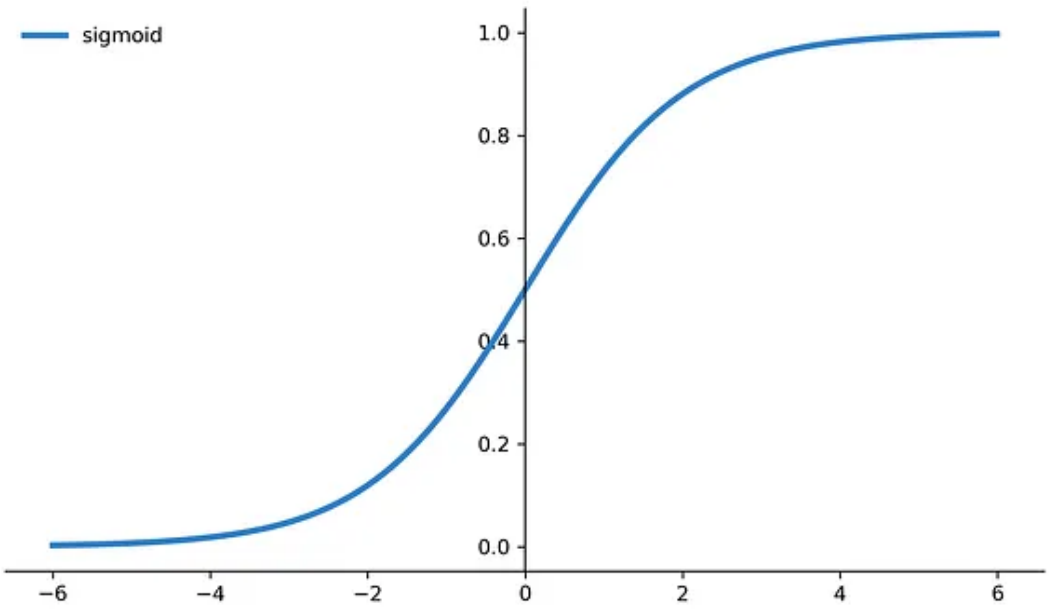
Maximum Likelihood Estimation in Logistic Regression

Arun Addagatla · Follow
7 min read · Apr 26, 2021

Listen Share

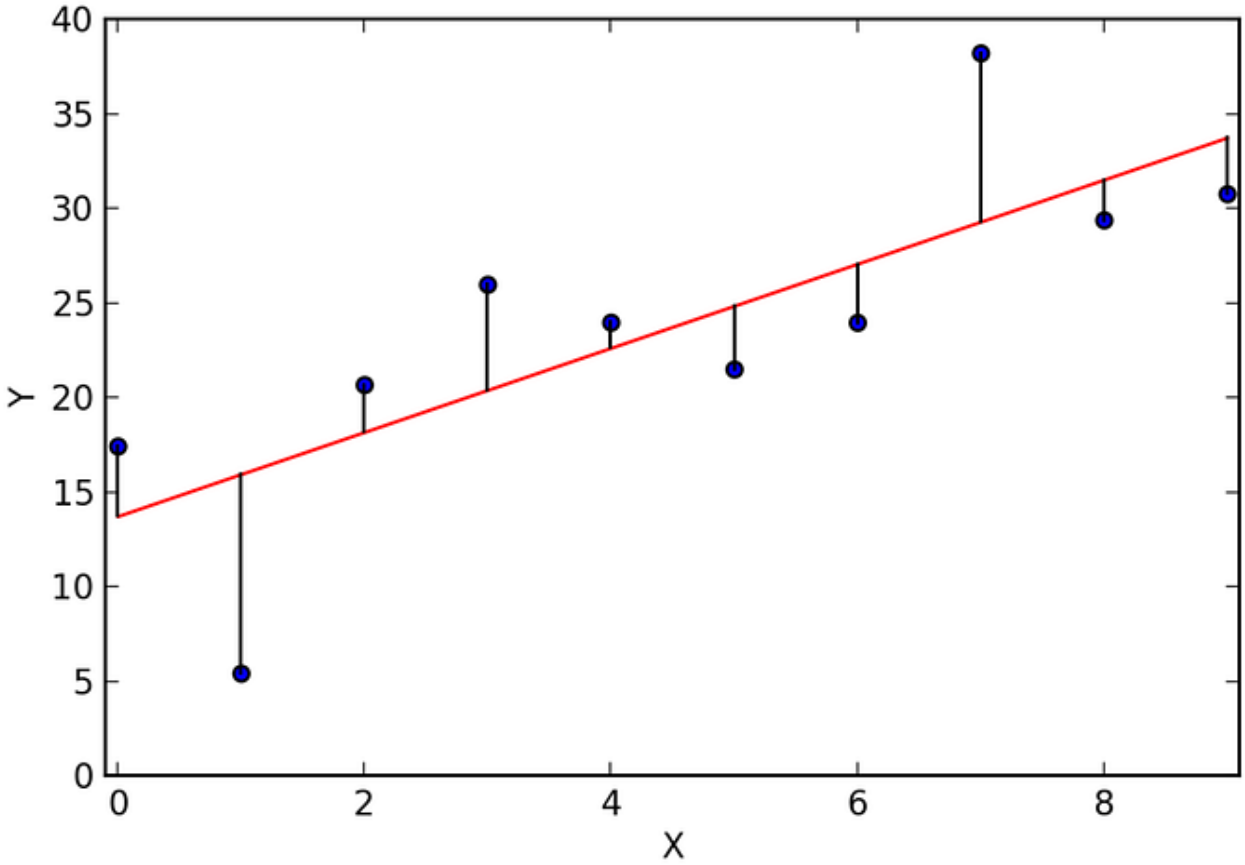


> *If you are not familiar with logistic regression, feel free to check out Understanding Logistic regression*

Logistic regression is very similar to regular old linear models like linear regression but the big difference is that logistic regression uses the log odds on the y-axis. In logistic regression, we make sure that the curve fitted makes the range of response variable y belong to 0 and 1.

> *If you are not familiar with linear regression, feel free to check out Linear Regression in a Nutshell*

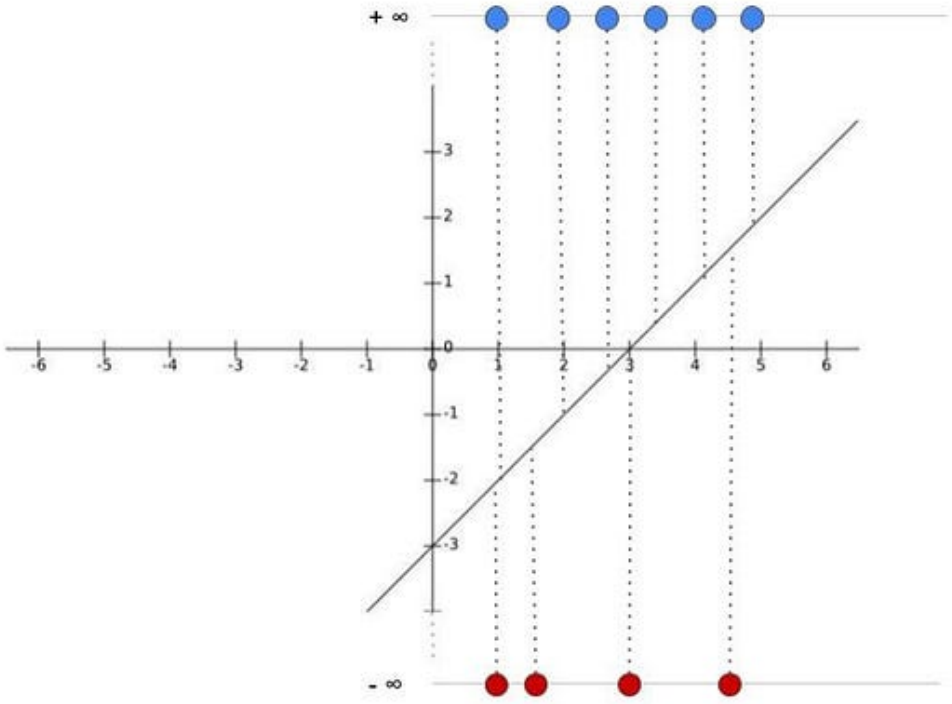
As we know that in linear regression to find the best fitting we start with some data and we fit a line to them using the least-squares i.e we measure the residuals (the distance between the data and the line) then square them, so that negative values do not cancel out positive values, and add them up.



Then we rotate the line a little bit and do the same. The line with the smallest sum of squared residuals is the line chosen to fit best.

Why can't we make use of least-squares to find the best fitting line in logistic regression?

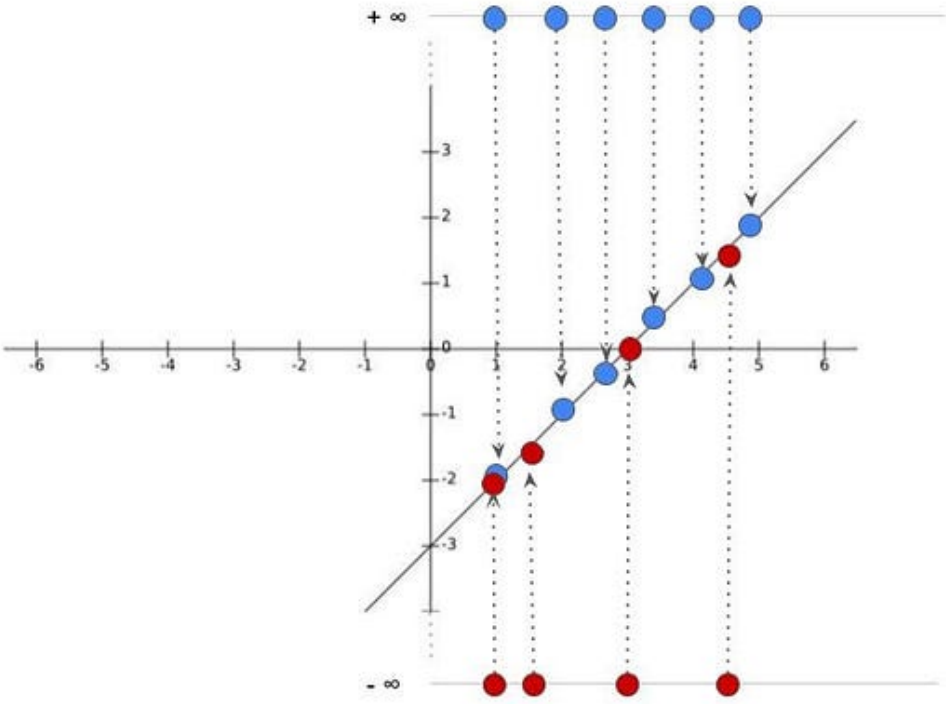
Well, to answer this we need to recall logistic regression. Our goal in logistic regression is to draw the best fitting S-curve for given data points. And in logistic regression, we transform the y-axis from the probabilities to log(odds). The problem is that this transformation pushes the data points to positive and negative infinity as shown below



So we can't use least-squares to find the best fitting line as the residuals are also equal to positive and negative infinity. Instead of least-squares, we make use of the maximum likelihood to find the best fitting line in logistic regression.

In Maximum Likelihood Estimation, a probability distribution for the target variable (class label) is assumed and then a likelihood function is defined that calculates the probability of observing the outcome given the input data and the model. This function can then be optimized to find the set of parameters that results in the largest sum likelihood over the training dataset.

By applying Maximum Likelihood estimation, the first thing we do is project the data points onto the line. This gives each data point a log(odds) value.



We then transform this log(odds) to probabilities using the formula

$$p = \frac{e^{log(odds)}}{1 + e^{log(odds)}}$$

Derivation of the above formula

As we already know that

$$log\left(\frac{p}{1-p}\right) = log(odds)$$

Exponentiate both the sides

$$\left(\frac{p}{1-p}\right) = e^{log(odds)}$$

$$p = (1-p) * e^{log(odds)}$$

$$p(1 + e^{log(odds)}) = e^{log(odds)}$$

$$p = \frac{e^{log(odds)}}{1 + e^{log(odds)}}$$

Once we calculate the probabilities, we will plot them to get an s-curve. Now, We just keep rotation the log(odds) line and projecting the data points onto it and then transforming it to probabilities and calculating the log-likelihood. We will repeat this process until we maximize the log-likelihood.

> *The algorithm that finds the line with the maximum likelihood is pretty smart each time it rotates the line, it does so in a way that increases the log-likelihood. Thus, the algorithm can find the optimal fit after a few rotations.*

Estimation of Log-likelihood function

As explained, Logistic regression uses the Maximum Likelihood for parameter estimation. The logistic regression equation is given as

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

The parameters to be estimated in the equation of a logistic regression are β vectors.

To estimate β vectors consider the N sample with labels either 0 or 1.

Mathematically, For samples labeled as ‘1’, we try to estimate β such that the product of all probability p(x) is as close to 1 as possible. And for samples labeled as ‘0’, we try to estimate β such that the product of all probability is as close to 0 as possible in other words (1 — p(x)) should be as close to 1 as possible.

The above intuition is represented as

$$for\ samples\ labelled\ as\ 1: \prod_{s\ in\ y_i=1} p(x_i)$$

$$for\ samples\ labelled\ as\ 0: \prod_{s\ in\ y_i=0} (1-p(x_i))$$

> *x_i represents the feature vector for the i^{th} sample.*

On combining the above conditions we want to find β parameters such that the product of both of these products is maximum over all elements of the dataset.

$$L(\beta) = \prod_{s\ in\ y_i=1} p(x_i) \ * \prod_{s\ in\ y_i=0} (1-p(x_i))$$

> *This function is the one we need to optimize and is called the **likelihood function**.*

Now, We combine the products and take log-likelihood to simply it further

$$L(\beta) = \prod_{s} \left(p(x_i)^{y_i} \ * (1-p(x_i))^{1-y_i} \right)$$

$$l(\beta) = \sum_{i=1}^{n} y_i\ log(p(x_i)) + (1-y_i)\ log(1-p(x_i))$$

$$where,\ l(\beta)\ is\ log-likelihood$$

Let's substitute p(x) with its exponent form

$$l(\beta) = \sum_{i=1}^{n} y_i \, log\left(\frac{1}{1+e^{-\beta x_i}}\right) + (1-y_i) \, log\left(\frac{e^{-\beta x_i}}{1+e^{-\beta x_i}}\right)$$

$$l(\beta) = \sum_{i=1}^{n} y_i \left[log\left(\frac{1}{1+e^{-\beta x_i}}\right) - log\left(\frac{e^{-\beta x_i}}{1+e^{-\beta x_i}}\right)\right] + log\left(\frac{e^{-\beta x_i}}{1+e^{-\beta x_i}}\right)$$

$$l(\beta) = \sum_{i=1}^{n} y_i \left[log\left(e^{\beta x_i}\right)\right] + log\left(\frac{e^{-\beta x_i}}{1+e^{-\beta x_i}} * \frac{e^{\beta x_i}}{e^{\beta x_i}}\right)$$

$$l(\beta) = \sum_{i=1}^{n} y_i \beta x_i + log\left(\frac{1}{1+e^{\beta x_i}}\right)$$

Now we end up with the final form of the log-likelihood function which is to be optimized and is given as

$$l(\beta) = \sum_{i=1}^{n} y_i \beta x_i - log\left(1+e^{\beta x_i}\right)$$

Maximizing Log-likelihood function

The goal here is to find the value of β that maximizes the log-likelihood function. There are many methods to do so like

- Fixed-point iteration
- Bisection method
- Newton-raphson method
- Muller's method

In this article, we will be using the **Newton-raphson method** to estimate the β vector. The Newton-raphson equation is given as

$$\beta^{t+1} = \beta^t - \frac{\nabla_\beta l(\beta^t)}{\nabla_{\beta\beta} l(\beta^t)}$$

$where,$

$\nabla_\beta l(\beta^t)\ is\ first\ order\ derivative\ (gradient)$

$\nabla_{\beta\beta} l(\beta^t)\ is\ second\ order\ derivative$

$t\ is\ current\ iteration\ number$

Let's determine the gradient first. To determine gradient we will take the first-order derivative of our log-likelihood function

$$\nabla_\beta l = \nabla_\beta \sum_{i=1}^{n} y_i \beta x_i - log(1 + e^{\beta x_i})$$

$$\nabla_\beta l = \sum_{i=1}^{n} \nabla_\beta \left[y_i \beta x_i - log(1 + e^{\beta x_i}) \right]$$

$$\nabla_\beta l = \sum_{i=1}^{n} \nabla_\beta \left[y_i \beta x_i \right] - \nabla_\beta \left[log(1 + e^{\beta x_i}) \right]$$

$$\nabla_\beta l = \sum_{i=1}^{n} y_i \, x_i \; - \left[\frac{1}{(1 + e^{\beta x_i})} \; * \; e^{\beta x_i} x_i \right]$$

$$\nabla_\beta l = \sum_{i=1}^{n} y_i \, x_i \; - \left[\frac{1}{(1 + e^{-\beta x_i})} \; * \; x_i \right]$$

Now, we will replace the exponential term with probability

$$\nabla_\beta l = \sum_{i=1}^{n} y_i \, x_i \; - \left[\, p(x_i) * x_i \, \right]$$

$$\nabla_\beta l = \sum_{i=1}^{n} \left[\, y_i - p(x_i) \, \right] * x_i$$

> *The matrix representation of gradient will be*

$$\nabla_\beta l = X^T (Y - \hat{Y})$$

We are done with the numerator term of newton-raphson. Now we will calculate the denominator i.e second-order derivative which is also called as Hessian Matrix.

To do so we will find derivate of gradient as

$$\nabla_{\beta\beta} l = \nabla_\beta \sum_{i=1}^{n} \left[\, y_i - p(x_i) \, \right] * x_i$$

$$\nabla_{\beta\beta} l = \sum_{i=1}^{n} \nabla_\beta \left[\, y_i - p(x_i) \, \right] * x_i$$

$$\nabla_{\beta\beta} l = \sum_{i=1}^{n} \nabla_\beta \, p(x_i) \; * x_i$$

Now, We will replace probability with its equivalent exponential term and compute its derivative

$$\nabla_{\beta\beta} l = \sum_{i=1}^{n} \nabla_{\beta} \left[\frac{1}{1+e^{-\beta x_i}}\right] * x_i$$

$$\nabla_{\beta\beta} l = \sum_{i=1}^{n} \left[\frac{1}{1+e^{-\beta x_i}}\right]^2 e^{-\beta x_i}(-x_i)\, x_i$$

$$\nabla_{\beta\beta} l = \sum_{i=1}^{n} \left[\frac{e^{-\beta x_i}}{1+e^{-\beta x_i}}\right] \left[\frac{1}{1+e^{-\beta x_i}}\right] x_i^T\, x_i$$

Resubstitute exponential term as probability

$$\nabla_{\beta\beta} l = \sum_{i=1}^{n} p(x_i)(1-p(x_i)) x_i^T\, x_i$$

> *The matrix representation of the Hessian matrix will be*

$$\nabla_{\beta\beta} l = -X^T P(1-P)X$$

$$\nabla_{\beta\beta} l = -X^T W X$$

As we have calculated gradient and Hessian matrix, plugging these two terms into the newton-raphson equation to get a final form

$$\beta^{(t+1)} = \beta^{(t)} - \frac{\nabla_{\beta} l(\beta^t)}{\nabla_{\beta\beta} l(\beta^t)}$$

$$\beta^{(t+1)} = \beta^{(t)} - \frac{X(Y-\hat{Y}^{(t)})}{(-X^T W^{(t)} X)}$$

$$\beta^{(t+1)} = \beta^{(t)} + X(Y-\hat{Y}^{(t)})(X^T W^{(t)} X)^{-1}$$

Now, we will execute the final equation for t number of iterations until the value of β converges.

Once the coefficients have been estimated we can then plug in the values of some feature vector X to estimate the probability of it belonging to a specific class.

We should choose a threshold value above which belongs to class 1 and below which is class 0.

Conclusion

The Maximum Likelihood Estimation (MLE) is a method of estimating the parameters of a logistic regression model. This estimation method is one of the most widely used. The method of maximum likelihood selects the set of values of the model parameters that maximize the likelihood function.

The likelihood function is the probability that the observed values of the dependent variable may be predicted from the observed values of the independent variables. The likelihood varies from 0 to 1.

The MLE is the value that maximizes the probability of the observed data. And is an example of a point estimate because it gives a single value for the unknown parameter.

Thanks for reading this article! Leave a comment below if you have any questions. Be sure to follow @ArunAddagatla, to get notified regarding the latest Data Science and Deep Learning articles.

> You can connect with me on **LinkedIn**, **Github**, **Kaggle**, or by visiting **Medium.com**.

Maximum Likelihood | Logistic Regression | Machine Learning | Data Science | Statistics

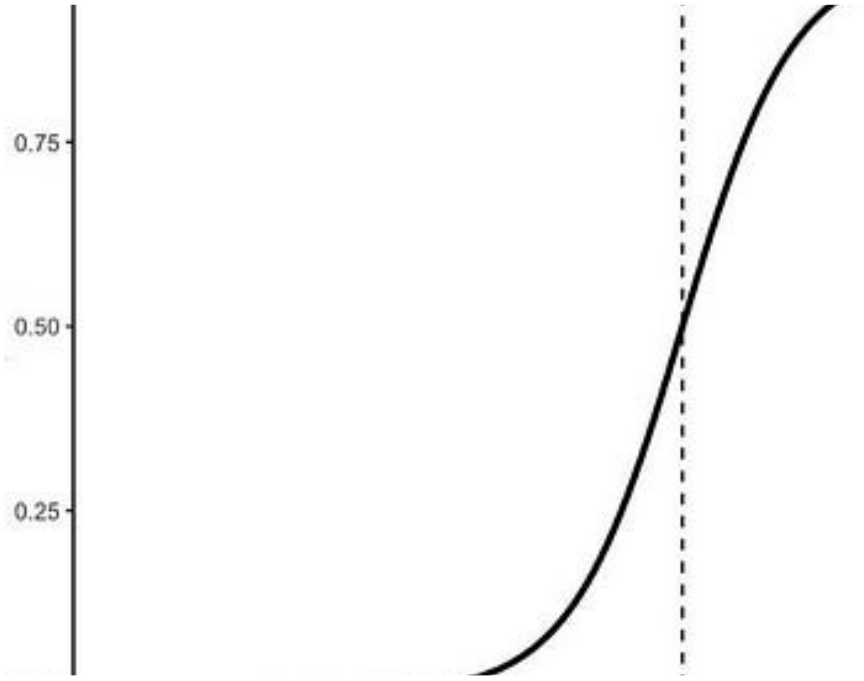
Follow

Written by Arun Addagatla

42 Followers

I am a Third-year Computer Engineering undergraduate student with an interest in Data Science, Deep Learning, and Computer Networking.

More from Arun Addagatla



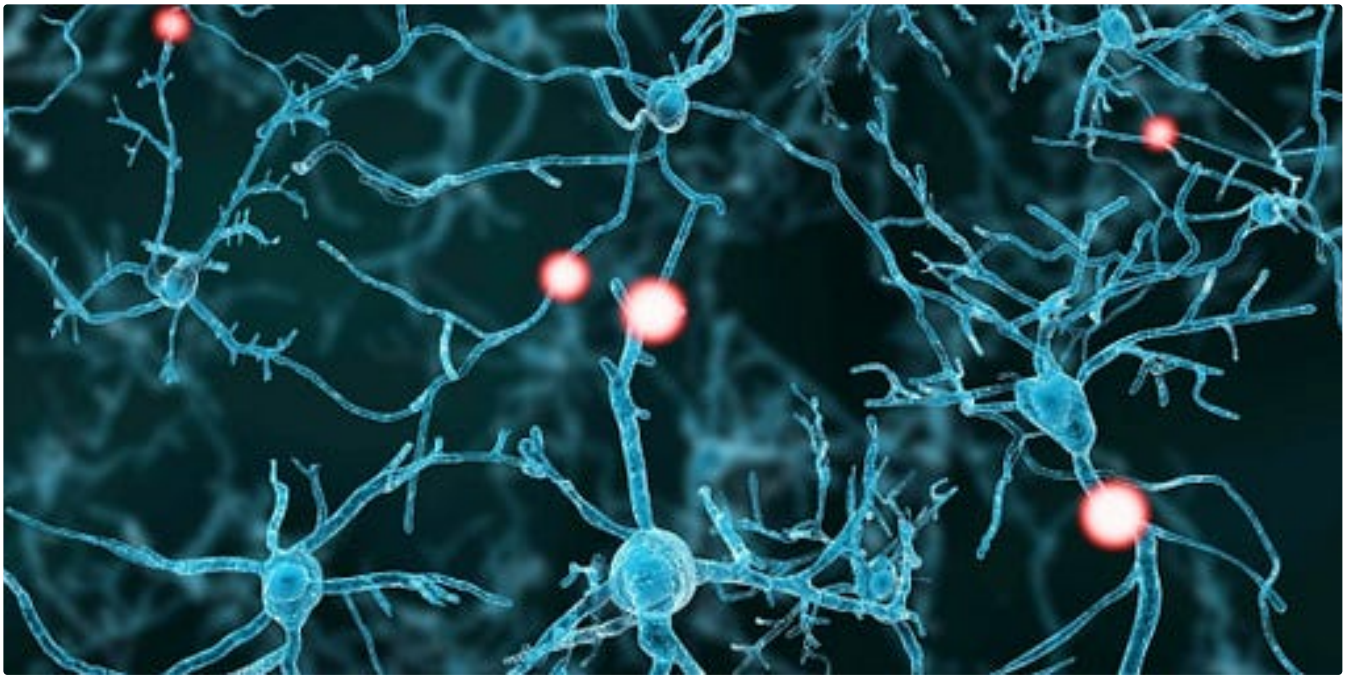
Arun Addagatla in Nerd For Tech

Understanding Logistic Regression

So far, we either looked at estimating the conditional expectations of continuous variables (as in regression). However, there are many...

7 min read · Apr 23, 2021

122



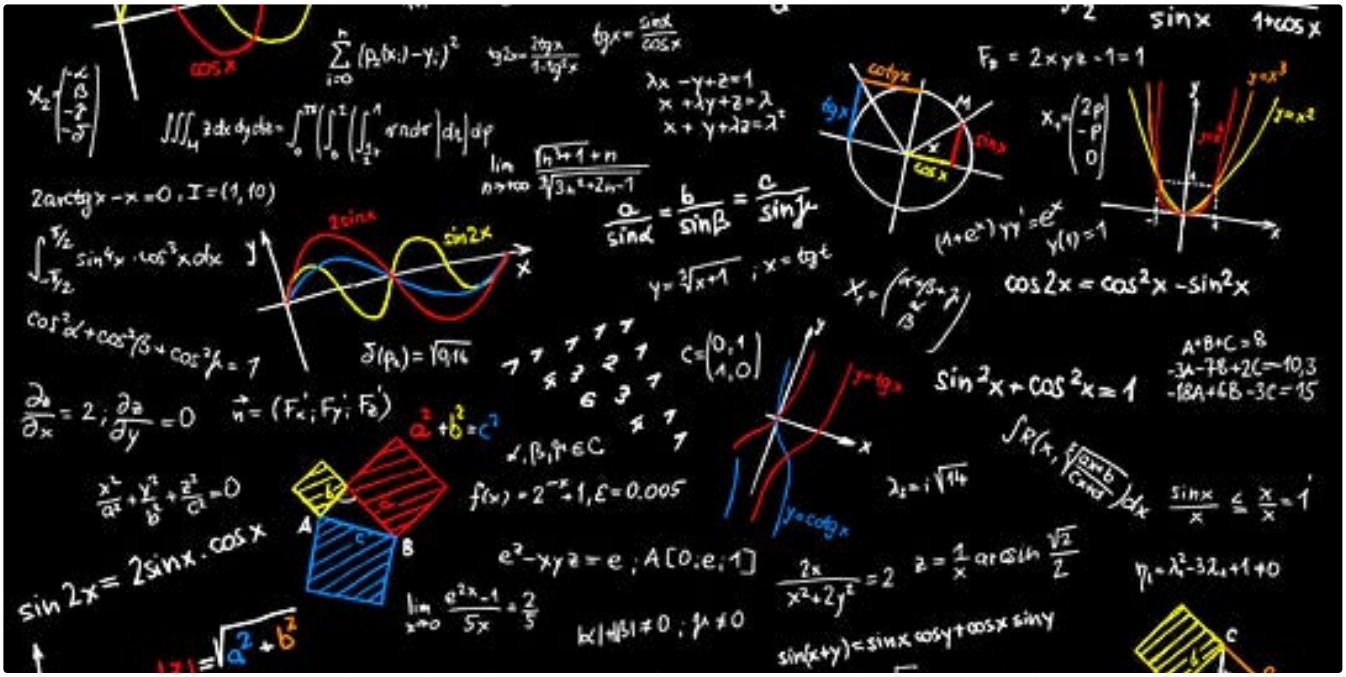
Arun Addagatla in The Startup

A Study of Artificial Neural Networks (ANN)

You may have heard the words Machine Learning, Artificial Intelligence, and Artificial Neural Networks in recent times. All of these are...

8 min read · Dec 8, 2020

69



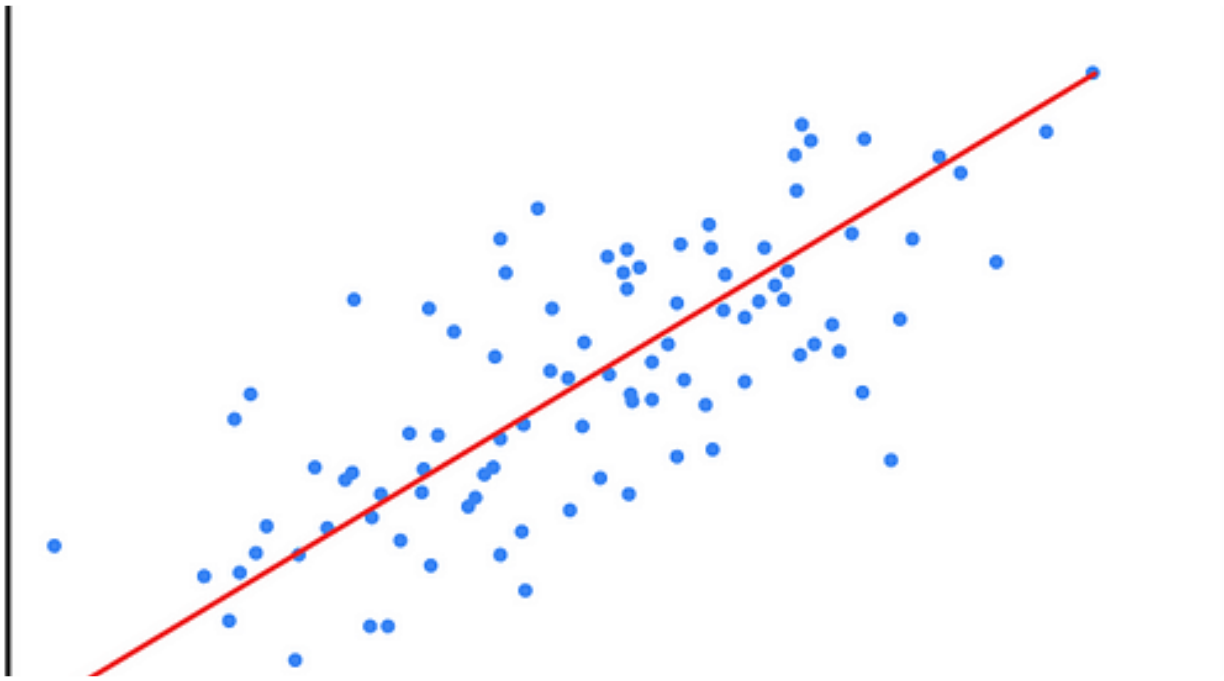
Arun Addagatla in Geek Culture

Performance Measures for Regression

Regression algorithms have been proven effective for making predictions in many sectors. One of the key phases in machine learning...

6 min read · Apr 5, 2021

1



Arun Addagatla in Geek Culture

Ordinary Least Squares Regression

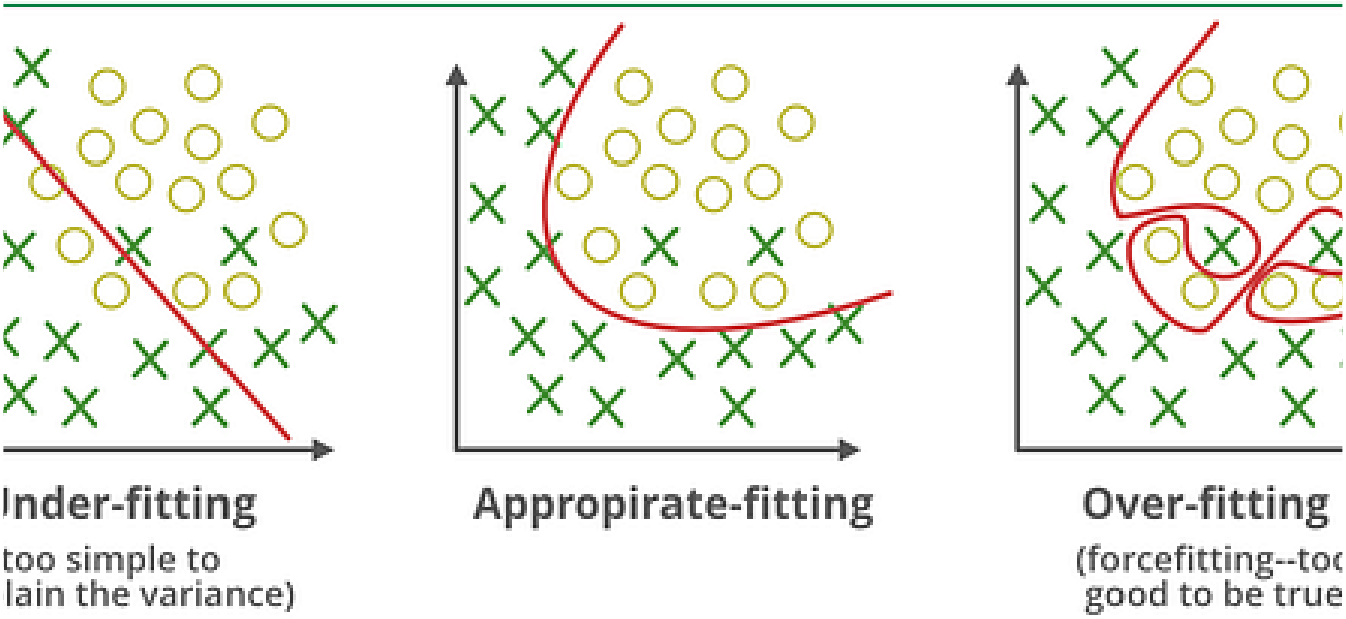
Ordinary Least Squares regression (OLS) is more commonly named linear regression algorithm is a type of linear least-squares method for...

7 min read · Apr 12, 2021

4 1

See all from Arun Addagatla

Recommended from Medium



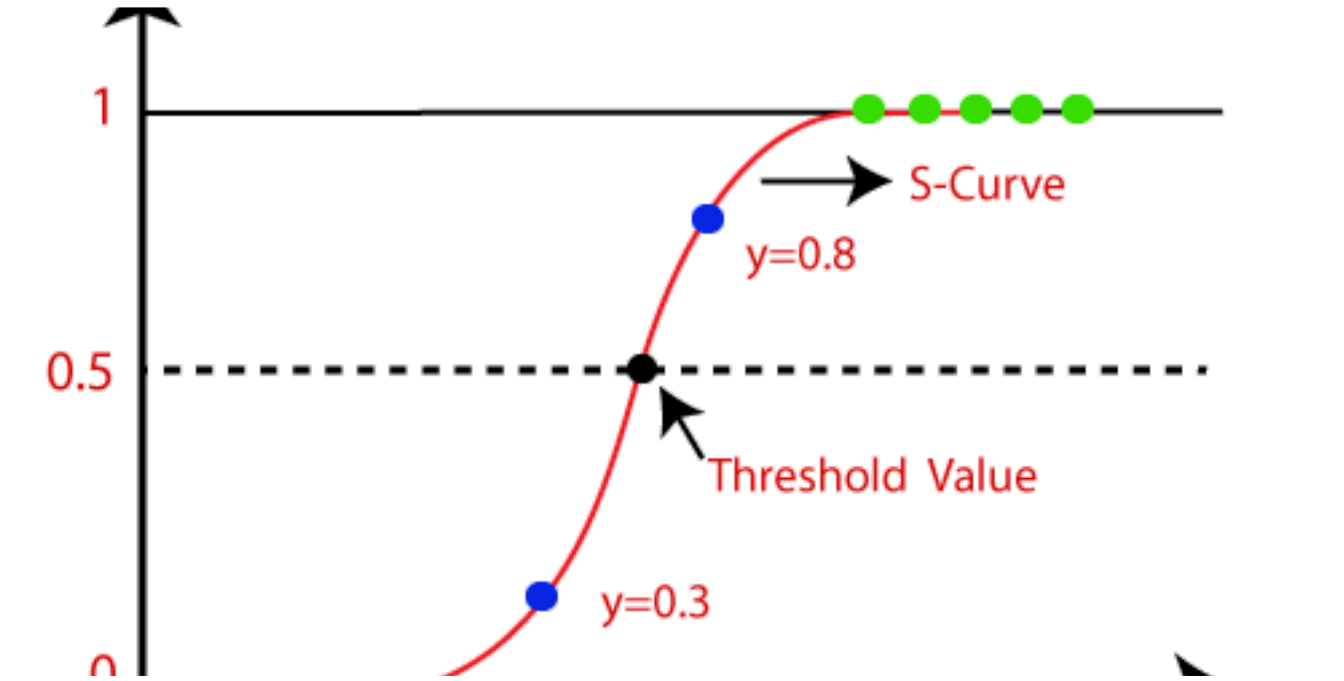
Peter Karas in Artificial Intelligence in Plain English

L1 (Lasso) and L2 (Ridge) regularizations in logistic regression

Logistic regression , Lasso and Rigde regularizations, derivations, math

· 6 min read · Feb 3

57 2



Peter Karas in Artificial Intelligence in Plain English

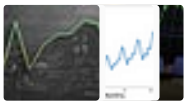
Logistic Regression in Depth

Logistic regression, activation function, derivation, math

✦ · 7 min read · Jan 31

281 1

Lists



Predictive Modeling w/ Python
18 stories · 58 saves

Practical Guides to Machine Learning
10 stories · 73 saves

Natural Language Processing
369 stories · 23 saves

New_Reading_List
173 stories · 8 saves

Matt Chapman in Towards Data Science

The Portfolio that Got Me a Data Scientist Job

Spoiler alert: It was surprisingly easy (and free) to make

✦ · 10 min read · Mar 24

Sadrach Pierre, Ph.D. in Towards Data Science

Mastering P-values in Machine Learning

Understanding P-values and ML use cases

· 7 min read · Jan 6

172

Unbecoming

10 Seconds That Ended My 20 Year Marriage

It's August in Northern Virginia, hot and humid. I still haven't showered from my morning trail run. I'm wearing my stay-at-home mom...

 · 4 min read · Feb 16, 2022

 51K  812