


# Linear Algebra and Calculus - Introduction



[\\_.\(https://github.com/learn-co-curriculum/dsc-linear-algebra-and-calculus-intro\).](https://github.com/learn-co-curriculum/dsc-linear-algebra-and-calculus-intro)   
(<https://github.com/learn-co-curriculum/dsc-linear-algebra-and-calculus-intro/issues/new/choose>)

## Introduction

In this section, we're going to take a step back to learn some of the basics of the math that powers the most popular machine learning models. You may not need deep knowledge of linear algebra and calculus just to build a scikit-learn model, but this introduction should give you a better understanding of how your models are working "under the hood".

## Linear Algebra

Linear Algebra is so foundational to machine learning that you're going to see it referenced many times as the course progresses. In this section, the goal is to give you both a theoretical introduction and some computational practice, solving a real-life problem by writing the code required to solve a linear regression using OLS.

We're going to kick this section off by looking at some of the many places that linear algebra is used in machine learning - from deep learning to natural language processing (NLP) to dimensionality reduction techniques such as principal component analysis (PCA).

## Systems of Linear Equations

We then start to dig into the math! We look at the idea of linear simultaneous equations - a set of two or more equations each of which is linear (can be plotted on a graph as a straight line). We then see how such equations can be represented as vectors or matrices to represent such systems efficiently.

## Scalars, Vectors, Matrices, and Tensors

In a code along, we'll introduce the concepts and concrete representations (in NumPy) of scalars, vectors, matrices, and tensors - why they are important and how to create them.

## Vector/Matrix Operations

We then start to build up the basic skills required to perform matrix operations such as addition and multiplication. You will also cover key techniques used by many machine learning models to perform their calculations covering both the Hadamard product and the (more common) dot product.

# Solving Systems of Linear Equations Using NumPy

We then bring the previous work together to look at how to use NumPy to solve systems of linear equations, introducing the identity and inverse matrices along the way.

## Regression Analysis Using Linear Algebra and NumPy

Having built up a basic mathematical and computational foundation for linear algebra, you will solve a real data problem - looking at how to use NumPy to solve a linear regression using the ordinary least squares (OLS) method.

## Computational Complexity

In the last linear algebra lesson, we look at the idea of computational complexity and Big O notation, showing why OLS is computationally inefficient, and that a gradient descent algorithm can instead be used to solve a linear regression much more efficiently.

## Calculus and Gradient Descent

Next, you'll learn about the mechanism behind many machine learning optimization algorithms: gradient descent! Along the way, we'll also look at cost functions and will provide a foundation in calculus that will be valuable to you throughout your career as a data scientist.

Just as we used solving a linear regression using OLS as an excuse to introduce you to linear algebra, we're now using the idea of gradient descent to introduce enough calculus to both understand and have good intuitions about many of the machine learning models that you're going to learn throughout the rest of the course.

## An Introduction to Calculus and Derivatives

We're going to start off by introducing derivatives - the "instantaneous rate of change of a function" or (more graphically) the "slope of a curve". We'll start off by looking at how to calculate the slope of a curve for a straight line, and then we'll explore how to calculate the rate of change for more complex (non-linear) functions.

## Gradient Descent

Now that we know how to calculate the slope of a curve - and, by extension, to find a local minimum (low point) or maximum (high point) where the curve is flat (the slope of the curve is zero), we'll look at the idea of a gradient descent to step from some random point on a cost curve to find the local optima to solve for a given linear equation. We'll also look at how best to select the step sizes for

descending the cost function, and how to use partial derivatives to optimize both slope and offset to more effectively solve a linear regression using gradient descent.


## Summary

It is possible to use machine learning models such as scikit-learn without understanding the underlying math, but a basic understanding of this math will help you to make the most informed choices about which models to use based on your particular project context.

How do you feel about this lesson?



Have specific feedback?

**Tell us here!**  (<https://github.com/learn-co-curriculum/dsc-linear-algebra-and-calculus-intro/issues/new/choose>)