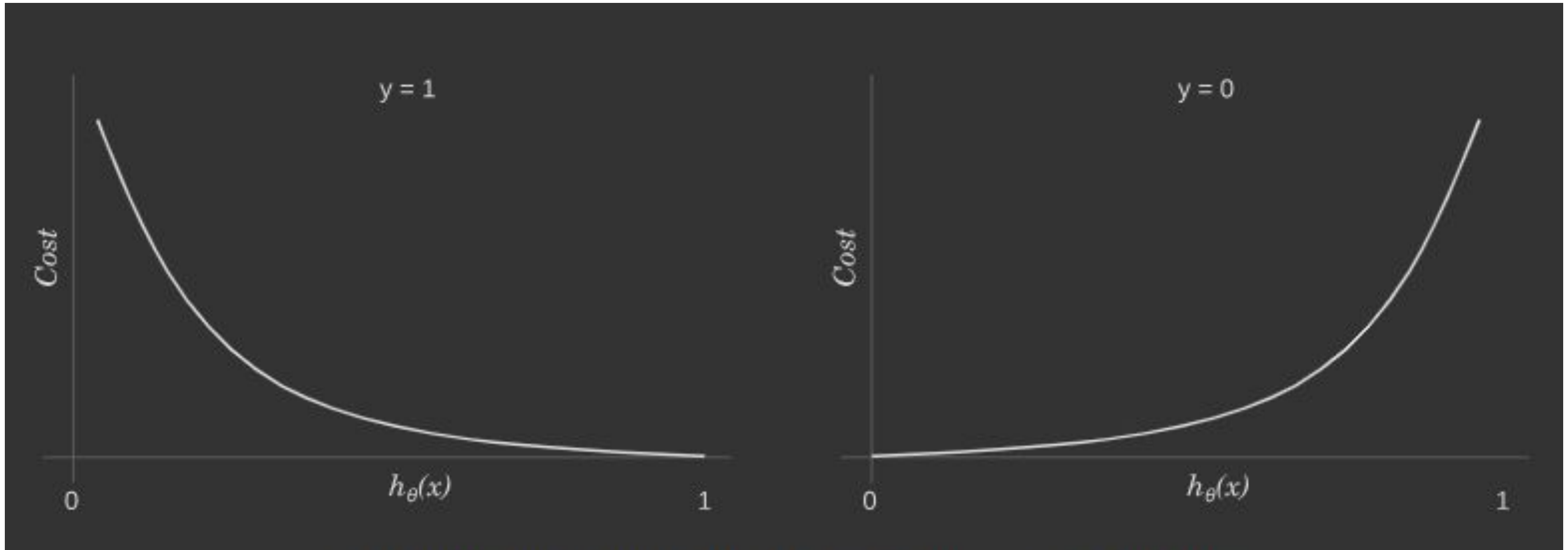
✔ To make it easier to work with the loss function we can compress the two conditional cases into one equation

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \ \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right]$$

✔ Notice that when y is equal to 1 the second term will be zero and therefore will not affect the loss.

✔ One the other hand if y is equal to 0 the first term will be zero and therefore will not affect the loss

Simplifying the Loss Function



2. How the cost function for logistic regression looks like.

In case $y=1$, the output (i.e. the cost) approaches to 0 as $h_\theta(x)$ approaches to 1. Conversely, the cost grows to infinity as $h_\theta(x)$ approaches to 0. You can clearly see it in the plot of the left side. If the label is $y=1$ but the algorithm predicts $h_\theta(x)=0$, the outcome is completely wrong.
Conversely, the same intuition applies when $y=0$, depicted in the plot of the right side. Bigger penalties when the label is $y=0$ but the algorithm predicts $h_\theta(x)=1$.

Derivation of Cost Function

✔ Before we derive our cost function we'll first find a derivative for our sigmoid function because it will be used in derivating the cost function.

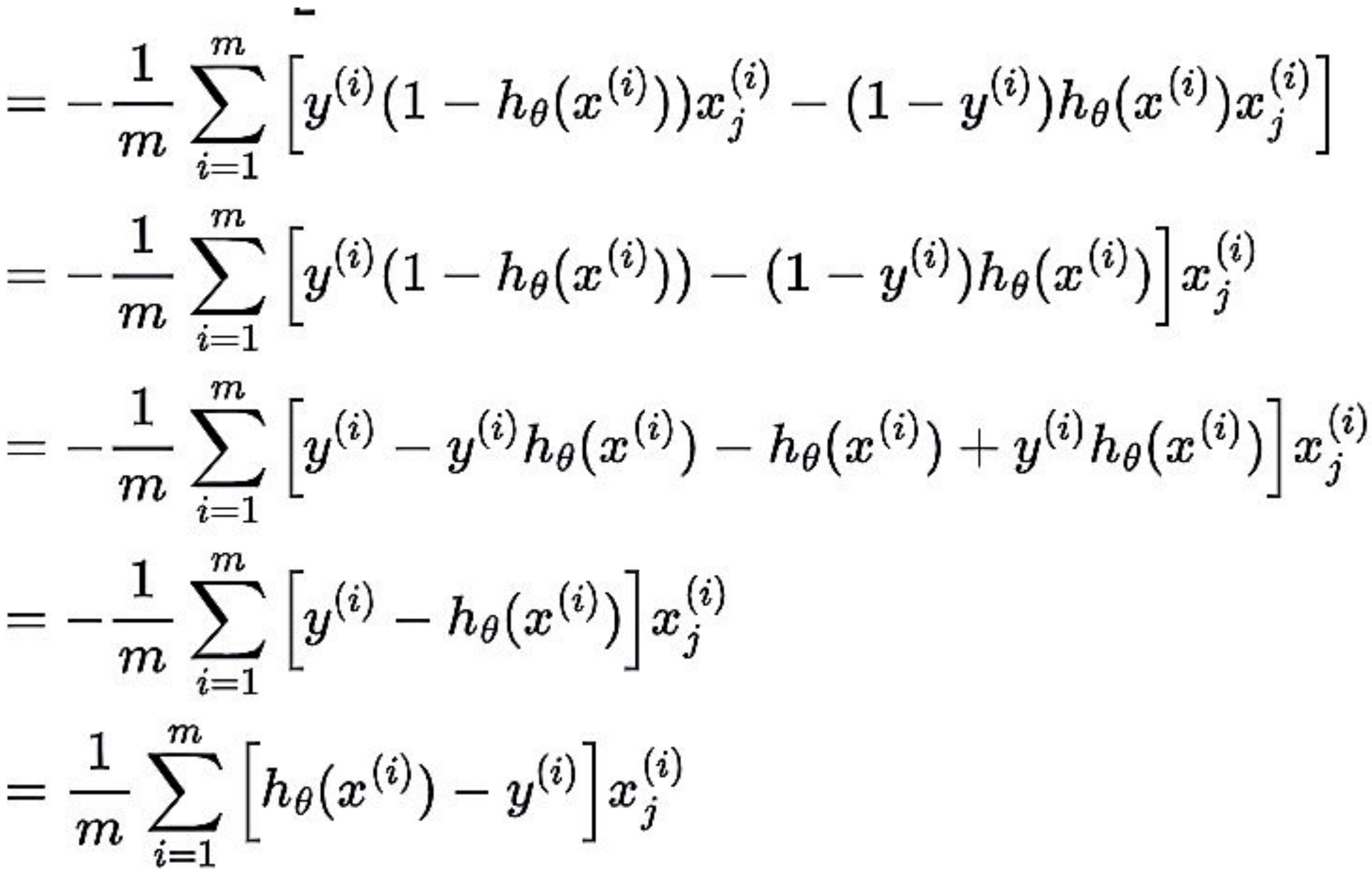
$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left[\frac{1}{1+\mathrm{e}^{-x}}\right] \\
&= \frac{d}{dx}\left(1+\mathrm{e}^{-x}\right)^{-1} \\
&= -(1+e^{-x})^{-2}(-e^{-x}) \\
&= \frac{e^{-x}}{(1+e^{-x})^{2}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{(1+e^{-x})-1}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \\
&= \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right) \\
&= \sigma(x) \cdot (1-\sigma(x))
\end{aligned}
$$

Deriving the Gradient Descent formula for Logistic Regression

✔ Now we are ready to find out the partial derivative:

$$
\begin{aligned}
\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{-1}{m}\sum_{i=1}^{m}\left[y^{(i)}log(h_\theta(x^{(i)}))+(1-y^{(i)})log(1-h_\theta(x^{(i)}))\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\frac{\partial}{\partial\theta_j}log(h_\theta(x^{(i)}))+(1-y^{(i)})\frac{\partial}{\partial\theta_j}log(1-h_\theta(x^{(i)}))\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[\frac{y^{(i)}\frac{\partial}{\partial\theta_j}h_\theta(x^{(i)})}{h_\theta(x^{(i)})}+\frac{(1-y^{(i)})\frac{\partial}{\partial\theta_j}(1-h_\theta(x^{(i)}))}{1-h_\theta(x^{(i)})}\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[\frac{y^{(i)}\frac{\partial}{\partial\theta_j}\sigma(\theta^T x^{(i)})}{h_\theta(x^{(i)})}+\frac{(1-y^{(i)})\frac{\partial}{\partial\theta_j}(1-\sigma(\theta^T x^{(i)}))}{1-h_\theta(x^{(i)})}\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[\frac{y^{(i)}\sigma(\theta^T x^{(i)})(1-\sigma(\theta^T x^{(i)}))\frac{\partial}{\partial\theta_j}\theta^T x^{(i)}}{h_\theta(x^{(i)})}+\frac{-(1-y^{(i)})\sigma(\theta^T x^{(i)})(1-\sigma(\theta^T x^{(i)}))\frac{\partial}{\partial\theta_j}\theta^T x^{(i)}}{1-h_\theta(x^{(i)})}\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[\frac{y^{(i)}h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))\frac{\partial}{\partial\theta_j}\theta^T x^{(i)}}{h_\theta(x^{(i)})}-\frac{(1-y^{(i)})h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))\frac{\partial}{\partial\theta_j}\theta^T x^{(i)}}{1-h_\theta(x^{(i)})}\right]
\end{aligned}
$$

Deriving the Gradient Descent formula for Logistic Regression



$$= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}(1-h_\theta(x^{(i)}))x_j^{(i)} - (1-y^{(i)})h_\theta(x^{(i)})x_j^{(i)}\right]$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}(1-h_\theta(x^{(i)})) - (1-y^{(i)})h_\theta(x^{(i)})\right]x_j^{(i)}$$

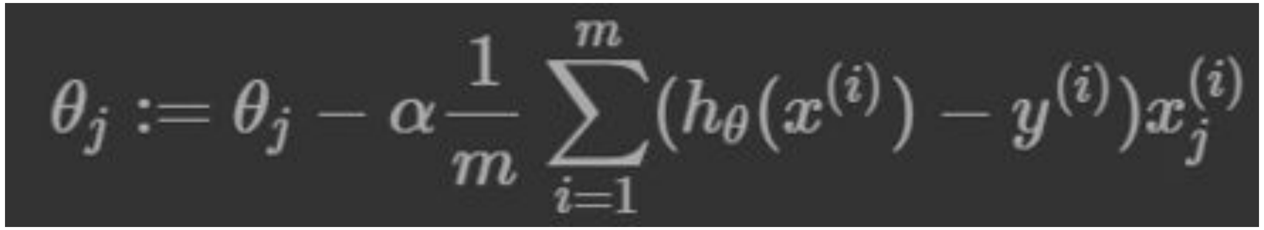
$$= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} - y^{(i)}h_\theta(x^{(i)}) - h_\theta(x^{(i)}) + y^{(i)}h_\theta(x^{(i)})\right]x_j^{(i)}$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} - h_\theta(x^{(i)})\right]x_j^{(i)}$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)}) - y^{(i)}\right]x_j^{(i)}$$

Derivation of Cost Function

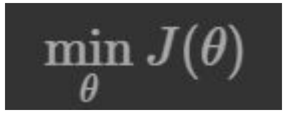
✔ Hence the derivative of our cost function is:



$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

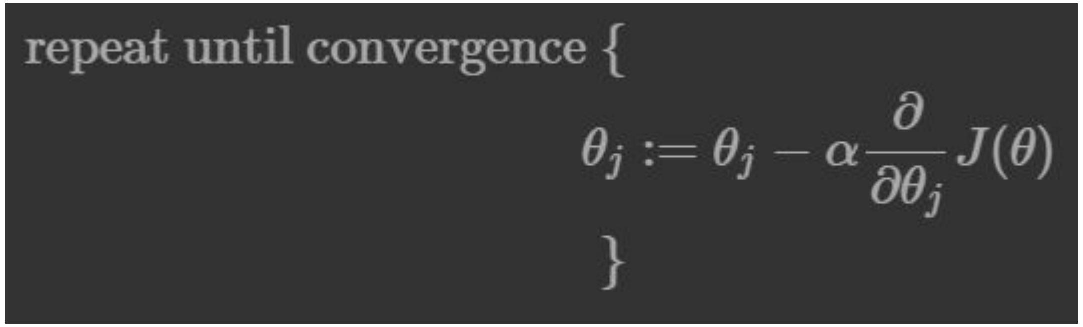
Derivation of Cost Function

It's now time to find the best values for θs parameters in the cost function,

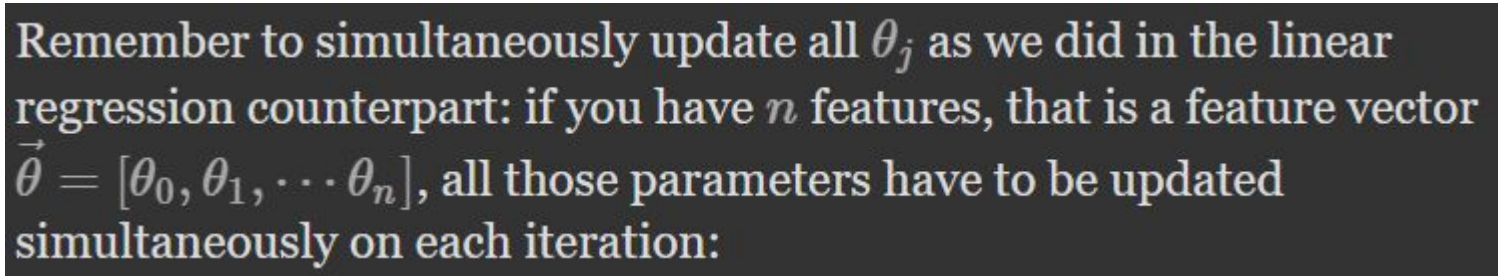


$$\min_{\theta} J(\theta)$$

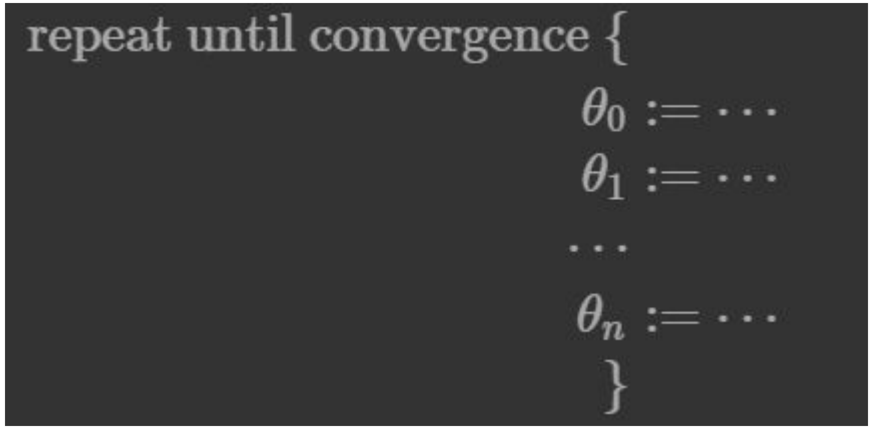
To minimize the cost function we have to run the gradient descent function on each parameter:



```
repeat until convergence {
    $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$
}
```

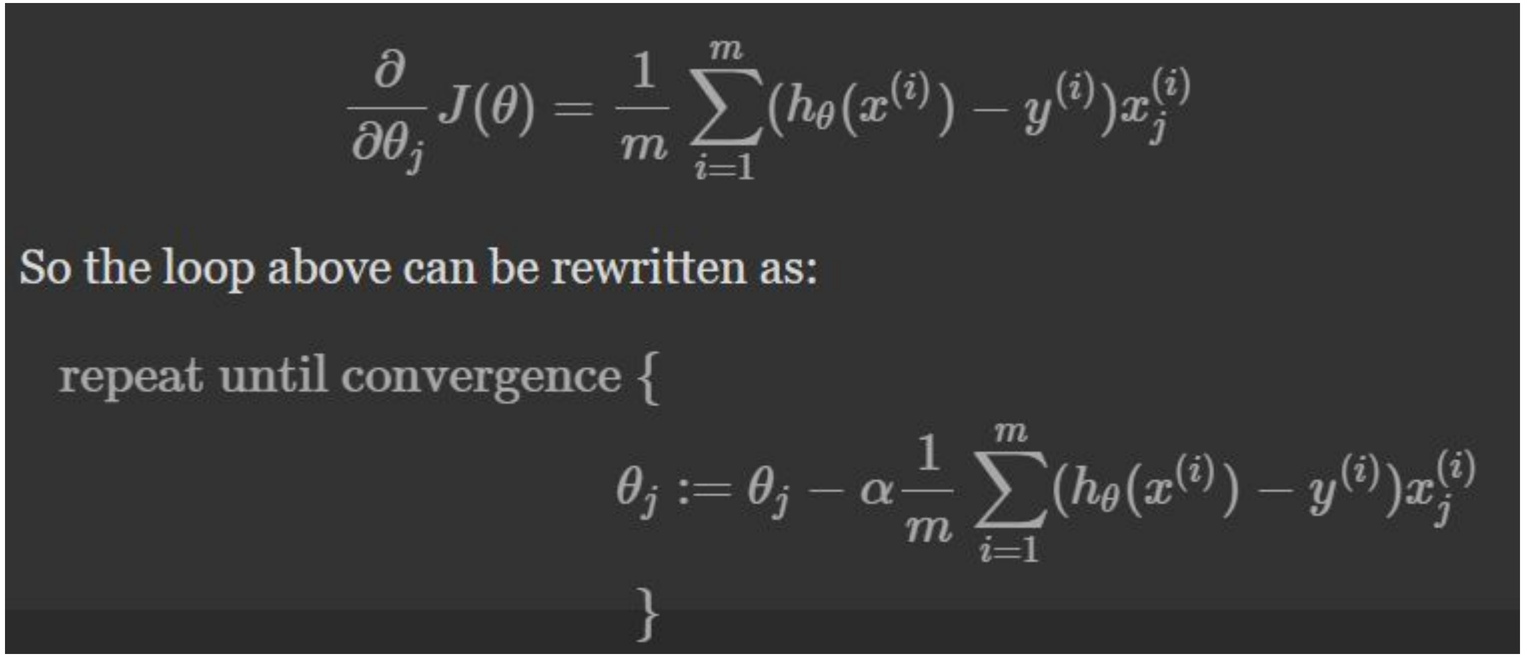


Remember to simultaneously update all θ_j as we did in the linear regression counterpart: if you have n features, that is a feature vector $\vec{\theta} = [\theta_0, \theta_1, \cdots \theta_n]$, all those parameters have to be updated simultaneously on each iteration:



```
repeat until convergence {
    $\theta_0 := \cdots$
    $\theta_1 := \cdots$
    $\cdots$
    $\theta_n := \cdots$
}
```

Derivation of Cost Function



$$\frac{\partial}{\partial\theta_j}J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

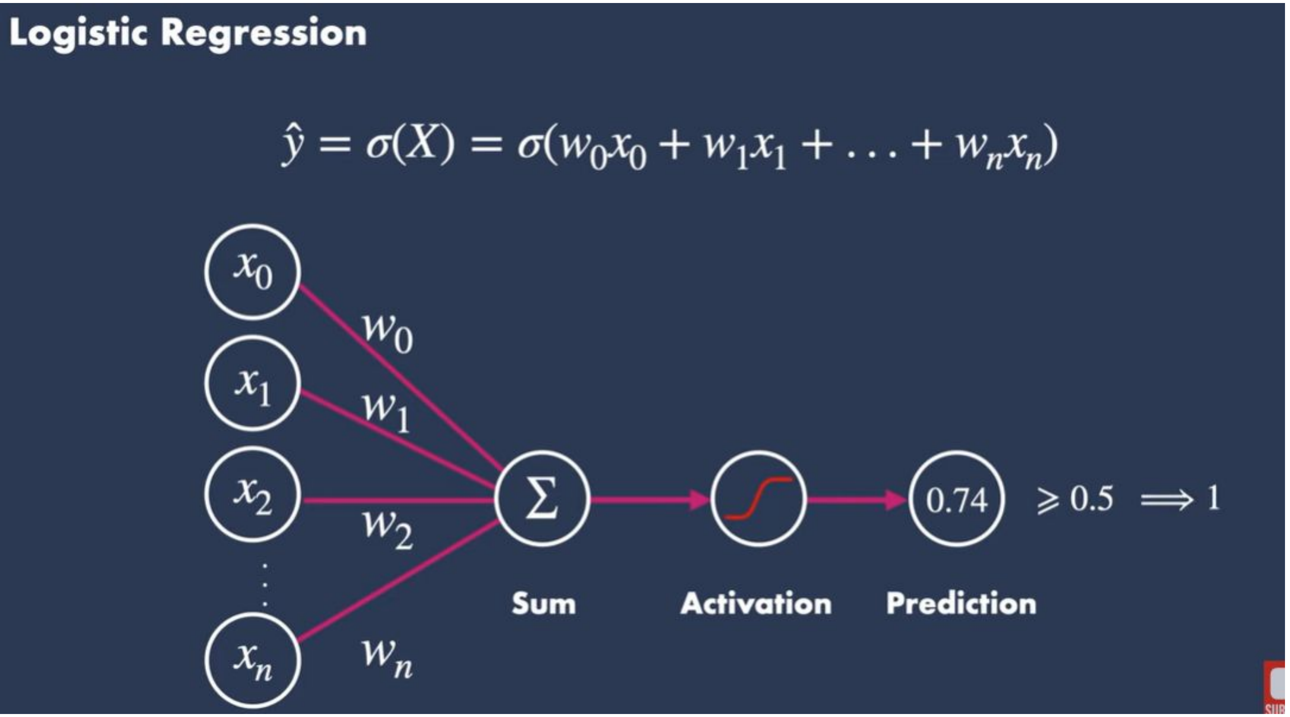
So the loop above can be rewritten as:

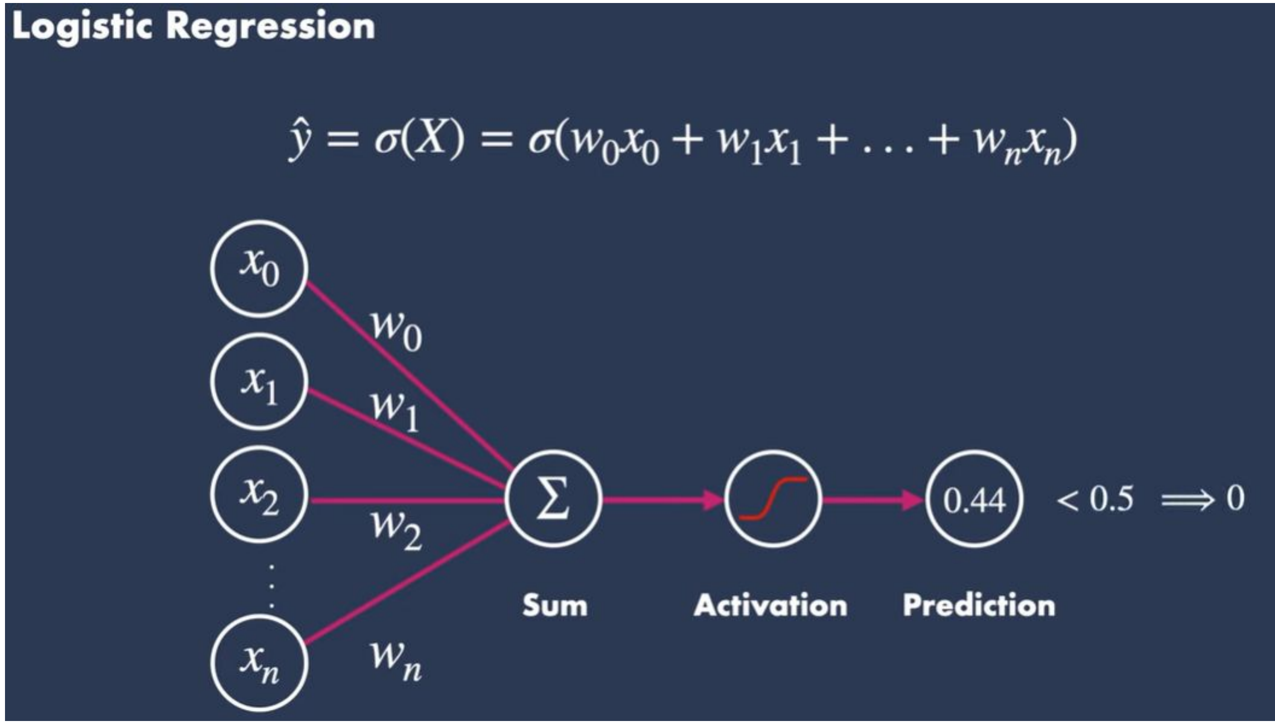
```
repeat until convergence {
        $\theta_j := \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$
}
```

- ✔ Now since we have our derivative of the cost function, we can write our gradient descent algorithm as:
  - – If the slope is negative (downward slope) then our gradient descent will add some value to our new value of the parameter directing it towards the minimum point of the convex curve. Whereas if the slope is positive (upward slope) the gradient descent will minus some value to direct it towards the minimum point.

Making Prediction

- ✔ Suppose we have two possible outcomes, true and false, and have set the threshold as 0.5. A probability less than 0.5 would be mapped to the outcome false, and a probability greater than or equal to 0.5 would be mapped to the outcome true.





Making Prediction

- ✔ Suppose, a regression model is fit using some training data to obtain *β* and *x* represents the input features:

$$z = \beta^{t} x$$

$$\beta^{t} = [\ 15\ 1]$$

$$x^{t} = [0.1\ -0.5]$$

- ✔ The probability of *z* being mapped to 1 is given by the equation:
- ✔

$$\sigma\,(z) = \sigma(\ 0.1\ *\ 15 + (-0.5*1)) = \sigma(\ 1\ )$$

$$= 1/\ (1 + e^{-1}\ ) \approx 0.7311$$

Making Prediction

✔ Let’s say we have a model that can predict whether a person is male or female based on their height (completely fictitious). Given a height of 150cm is the person male or female.

We have learned the coefficients of b0 = -100 and b1 = 0.6. Using the equation we can calculate the probability of male given a height of 150cm or more formally P(male|height=150).

$y = e^{(b0 + b1*X)} / (1 + e^{(b0 + b1*X)})$

$y = \exp(-100 + 0.6*150) / (1 + \mathrm{EXP}(-100 + 0.6*X))$

$y = 0.0000453978687$

Or a probability of near zero that the person is a male.

In practice we can use the probabilities directly. Because this is classification and we want a crisp answer, we can snap the probabilities to a binary class value, for example:

0 if $p(\text{male}) < 0.5$

1 if $p(\text{male}) >= 0.5$

Sample Short questions

- ✔ **Why logistic regression cost function is non convex?**
  - – The sigmoid function introduces non-linearity, resulting in a non-convex cost function. It has multiple local minima, making optimization challenging, as traditional gradient descent may converge to suboptimal solutions.
- ✔ **What is a cost function in simple terms?**
  - – A cost function measures the disparity between predicted values and actual values in a machine learning model. It quantifies how well the model aligns with the ground truth, guiding optimization.
- ✔ **Is the cost function for logistic regression always negative?**
  - – No, the cost function for logistic regression is not always negative. It includes terms like -log(h(x)) and -log(1 – h(x)), and the overall value depends on the predicted probabilities and actual labels, yielding positive or negative values.
- ✔ **Why only sigmoid function is used in logistic regression?**
  - – The sigmoid function maps real-valued predictions to probabilities between 0 and 1, facilitating the interpretation of results as probabilities. Its range and smoothness make it suitable for binary classification, ensuring stable optimization.

Thank You