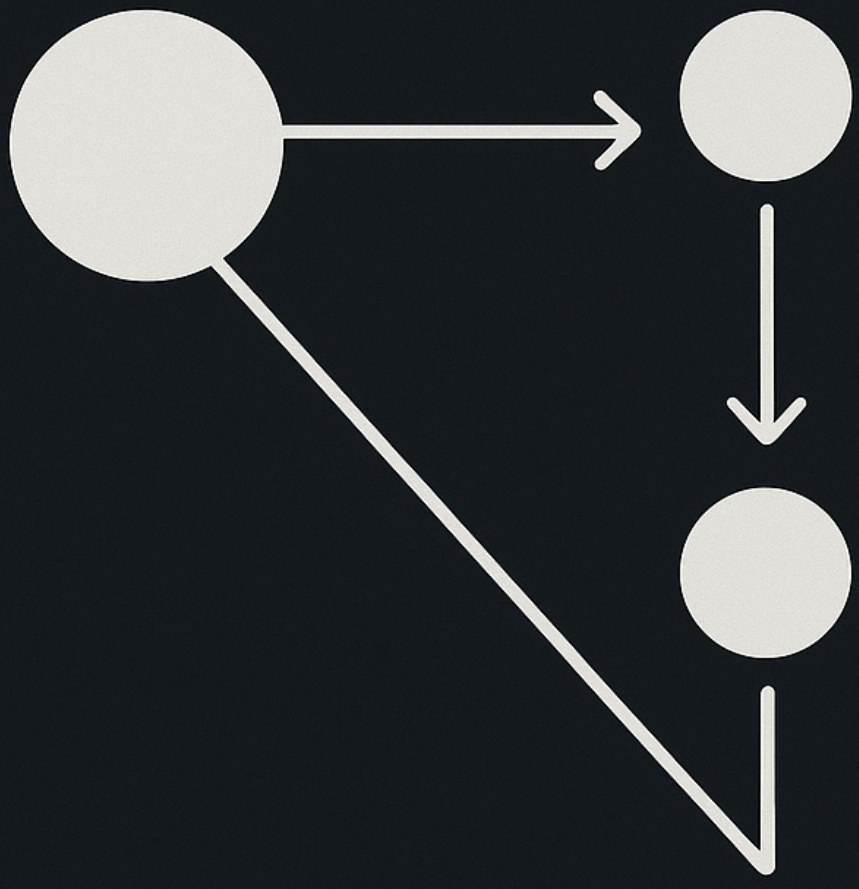
SUPERVISED LEARNING

FROM FIRST PRINCIPLES



AMIT GHARAT

Show code

Author

Amit Gharat is a seasoned technology leader and hands-on engineer with deep expertise in frontend architecture, full-stack development, and modern software systems. With over two decades of experience spanning startups and large-scale enterprises, Amit combines engineering precision with a passion for clarity and design. His work often bridges the gap between user experience and robust system design, ensuring performance, scalability, and developer delight.

When he's not building or teaching, Amit enjoys spending time with his family, including his energetic 4-year-old daughter, and tinkering with new ideas that merge creativity, AI, and engineering craftsmanship. He can be reached on https://www.linkedin.com/in/amitgharat

Reviewer(s)

Peter Beardsmore is Head of AI & Machine Learning for SS&C GIDS UK. Responsible for a small team of ML engineers and developers at SS&C Technologies, applying his background in data analytics, data engineering and machine learning in the financial services industry. The team delivers an AI innovation programme, creating proof-of-concepts and progressing them to internal automation solutions in-production. He can be reached on https://www.linkedin.com/in/peter-beardsmore

Show code

Preface

Six months ago, I knew nothing about Machine Learning since I had always viewed Machine Learning as a distant, almost mystical field reserved for mathematicians and data scientists. Then one day, curiosity got the better of me. I wanted to know, **how does a machine actually learn?**

That simple question sent me down a rabbit hole. I read countless books, watched hours of videos, and wrestled with concepts that stretched my understanding of math, statistics, and algorithms. The journey was humbling. There were moments when I felt utterly lost, and others when a concept finally clicked and lit up my brain like a light bulb.

This book was born out of that experience: the struggle, the joy, and the sheer curiosity that kept me going. It's written with pure love for understanding how the hell machine learning actually works from the ground up. Note that I'm not an expert in Machine Learning or Artificial Intelligence and by the end of this book, you won't be one either. That's not the goal. What this book aims to do is something more fundamental: to help you see a drop from the ocean of ML knowledge through the lens of first principles, much like understanding physics through its equations. By the time you finish, you'll have the intuition, mathematical grounding, and confidence to explore more advanced algorithms and concepts on your own, but with true understanding. We'll explore the underlying mathematics, visualize concepts through graphs, derive algorithms step by step, and eventually bring them to life with Python.

The goal is not to rush to the latest buzzwords or neural networks, but to peek under the hood of machine learning to see how equations turn into intelligent behavior.

As a full-time frontend architect, I'm writing and releasing this book chapter by chapter, at a pace that fits around my day job and life. So, think of this as a journey we're taking together, one concept at a time.

Thank you for joining me. Let's figure out, together, how machines learn.

— Amit Arun Gharat

November, 2025

Show code

Chapter 1: Simple Linear Regression

What is Machine Learning

You're already a great software engineer, writing **explicit rules** (if/else statements, complex functions) to solve problems. But what happens when the rules are impossible to write? That's when you pivot to ML. As an ML engineer, you will write code to make predictions or generate content. However, before running it, you must provide it with large amounts of data so it can learn patterns and mathematical relationships within that data. This process is called **Machine Learning**.

Hence, the combination of the code and the learned patterns forms **the model**, which you can deploy to make predictions on new data and evaluate whether it is performing as expected.

As you can see, it is not magic - you still have to write code and design algorithms effectively, which requires knowledge of mathematics, data management, evaluation metrics, and, of course, programming.

When to use Machine Learning?

Now that you know what "Machine Learning" is, you might wonder when to use it. The three points below will help you decide for sure:

1. **Sufficient Data**

   The most important rule. Just like industrial machines need oil to run, Machine Learning models need data. Without enough data, the model cannot learn patterns effectively and will not perform well.

2. **Hard to define rules**

Even with plenty of data, some problems are too complex for manual rules. For example, you can not simply tell a robot, "a cat has two ears and a tail", because many animals share those traits. Instead, we show the robot many pictures of cats so it can learn those features on its own.

3. **Existence of Patterns**

   The data should contain meaningful trends. For example, in weather prediction, "cloudy and humid" often means rain — a pattern the model can learn. On the other hand, boring a well in a land without water will lead you to nothing. There is no pattern in flipping coins. It can land heads or tails, but each flip is random and hence unpredictable.

By now, you might be eager to build your first model, but hold your horses! *If it's not magic then it's science.*

Before we start, we need to understand the different ways we can teach a machine to recognize patterns.

Types of Machine Learning

There are several ways to teach machines. If we group them by how the *learning* happens, they fall into four main categories:

1. **Supervised Learning**: The model learns from labeled data to predict outcomes for new, unseen data.
2. **Unsupervised Learning**: The model discovers hidden patterns or groupings in unlabeled data without predefined outputs.
3. **Semi-supervised Learning**: The model learns from a small amount of labeled data combined with a large amount of unlabeled data to improve accuracy.
4. **Reinforcement Learning**: The model learns by interacting with an environment and receiving rewards or penalties to optimize decision-making.

In this book, we'll explore **Supervised Learning** step by step, using both theory and code as though we are inventing Machine Learning from scratch.

Are you excited?? I indeed heard a big "YES" so let's begin.

Supervised Learning

Imagine you are teaching table-setting etiquette to your child: "Plate in the middle, fork on the left, and spoon on the right." After many dinners, the child notices the pattern — no

matter how the plates or forks look, they always go in the same spots.

One day, you surprise them: "Tonight we have different plates and a new Peppa Pig spoon and fork. Please set the table." Even though the items are new, the child applies what they learned before and sets the table correctly.

This is an example of **Supervised Learning**. Here, your child **(a model)** was trained on table-setting etiquette **(the training data)** with clear guidance on where to place each item **(features and targets)** over many days **(epochs)**. Eventually, figured out where to put what **(convergence)**. When tested with new items **(the test data)**, the child/model still performed well. This means your child/model has successfully learned what you were trying to teach.

In short, a supervised ML system is trained on data where the correct answers are known, and it learns to identify the patterns that produce those answers. We will focus on the two most common types of supervised learning:

- Regression
- Classification

In everyday English, "regression" can mean returning to **a less developed state**. Think of it like this: saying "India won the 1985 Cricket World Cup" is a fact about the past — **a developed state** and unchangeable. But saying "India will win the 2085 World Cup" is a less developed state or future (which may or may not happen), hence **a prediction**. By the same token, any kind of problems requiring prediction of a continuous numerical value will fall under Regression.

Can you think of examples around you? Compare them with mine below — if they match, you’re on the right track in understanding regression.

Practice what you preach, they say so I looked around and came up with a few real-world examples:

1. Predicting your weight loss after following a diet for a certain number of weeks.
2. Predicting your monthly electricity bill based on past usage and the number of appliances in your home.
3. Predicting your child will pass or not pass an exam based on study hours and past performance.
4. Predicting whether a bank loan application will be approved or rejected based on the applicant’s profile.
5. Predicting the price of a house from its size, location, and number of rooms.
6. Predicting whether a patient has diabetes or not based on medical test results.

For instance, imagine you reduce your daily calorie intake from 3,000 to 1,500 for a month and lose about 1 kg. If you reduce it further to 1,000 calories per day, you might lose about 2 kg. Here, different diet plans produce different weight-loss predictions. This is simply

predicting a continuous numerical value. In statistics, the process of estimating relationships between variables (calorie intake and weight-loss in this case) to make such predictions is called **Regression**.

If you look back at the earlier examples, some involve a different kind of prediction — for instance, deciding whether your child will pass or fail an exam based on study hours and past performance. Here, the outcome is not a continuous number; it's **a category or a class**: "pass" or "fail." In statistics, predicting categories is called **Classification**.

> Try spotting the regression/classification problems from the list above and see if you can identify them correctly before looking at the below table.

Problem	ML System
Predicting your weight loss	Regression
Predicting your monthly electricity bill	Regression
Predicting your child will pass or not pass	Classification - Pass/Fail
Predicting whether a bank loan application will be approved or rejected	Classification - Approved/Rejected
Predicting the price of a house	Regression
Predicting whether a patient has diabetes	Classification - Yes/No

Regression

Now that you can tell the difference between various machine learning problems, it's time to roll up your sleeves and dive into the maths, graphs, and code that bring regression models to life. Knowing all of these will help you debug a model that's performing poorly. So let's pick a problem first.

Most Indian cities struggle with poor air quality lately. What could be a better regression problem than predicting the Air Quality Index (AQI)? Usually, there are many factors why AQI worsens, but for our understanding, let's blame just one culprit - PM2.5 particle in the air. You might recall the news every winter about Delhi's pollution crisis, often triggered by a record spike in PM2.5 levels. It's clear that as PM2.5 levels rise, AQI worsens, and as they fall, AQI improves. This hints at a strong relation between PM2.5 levels and AQI.

At the end of the excercise, you will end up with a **Simple Linear Regression** model/blackbox to which you can feed an air quality data in India (2015-2020) from one end for it to predict AQI from other end. However, before arriving on the solution, we will learn/explore the Machine Learning concepts along the way.

Mathematical Foundations