Linear Regression: Contrasting the Statistical and the Machine Learning View

May 2, 2026

Linear Regression is often used as the "Hello World" of deep learning, i.e. it is used as the very first example to introduce concepts of artificial neural networks and as such it is a fundamental concept in machine learning (ML). However, the method is deeply rooted in statistics dating back to the early 18th century. Thus, since the same problem is tackled from two different perspectives (statistics and ML) different wordings exist for similar concepts and different aspects are emphasized depending on if we are team statistics or team machine learning. Inevitably, this can get confusing and therefore its worthwhile to take a closer look at the overlap and distinctions of both approaches. In this article we will understand the basics of linear regression from a statistical perspective (where I put the focus) and contrast them with the machine-learning view. We will look at what

- regression analysis is about,
- what simple linear regression is,
- learn about the normal error regression model
- and how its vectorized for efficient computation.
- I will briefly sketch which methods are used to compute the model parameters.

The post is beginner-friendly, however it requires some basic statistical knowledge, such as random variables, parameter estimation (covered here), and confidence intervals (introduced here). You also need some basic knowledge of linear algebra to follow the vectorization parts. The main resources for this article were [2], [3], and [1].

1 What is regression analysis?

Regression analysis is about:

- Predicting a numerical (continuous) variable by means of one (simple regression) or several other numerical variables (multiple regression).
- Investigating whether there is a relationship between one or several numerical variables and, if so, what this relationship looks like.
- Making statements about how trustworthy possible predictions are.

For example, can you make a statement about the weight of a person if you know his or her height? You probably reason that, on average, a taller person is heavier than a shorter person. But how heavy? And is this assumption even correct? To investigate these questions systematically, we need to formalize the problem. First, we model "height" (X) and "weight" (Y) as quantitative random variables taking numercial values (as opposed to categorical variables that take values like "red", "green", "hot", "cold", etc.). Next, we observe a specific height value x (the specific value that a random variable can take is usually denoted as a lowercase letter; for example, $X = x$ indicates that the random variable X takes the value x.) and wonder if we can make a statement about the weight of a person given that we know his or her height. This can be formalized as a conditional probability problem. We are asking what is the conditional distribution of weight given a particular observation for height. That is,

$$P(Y = y \mid X = x).$$

This is the focus of regression analysis: it provides the framework to study whether there is a relationship between one quantitative variable and another (or multiple others), and what that relationship looks like.

2 Linear regression in machine learning vs statistics

In ML, linear regression is considered a simple problem that is just complicated enough to introduce the basics of the ML-pipeline for artificial neural networks. Thus, it serves to illustrate the well known procedure of: choose a **model** and a **loss function** to create an **optimization problem** that you can then solve with an appropriate strategy - usually **gradient descent**. Then the **backpropagation algorithm** that efficiently computes the gradients (for gradient descent) is introduced, it is explained how to avoid unwanted artifacts like over- or underfitting, how to evaluate the performance of the model etc. Thus, as already stated - the focus is on introducing the ML-pipeline. The "learning" is considered done once the model makes good enough predictions on the test data.

In statistics we approach the problem from the field of **Statistical Inference**. We interpret the data as a **sample** (a small observation) of the **population** (the entire data that is unfortunately not available). We assume that the population was produced by a model or

process that has parameters. We can't know these parameters but only estimate them from the available sample. **Parameter estimation** is what the field of **Statistical Inference** is about. Therefore, we want to do parameter estimation as opposed to learning weights. In particular, as statisticians we do not suffice with "good enough" predictions. We aim at **optimal** estimates and importantly - we want to quantify how good they are using techniques like confidence intervals and hypothesis tests. To accomplish this statisticians put a lot of emphasize on the noise, i.e. the inability to perfectly predict the dependent by the independent variable. ML often ignores this or treats it only implicitly.

Also, the mean squared error, i.e. the difference between prediction and actual observed value is not just an intuitive loss (as it is often treated in ML). In statistics it is derived as the expected value of the squared difference between prediction and truth, which connects to the bias–variance decomposition [4].

Summing up, the methods that statisticians use are theoretically well justified. In ML literature the same concepts may be used but often - not always - the reader is spared the heavy theory. (This is the sole opinion of the author of this article.)

Also the terms and notations in ML differ slightly from the ones used in statistics. I'll point out to differences throughout the text.

3 What is simple linear regression?

Let us assume a simple numerical example to illustrate the ideas. Suppose you have measured the heights and weights of 10 people (in cm and kg), as shown in Table 1.

person	height (X)	weight (Y)
1	186	90
2	174	70
3	186	93
4	176	82
5	182	89
6	176	80
7	184	86
8	176	79
9	179	85
10	186	98

Table 1: Exemplary heights and weights of ten people.

We call the variable to be predicted the **dependent variable** or **response** Y, and the variable used for prediction the **predictor**, **independent**, or **explanatory** variable X. In

ML, these are typically called "input" and "output" variables. In our example, height is the predictor and weight the response. A problem involving only one predictor is called **simple linear regression**. When more than one predictor is involved, we speak of **multiple linear regression**.

Regression analysis does not magically tell us the true relationship; instead, we *assume* a form for that relationship and then evaluate how well it fits the data. In **linear regression** we assume this relationship is - well - linear. Thus, **simple linear regression** (SLR) assumes that one numerical variable can be predicted from another assuming a linear relationship.

The data in Table 1 is a **sample** from a larger **population**. Sample and population are fundamental concepts in statistics, the sample is used to make inferences about the population parameters. In ML, the same sample is referred to as "training data." The columns are the variables in statistics or "features" in ML. Each row is one observation. In statistics the i'th observation is denoted by a subscript in ML often a superscript in round parentheses is used, e.g. $\mathbf{x}^{(i)}$. The round parentheses distinguish it from an exponent. In the following, I'll borrow the notation from ML, i.e. the superscript, because I find it more convenient.

(Test yourself: What does regression mean? What is simple linear regression? What is multiple linear regression?)

4 Regression analysis is concerned with statistical relations

We distinguish between two types of relations between two variables:

- A **functional relation** (or deterministic relation) is of the form $Y = f(X)$, where the same x always maps to the same y. For example, in the deterministic relation $y = 3x$, when $x = 2$ the value of y is always 6.
- A **statistical relation**, in contrast, is one where the same value of X can correspond to different Y values due to random variation. For example, the same $x = 2$ might yield $y = 6$ in one observation and $y = 6.5$ in another, even though the central tendency lies along a line.

This is illustrated in Figure 1, which shows a functional relation (left) and a statistical relation (right). In the statistical case, the y-values are not completely random; they roughly follow a line but **scatter** around it. Such a plot is called a **scatterplot**. Regression analysis is concerned with statistical relations of this kind and aims to find the line (or more general function) that the observed values scatter around. This line is the **regression function**. In the case of simple linear regression, the regression function is a straight line.

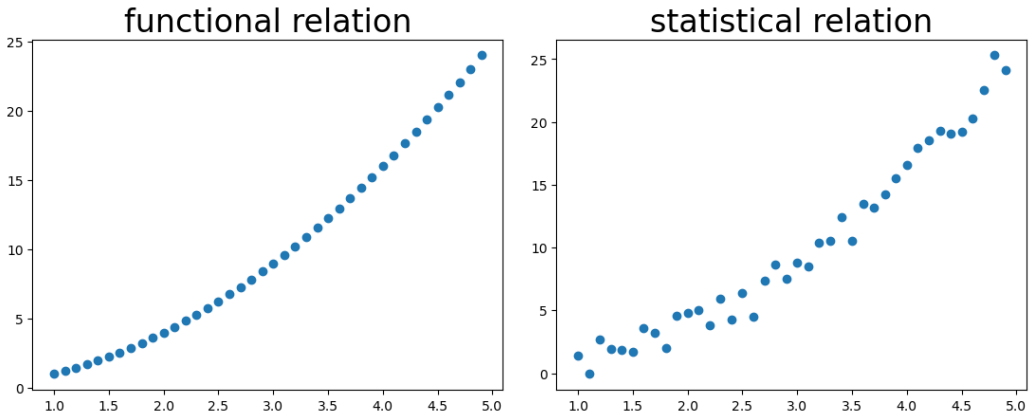

Figure 1: Example of a functional (left) and statistical (right) relation between two numerical variables.

In Figure 2 we apply this idea to our example. The blue points correspond to the data in Table 1. They form a statistical relation: the same x-values appear with different y-values, and the points seem to scatter around a line (shown in orange). This orange line is the **regression line** for simple linear regression.

5 The simple (linear) regression model and the normal error regression model

As just stated, in the case of simple linear regression the regression function is a line. (There are other regression models that assume different functional forms.) A line can be described mathematically by a line equation. One form of line equation that we are probably all familiar with is

$$Y = mX + b,$$

where m is the slope and b the y-intercept. In regression, X is the independent (predictor) variable and Y the dependent (response) variable. We treat Y as a random variable that can take specific values. In contrast, in the standard simple linear regression model, the predictor X is treated as a fixed (non-random) variable.

It is also possible and sometimes necessary to treat X itself as a random variable; in this case, the model is often referred to as a **correlation model**, in contrast to the standard regression model where X is fixed and only Y is random [2].

In our example, the pairs $(x^{(i)}, y^{(i)})$ correspond to the height–weight measurements. Each $x^{(i)}$ is the value of the i-th observation of the predictor, e.g. $x^{(1)} = 186$ and $y^{(1)} = 90$. The

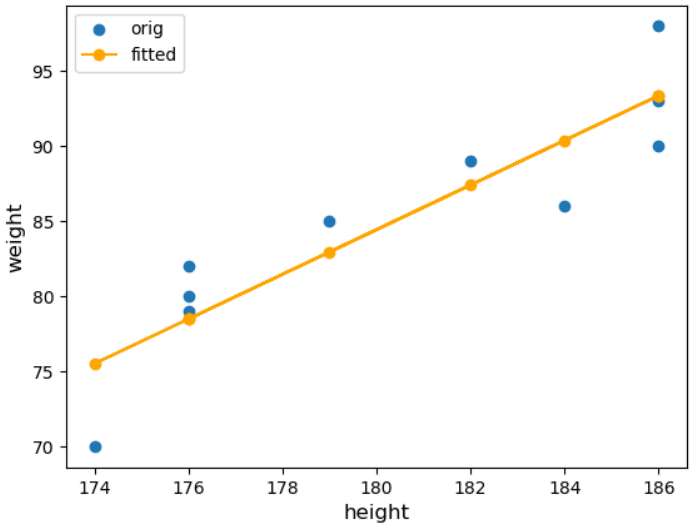

Figure 2: Plotting height vs. weight for the example data shows a statistical relation. The same x-values can have different y-values. The data appears scattered around a line (regression line, shown in orange).

line equation captures the deterministic part of the relationship, but it does not account for the **scatter** around the line that we observe in the data.

To incorporate this scatter, we add a random error term ϵ to the model:

$$[y^{(i)} = \beta_0 + \beta_1 x^{(i)} + \epsilon^{(i)} \tag{1}$$

We renamed m and x, here, β_1 is the slope and β_0 the y-intercept. Denoting these parameters β is common in the statistics literature, they are called the **regression coefficients**. In ML we would refer to them as "weights", and denote them as ω_0 and ω_1. The term $\epsilon^{(i)}$ represents the noise or error, which we model as a random variable. The main task in regression analysis is to **estimate** these coefficients from the available data. Note, that adding the noise is a design choice, we could think of many other ways to incorporate it into the model, e.g. we could also multiply the y-value by ϵ which would be called "multiplicative noise". However, as just said, in SLR the simple assumption is additive noise. We make the following assumptions about the errors and their distribution:

- All $\epsilon^{(i)}$ follow the same distribution.
- This distribution has mean zero: $E[\epsilon^{(i)}] = 0$.
- This distribution has a variance σ^2 that is constant but unknown.
- The errors are uncorrelated, i.e., there is no linear relationship between different $\epsilon^{(i)}$ and $\epsilon^{(j)}$ for $i \neq j$. (Thus, their covariance is 0. It does not imply that the errors are independent.)

The property that the error terms have the same (but unknown) variance is called **homoscedasticity**. Models that satisfy these assumptions are called **simple linear regression models**. If we additionally assume that the $\epsilon^{(i)}$ are independent and normally distributed, then

$$\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2), \tag{2}$$

and the model in 1 becomes the **normal error regression model**, a more specific version of the simple linear regression model.

A useful mnemonic for the assumptions of the normal error regression model comes from the word **LINE** [3]:

- **L** – the model is **linear** in the parameters β_0, β_1.
- **I** – the errors are **independent** of each other.
- **N** – the errors follow a **normal** distribution with zero mean.

- **E** – the errors have **equal** variances (homoscedasticity).

(Test yourself: What is the difference between the normal regression model and the regression function? What assumptions are made concerning the error in the normal error regression model? What is homoscedasticity?)

6 Vectorization

By **vectorization** we mean rewriting the model using linear algebra so that sums become matrix–vector operations. This is important both mathematically and computationally:

- It allows us to write equations more concisely and cleanly.
- In practice, matrix operations are highly parallelizable and can be accelerated on GPUs.

In programming, vectorization often refers to rewriting loops as array operations in libraries like NumPy, which is both simpler and faster.

We use the following notations:

- Matrices: bold uppercase, e.g. $\mathbf{A}$.
- Vectors: bold lowercase, e.g. $\mathbf{x}$; all vectors are column vectors.
- Scalars: regular lowercase, e.g. a.

So far our model (1) referred to a single observation. If we have n observations, we can write the collection of responses as a vector:

$$\mathbf{y} = \beta_0 + \beta_1 \mathbf{x} + \boldsymbol{\epsilon} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix} = \beta_0 + \beta_1 \begin{bmatrix} x^{(1)} \\ \vdots \\ x^{(n)} \end{bmatrix} + \begin{bmatrix} \epsilon^{(1)} \\ \vdots \\ \epsilon^{(n)} \end{bmatrix}. \tag{3}$$

This could be written more concisely by introducing a matrix form that incorporates the intercept. We define the **design matrix** as

$$\mathbf{X} = \begin{bmatrix} 1 & x^{(1)} \\ \vdots & \vdots \\ 1 & x^{(n)} \end{bmatrix},$$

and the parameter vector as

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}.$$

Then the model becomes

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}. \tag{4}$$

Here, the unknowns are the regression coefficients β_0, β_1 and the error variance σ^2, which we will estimate from the data.

(Test yourself: Why do we need vectorization? What is the design matrix?)

7 Vectorization for MLR

In the ML literature you will often see

$$y = \mathbf{w}^\top \mathbf{x} + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2) \tag{5}$$

This looks slightly different from our vectorized expression from above. The reason is that now we are looking at **multiple** linear regression. This means that each single (scalar) value of $y^{(i)}$ is computed not from a scalar $x^{(i)}$ as in **simple** linear regression, but that now $\mathbf{x}^{(i)}$ is a vector. In ML we would say that we use various **features** to predict $y^{(i)}$. An example would be to predict weight not only by height, but also by age. In this case $\mathbf{x}^{(i)} = [x_1, x_2]^T$, with x_1=height and x_2=age. For the case that we have two features we get:

$$y^{(i)} = \beta_0 \cdot 1 + \beta_1 x_1^{(i)} + \beta_2 x_2^{(i)} + \varepsilon^{(i)} \tag{6}$$

If we replace β_i by w_i and write this as a vector expression we get

$$y^{(i)} = \mathbf{w}^T \mathbf{x}^{(i)} + \epsilon^{(i)}$$

with

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}, \quad \mathbf{x}^{(i)} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

This is for a single observation indicated by the superscript i. If we want to compute the entire dataset at once - as is standard we get:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon}$$

where each row in $\mathbf{X}$ is an observation of the x values for a single y, thus for our example where the x-vector has two entries (features):

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} \\ \vdots & \vdots & \vdots \\ 1 & x_1^{(N)} & x_2^{(N)} \end{bmatrix}$$

8 Alternative regression model

For completeness, there is an **alternative form of the simple linear regression model** where we subtract the mean of $x^{(i)}$. In this form, the y-intercept represents the predicted value at the mean of x rather than at $x = 0$. The model is

$$y^{(i)} = \beta_0^* + \beta_1 \left(x^{(i)} - \bar{x} \right) + \epsilon^{(i)}, \tag{7}$$

with

$$\beta_0^* = \beta_0 + \beta_1 \bar{x}.$$

Here, β_0^* is the predicted value at the mean $\bar{x}$, and β_1 remains the same. Both forms are equivalent; the choice depends on whether you find it more interpretable to talk about the intercept at $x = 0$ or at the center of your data.

9 Implications of the normal error regression model

Above, we have defined the **normal error regression model**

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$: the errors are independent, identically distributed normal random variables with zero mean and constant variance σ^2. Because functions of random variables are random variables and because the $y^{(i)}$ are linear functions of the $\epsilon^{(i)}$, each $y^{(i)}$ is a random variable which is also normally distributed, with

$$E[y^{(i)}] = \beta_0 + \beta_1 x^{(i)} \quad \text{and} \quad \text{Var}(y^{(i)}) = \sigma^2.$$

The regression function—the line $y = \beta_0 + \beta_1 x$—is therefore the expected value of the response at each predictor level. Dropping the error term from (4) gives us this regression function, which is the "central line" that the observed data points scatter around.

Figure 3 (adapted from [2]) visualizes this idea. Each $y^{(i)}$ lies in a band around $E[y^{(i)}]$, and the distance between $y^{(i)}$ and its mean is determined by the realization $\epsilon^{(i)}$ of the normal error distribution. The arcs in the figure depict the underlying normal densities whose centers are the predicted means.

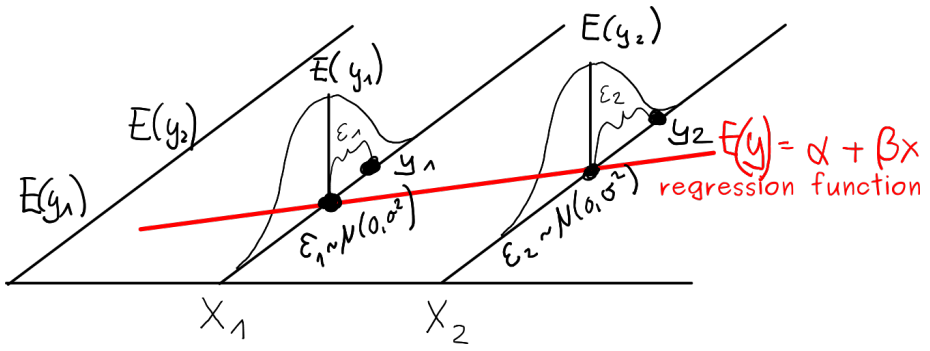

Figure 3: Visualization of the normal error linear regression model. Here, α corresponds to β_0 and β to β_1. Each $y^{(i)}$ lies within a band around its mean $E[y^{(i)}]$, and the distance is determined by the error term $\epsilon^{(i)}$.

(Test yourself: What is the difference between the normal regression model and the regression function? What assumptions are made concerning the error in the normal error regression model? What is the regression function?)

10 Be careful when you look at values beyond the scope of the model

Recall our example data in Figure 2. The original data consists of input values $x \geq 0$; the model was fit only to this region. The fitted line can be extended to negative x, but such an extrapolation may not make sense. For instance, predicting weight from negative heights would give a negative number which is completely meaningless. This illustrates that we must be cautious when interpreting results beyond the **scope** of the model, i.e., outside the range of x-values used to fit it.

11 Outlook - estimating the regression coefficients vs learning the weights

We now turn to the core task of regression analysis: **estimating** the unknown parameters β_0 and β_1 from the data. Since this article is already long I will suffice to sketch an

outlook for this. As explained, in statistics, we treat these as **parameters** of an underlying data-generating process; in machine learning, the same quantities are often called **weights** or **learned parameters**.

Let's look at the statistical perspective first: Because we have only a sample of the population, we cannot know the true parameter values exactly. Instead, we compute **point estimates** using appropriate criterion functions. (criterion functions are the term that statisticians use for cost, loss, or objective functions, where the three latter names are often used by the ML community). Two common approaches are:

- **Method of Least Squares**: uses only the model form without explicit distributional assumptions. It minimizes the sum of squared residuals:

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n} \left(y^{(i)} - \beta_0 - \beta_1 x^{(i)} \right)^2 .$$

  The resulting estimators are called the **ordinary least squares** (OLS) estimators.

- **Maximum Likelihood Estimation** (MLE): uses the normal error regression model. It treats the likelihood of the data under the assumed normal distribution and finds the parameter values that maximize this likelihood. For normally distributed errors, the MLE estimators of β_0 and β_1 coincide with the OLS estimators, and the MLE for σ^2 is closely related to the estimated residual variance.

As explained, in statistics, estimation does not stop at point estimates. Once β_0 and β_1 are computed, one usually constructs **confidence intervals** for these coefficients (and for predictions) to quantify the uncertainty in the estimates. These intervals are derived from the estimated error variance and the distributional assumptions encoded in the normal error model. Alternatively we can state hypotheses that the estimated value is correct or not and test the hypothesis.

In machine learning, the same goal—finding good values for the coefficients—can be approached differently. Instead of solving the least squares problem in closed form or via maximum likelihood, one can frame the task as an optimization problem and use a **neural network** or a simple linear model with **iterative optimization** (e.g., gradient descent). The model is then "trained" on the data until the predictions are sufficiently accurate.

12 Summary

Congratulations for having it made thus far. We've covered the basiscs about simple linear regression and contrasted the statistical approach with the one taken in machine learning.

In general, simple linear regression assumes a linear relationship between two numerical variables and attempts to predict the value of one variable (the dependent) by means of the other (the independent variable).

In statistics various assumptions are made concerning the mathematical model for this relationship. It emphasizes the role of the error which is reflected by the name "normal error model" meaning the error/noise in the model has a normal distribution and is additive. It emphasizes uncertainty and makes a couple of assumptions about the model (LINE: Linear, Independent, Normal, Equal variance). It also doesn't stop at estimating only the correlation coefficients (slope and y-intersection of the line equation) but continues with estimating confidence intervalls for those.

In machine learning linear regression is often used as an entry point for explaining deep learning. The correlation coefficients are the weights that are learned through optimization (usually backpropagation and gradient descent). As such it suffices with point estimates, i.e. usually no attempt is made to compute confidence intervalls over the computed weights.

Distinguishing between these two perspectives should help you thematically orient yourself. When you know whether an author takes a primarily statistical or machine learning perspective, you know what to expect. You'll also know where to look for further reading when you have questions about topics that are more statistics-heavy or ML-focused.

References

[1] Joseph K. Blitzstein and Jessica Hwang. *Introduction to Probability Second Edition*. 2019. URL: `https://drive.google.com/file/d/1VmkAAGOYCTORq1wxSQqy255qLJjTNvBI/view`.

[2] Michael H. Kutner. *Applied LinearStatistical Models, Fifth Edition*. Ed. by Brent Gordon. McGraw-Hill Irwin, 2004. ISBN: 0-07-238688-6.

[3] *Linear Regression Model*. url=https://online.stat.psu.edu/stat415/lesson/7/7.4.

[4] Marco Taboga. *Mean squared error of an estimator*. Online appendix. 2021. URL: `https://www.statlect.com/glossary/mean-squared-error` (visited on 04/07/2026).