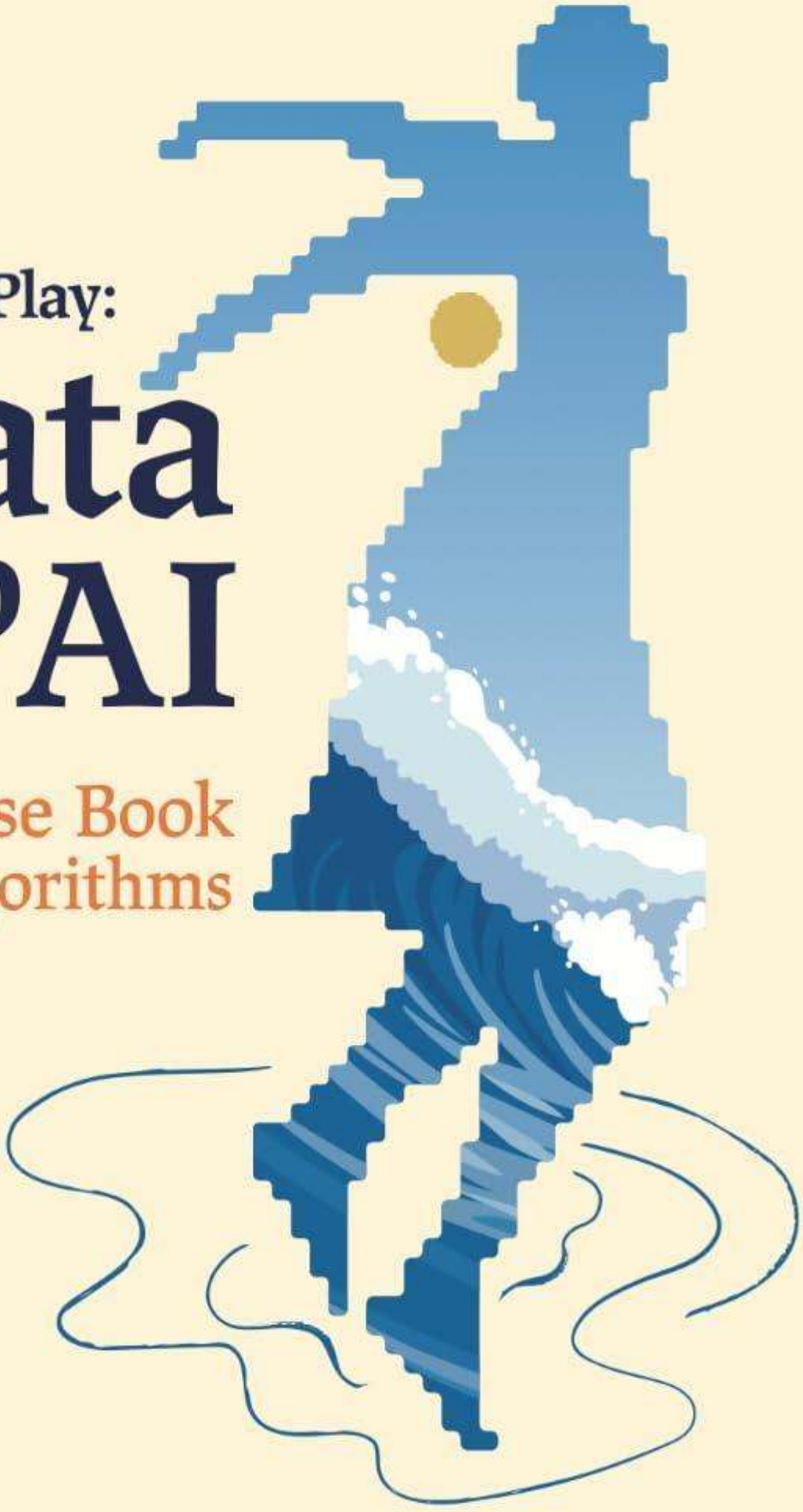
Call to Play:

Data to AI

An Exercise Book on Algorithms



01
What is Machine Learning?

Story

Usually, little people are the first to notice big things. But on this summer evening, Appu was the last to notice the torrential rain tearing through his town.

He had been sprawled across the cool, red oxide floor, clutching the freshly looted crayons from his sister's bag. He was sketching a solar system of monsters with a toothy vampire shining at the centre.

Then the power died. Just like that. The way important things usually do.

Amma balanced a candle on an upturned steel tumbler beside him. This little wobbly lighthouse toppled whenever a strong west wind blew.

Thankfully, a pregnant mango tree had stood to the west of his house. Thud, thud, thud! Falling mangoes meant shielding the candle with his small palms.

At first, Appu's timing was all wrong. Sometimes he cupped his hand, but the wind never came. At times, he was too late. The flame disappeared before he heard a sound. And once, mangoes dropped without any wind. Appu's patterns were not perfect, but they worked most of the time during the storm.



Storms are patient teachers of resilience, and observant children are good students. Soon, his palm knew exactly when to rise, curve, and protect.

Appu learned to predict when his candle would blow out from the patterns of air movement.

Similarly, machine learning algorithms find patterns in data and make predictions based on these patterns. Each algorithm uses its own unique method to spot these patterns.

Predictions can be numbers, text, images, audio, and more. YouTube, for example, uses our viewing history to recommend what to watch next. We do not manually program what we want to watch, and when; that would be impossible. Instead, algorithms build their own understanding from thousands of examples and predict what we would like to see.

After training on huge volumes of data, algorithms make predictions. They check if their predictions are right or wrong and then adjust them to improve. The idea is to electronically mirror what we do organically, finding patterns and meaning through observation, trial, and error.

Machine learning is one way machines learn from experience instead of following fixed rules. But it is only one part of a larger idea called artificial intelligence, or AI.

AI is the broader goal of building machines that can perform tasks we usually associate with human intelligence, such as recognising images, understanding language, making decisions, or planning actions.

Some AI systems rely on rules written by humans, while others, like machine learning systems, learn their behaviour from data.

In this book, we will begin by understanding how machines learn from data. We will then gradually widen our view to explore how these learning systems come together to form what we now call AI.

From Story to Science

Machine learning makes predictions based on patterns found in data.

Once a tech novelty, it is now everywhere. From the face unlock that greets us before the sun does to social media networks, large language models, voice assistants, and translation apps, machine learning is omnipresent.

While traditional programs follow explicit instructions, machine learning algorithms learn patterns from examples and improve with experience.

Understanding these algorithms is not about becoming a data scientist.

It is about understanding the hidden force behind every application and “suggested for you” recommendation. It is about looking under the hood of the AI that answers questions like “How do I talk to my father-in-law?” or “Why is my printer giving this error?”

Complex models such as neural networks need large datasets. However, there are simpler models that work with modest data. The quality of the dataset matters as much as the quantity.

These algorithms inherit and amplify biases in their training data.

A hiring algorithm trained on historical data will only repeat past patterns. If past hiring favoured a particular demographic, then the algorithm will do the same.

Some models operate as “black boxes.” Their decision-making processes are difficult to interpret.

Origins

We have always dreamt of thinking machines, and at times, we have been haunted by them. From myths of walking clay men to revenge-planning metallic robots, the idea of teaching machines to learn began in the 1940s. This was the year when Alan Turing asked a question that would define a field: "Can machines think?"

We will explore the history of these creators and their digital children in the next chapter, where we have outlined the key milestones in the evolution of AI.

Call to Play:

Sound Prediction

Machine learning is about learning patterns from past data to make predictions about new, unseen situations.

This short data sketching activity will help you feel the essence of machine learning. It is not a technical breakdown of ML algorithms. It is a simple way to experience the concept of learning and predicting through a familiar, everyday experience. An example is shown below. Draw your version in the space on the right.

Find a spot, maybe your balcony, a park bench, or beside a window. This is your data collection station.

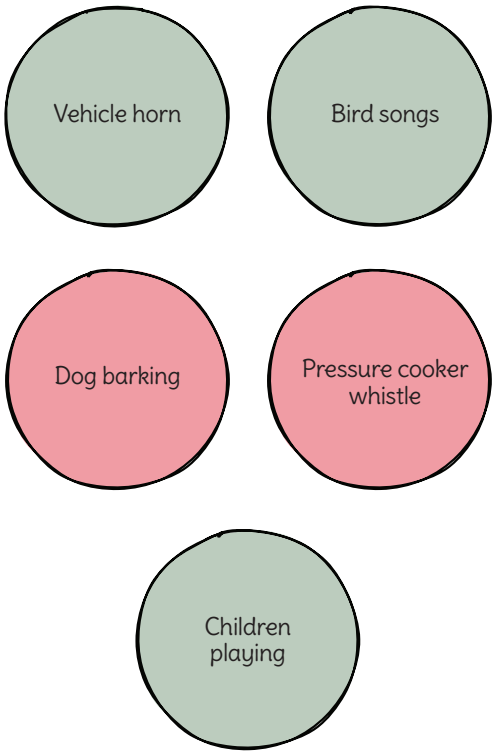
Collect

Set a 2-minute timer. Close your eyes and listen carefully. This is your training phase. For each sound you hear, note down the following in the form of a table as given below:

1. The sound you heard.
2. How often did you hear it? Mark 1 for rare (once), 2 for occasional (2-3 times), 3 for frequent (continuous/ repeating).

Sound	Frequency
Vehicle horn	2
Bird songs	3
Dog barking	1
Pressure cooker whistle	1
Children playing	2

Draw



Prediction accuracy = 3/5 = 60%

1. Based on your “training data,” pick 5 sounds you expect to hear in the next 3 minutes. Write each prediction in a circle.
2. Set a 3-minute timer. Listen. This is your testing phase. Shade correct predictions green and wrong ones red.
3. Count your correct predictions. This is your accuracy score out of 5.

? Were there any rare sounds that surprised you in the test phase?

In ML, rare events can be hard to predict.

make your own

We invite you to try this exercise from a different location or at a different time of day. Notice how your learned patterns change. Models depend on the data they see.

Call to Play:

Colour the Flowchart

Given below is an overview of the typical machine learning workflow. It starts with defining a problem and ends with a working model.

Along the way, we collect data, train a model, test it, and refine it until it performs well.

Even though this process seems linear, it rarely is. Just like in life, we may need to take a U-turn, revisit previous steps, and make adjustments.

Machine learning is as much about iteration as it is about algorithms.

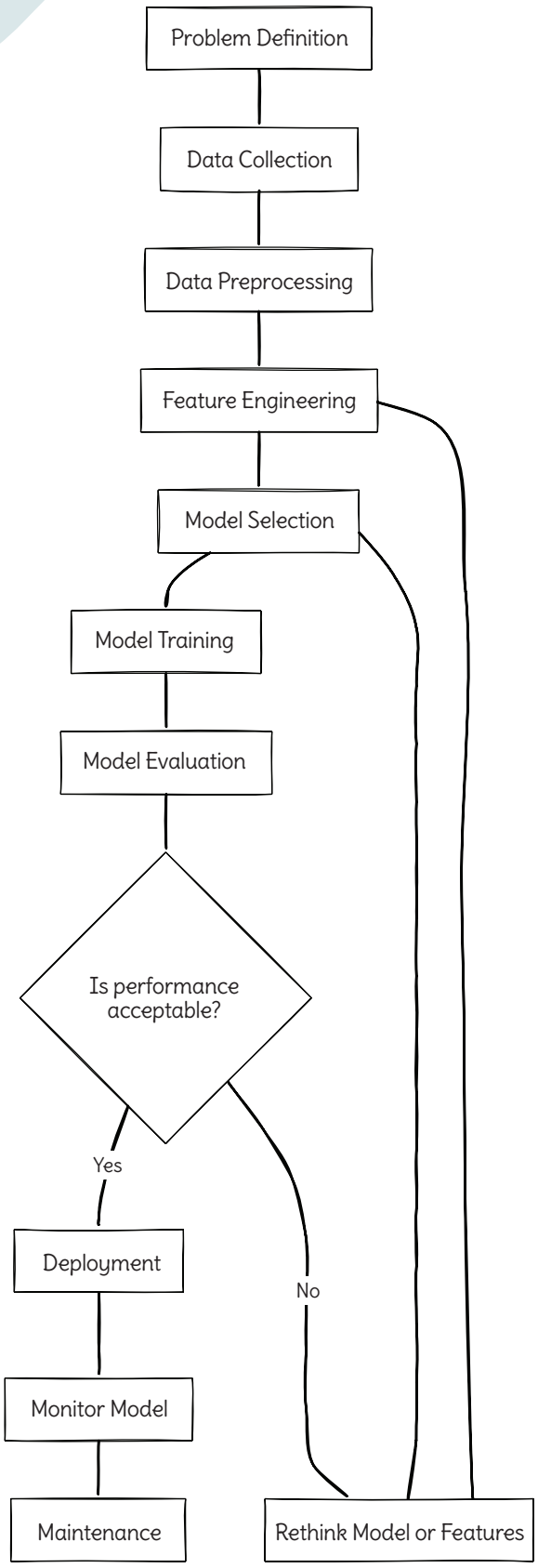
1. We first define what we want to predict: house prices? Is an email spam or not?
2. Then we collect and clean the data.
3. We choose an algorithm and train it on a portion of our data. Then we test the trained model on the remaining data that it has not seen before.
4. If it works well, we deploy the model. However, we keep a close eye on its performance.
5. If it does not work well, we go back and try different approaches until it does.

This flow is just the big picture. The finer details will click into place as you work through the coding exercises in the Create section at the end of this chapter.

Go through the flowchart and colour it using a colour code. Blue for process steps, orange for decision diamonds, green arrows for “yes” paths, and red arrows for “no” paths. Purple for special elements like aggregations or distinctions. For output types, use green for classification and yellow for regression. Or if you would like to use a coding system of your own, please do so. Colouring as you learn helps your brain build stronger connections with the material.

Call to Play:

Draw Parallels



How do you learn a new skill?

Say you want to learn to make the perfect cup of chai. First, you decide what "perfect" means to you. Should it be more or less milky? How strong should it be? How sweet should it be? You collect recipes from family, YouTube, and experiment in the kitchen. You try different combinations of tea, sugar, and milk, and brew them for different times. You taste each attempt and ask others for feedback. Some attempts fail. You adjust and try again. Finally, you lock in the recipe and make it the same way every day.

In the space provided on the next page, draw your learning process and compare it with how machine learning works.

Do not worry about matching every box one-to-one. Instead, find where your journey follows similar methods. Our goal here is simply to help you connect new information to your existing mental frameworks. This helps improve comprehension and retention.

Here is one mapping to get you started: deciding what "perfect" means to you is Problem Definition. Now find the rest yourself. Where is Data Collection in your chai brewing journey? What step matches Model Training? When does Evaluation happen? What does Deployment look like?

make your own

If you are confused about how the process works, watch the videos suggested in the "**Explore More**" section first, and then come back to this activity.

Call to Play:

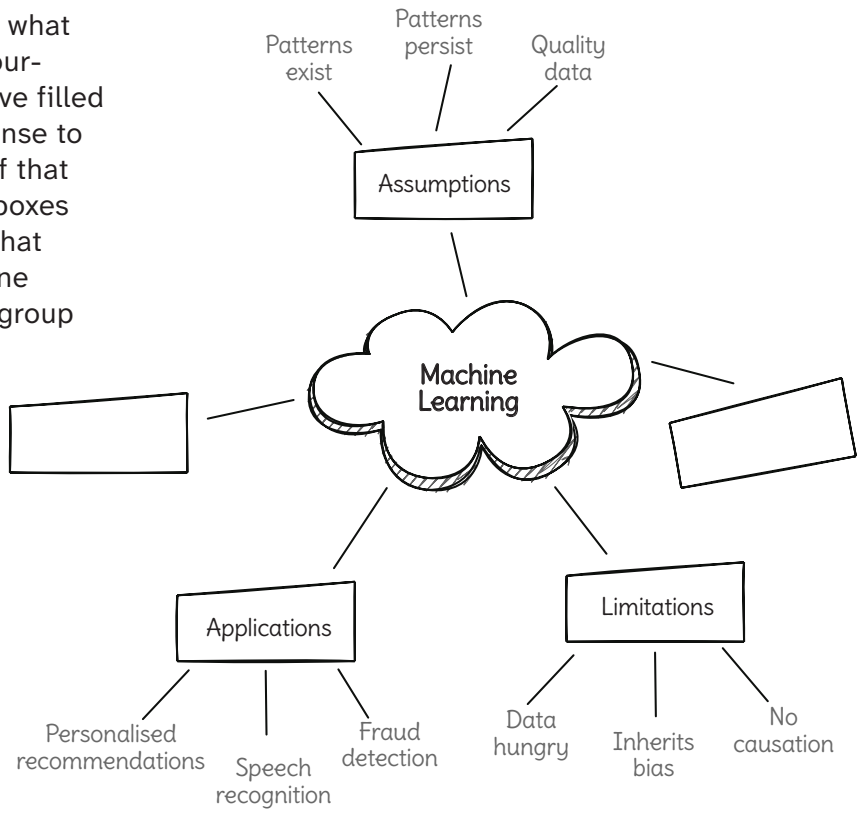
No Code Machine Learning

Go to Google's Teachable Machine: **teachablemachine.withgoogle.com**. Experience training, testing, and overfitting without writing any code.

Call to Play:

Build the Mind Map

Let us use the space below to create a mind map based on what you have learned so far. Colour-code the three groups we have filled in. Pick colours that make sense to you, or skip colours entirely if that is not your style. The empty boxes are yours to name and fill. What else stands out about machine learning? Maybe you want a group for "Definitions", "Key Ideas", or something we have not thought of.



Make this map your own.

Explore More

True understanding comes from the cycle of thinking and doing. The projects in later sections will teach you what no amount of reading can. In the meantime, if you need more explanation about how this concept works, the curated sources below will help. Grab your notebook and take active notes through sketches, concept maps, the slip-box method, or whatever works for you.

For the Visual Learner	Search for "**A Gentle Introduction to Machine Learning**" by **StatQuest with Josh Starmer** on **YouTube**. His cheery and visual explanations and clear visuals make these often-confusing ideas click.
For the Interactive Learner	Visit **r2d3.us** and explore their visual introduction to machine learning. It uses animated scrolling graphics to show how a model learns to classify data. It is a beautiful, hands-on way to see ML in action.
For the Reader	Search for "**Machine Learning for Everyone**" by **Vas3k** online. It is a comprehensive illustrated blog post that walks through the concept with playful analogies and friendly hand-drawn concept maps.

Call to Play:

Your First Colab Notebook

Before we dive into coding, let us understand our playground. Research what Google Colab is. It is an amazing tool for anyone who wants to learn machine learning but does not have access to a powerful computer. Find a beginner's guide that explains how to get started. We recommend the YouTube video, "**How to Use GOOGLE COLAB | Google Colab Tutorials for Beginners | GeeksforGeeks**". Try working out the sample code mentioned in the video. You will be amazed that something this powerful is completely free.

Call to Play:

Project 1

Let us see machine learning in action using generated code and play with it. Open a new Google Colab notebook and enter the prompt below. First, run each cell one by one. Pay attention to the print statements. Then, take a step back and trace the flow from start to finish. Finally, play. Try a different dataset. Restructure the code. This is your sandbox.

Help me understand machine learning through code. Build a model that predicts if numbers are even or odd using their patterns. Train it with 20 examples, test with 5 new numbers, and show how it learns this simple rule. Add beginner-friendly print statements.

Call to Play:

Explore Kaggle

Kaggle is like a playground where data scientists compete, share datasets, and learn from each other. Create a free account at **kaggle.com** and spend 15 minutes exploring two sections: Datasets (where people share a variety of datasets) and Competitions (where companies post real problems with prize money).

Spend time clicking around to understand how Kaggle works. Find and follow a few Grandmasters to see how experts work. Don't feel overwhelmed by their expertise. They all started exactly where you are today, clicking around Kaggle, trying to figure out what all this means. Let's take it one day at a time.

Call to Play:

Your First GitHub Repository

GitHub is a tool where you can connect and collaborate with coders all around the world. It lets you show your work, follow others, contribute to projects, and build a professional identity.

Watch the freeCodeCamp "**Git and GitHub for Beginners - Crash Course**" on **YouTube** and follow along to create your first repository.

Think of it as setting up your coding diary where all your learning experiments will be stored. Soon, as you go through each chapter, you will be proudly sharing your projects. Don't worry if Git commands feel strange now. You are not supposed to master everything in one go. Just get a feel for what GitHub is and create that first repository for now. The real understanding comes from using it chapter by chapter, project by project, mistake by mistake.

Call to Play:

Teach What You Learned

Think of the questions you had when you first heard about machine learning. What confused you? What excited you? Create an FAQ document answering the ten most common questions a beginner might have. Or record a simple two-minute video explaining the concept to a friend or family member who has never heard of it.

Share it as a blog post, video, or social thread.

To share your creation with us, tag us @rowrowlearn or DM us on Instagram.

25
Generative Adversarial Network

Story

The art of cracking exams is an art and a science that only a few manage to master. Supriya was one of them. Never not the topper. She graduated from college with a GPA so perfect that her neighbour aunties' bindis practically combusted with envy.

Supriya had a passion for coding. When she entered the workforce, she easily evolved to become a 10x coder.

However, no one really knew Supriya in the office. She was usually huddled at the corner table, typing away with her paper cup with a teabag inside. She came in and went out with a stoic expression, making no eye contact with anyone. Everyone just assumed she was reserved.



One day, during a Friday evening huddle, the HR tried a team-building exercise. She brought a bag full of names. She would pick a name from the bag, and that person would ask a team member to perform something.

To Supriya's bad luck, the first name out was her manager's. Because it was Friday and he was

in a rather jolly mood, he wickedly asked Supriya to sing a song.

"No. Absolutely not. I cannot sing," she said.

"Oh, c'mon, you can't let the team down like this!"

"I cannot sing, but I can act. Can I do that?"

That was the moment Supriya transformed from zombie coder to star performer.

She imitated famous characters: Amitabh Bachchan, Naseeruddin Shah, and even Maggie Smith. The team loved it, but they also started testing her. "Do Rajinikanth!" someone shouted.

Her first attempt fell flat. The pitch was wrong, the cadence off. But she went home, incorporated the feedback, and practised. Her character impersonations became a regular feature at office parties.

Each time she performed, the team became better critics, and she became a better mimic. After months of this playful competition, even the harshest critics agreed she sounded exactly like the characters.

Similarly, GANs focus on two main parts working together. One part creates data (the generator), and the other part tries to determine if the created data is real or fake (the discriminator). They are trained in a continuous loop and compete with each other.

As they compete, the generator gets better at creating data that looks very much like the real thing, based on continuous feedback from the discriminator on what needs improvement. This process helps GANs create synthetic data that looks real.

From Story to Science

We use GANs when we need synthetic data that mimics real data.

GANs can generate fake faces, create digital art, improve blurry photos, make deepfakes, and produce synthetic voices.

They have been used in healthcare (generating medical imaging like X-rays and MRIs), data augmentation for training machine learning models, privacy-preserving data synthesis, fashion design, music creation, and video scene generation.

This algorithm consists of two neural networks: a generator (which creates synthetic data from random noise) and a discriminator (which tries to spot real vs. fake data).

Training consists of these two components competing back and forth.

As training progresses, ideally, the generator gets better at fooling the discriminator, while the discriminator gets smarter at detecting counterfeits.

GANs assume their adversarial training will eventually balance out, with the generator creating convincing fakes.

They face unique challenges like unstable training, mode collapse (where the generator makes limited samples), and sensitivity to hyperparameter settings. Innovations like Wasserstein GANs, Conditional GANs, and StyleGANs aim to solve these issues.

GANs excel at generating images and audio but struggle with text because words are discrete units rather than continuous values.

While some specialised GANs (like TextGAN) can generate short text or captions, large language models (like Transformers) are far more effective for long-form, coherent text generation.

These algorithms learn complex patterns but can also learn and amplify biases from their training data.

Origins

Ian Goodfellow invented GANs in 2014 at the University of Montreal. The idea came to him during a night out with friends at a bar, and he coded the initial prototype that night. It worked.

GANs use two neural networks that compete with each other to improve their output. The networks automatically learn features from data, eliminating the need for manual feature engineering.

Today, GANs drive applications in entertainment, art, medicine, and data privacy. They have also sparked vital conversations about deepfakes, synthetic media, and AI ethics.

"This Person Does Not Exist" is an online demonstration where every refresh produces a new, photorealistic face created by a GAN. It showcases both the creative power and the ethical questions raised by these technologies.

Call to Play:

Celebrity Doppelgangers

A GAN has two parts: a Generator that creates fakes and a Discriminator that judges them. Through competition, both improve until the fakes become hard to spot.

This short data sketching activity will help you feel the essence of a GAN. It is not a technical breakdown of how the algorithm works. It is a simple way to feel the concept through a familiar, everyday experience. An example is shown below. Draw your version in the space on the right.

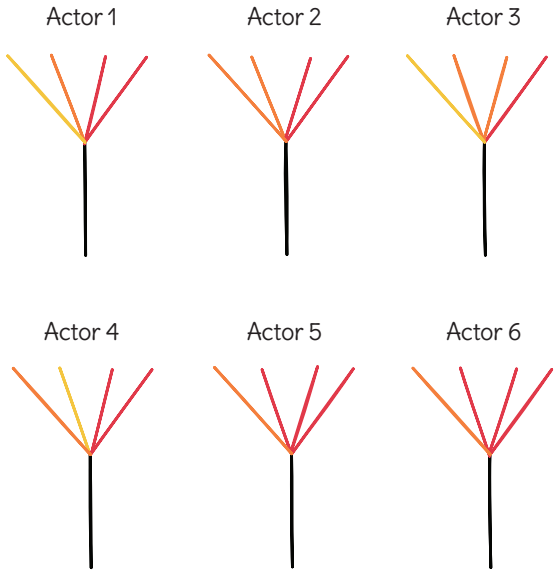
Choose 6 of your favourite celebrities. Search for each name followed by "lookalike" or "doppelganger." This is your dataset.

Collect

1. For each celebrity, look at the first 4 image results.
2. Rate each lookalike: 1 = barely similar, 2 = pretty similar, 3 = almost fooled you.

Celebrity	Result 1	Result 2	Result 3	Result 4
Actor 1	3	2	1	1
Actor 2	2	1	1	1
Actor 3	3	2	2	1
Actor 4	2	3	1	1
Actor 5	2	1	1	1
Actor 6	2	1	1	1

Draw



1. For each celebrity, draw a black vertical line as the base.
2. From the top, draw 4 lines radiating outward like a starburst.
3. Colour each line based on your rating: yellow for 3, orange for 2, red for 1.
4. Write the celebrity's name above.

? Look at your 6 starbursts. Which has the most yellow? Which has the most red?

The yellow lines are the best fakes. These are the ones that almost fooled you.

You just played the Discriminator. In a GAN, the Generator starts creating terrible fakes (all red). It learns from feedback and improves. Eventually, some fakes become so convincing (yellow) that even the Discriminator struggles to tell the difference.

make your own

Call to Play:

Colour the Flowchart

Given below is the training process where GANs learn to generate realistic synthetic data through adversarial competition. During training, the generator learns to create convincing fake data while the discriminator learns to distinguish real from fake.

Once training is complete and the models have reached equilibrium, the trained generator can create new, realistic data. The generator produces outputs that are closely similar to the training data.

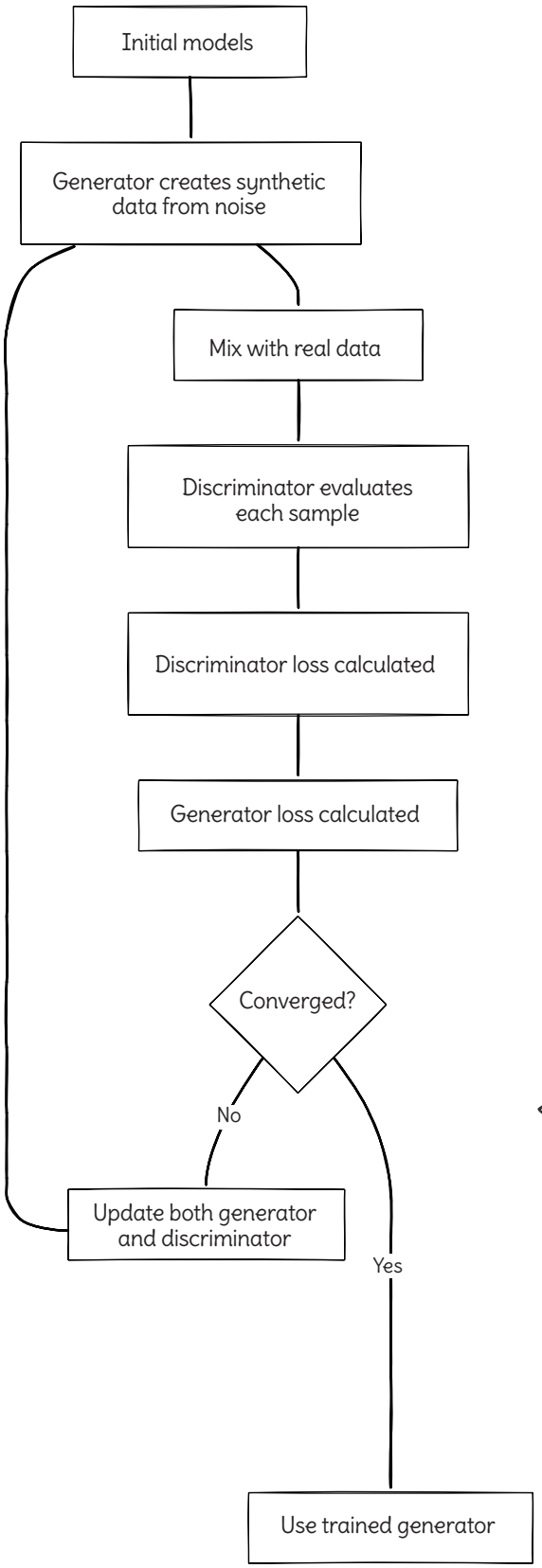
1. Set up two networks with random weights. The generator creates fake data. The discriminator judges real versus fake.
2. The generator creates fake data from random noise. It tries to mimic the characteristics of real training data.
3. Combine the generated fake samples with real samples from the training dataset. This creates a mixed batch for evaluation.
4. The discriminator examines each sample and predicts whether it is real or fake. It outputs probability scores.
5. Compute two losses. The discriminator loss measures how well it detects real versus fake. The generator loss measures how well it fooled the discriminator.
6. Check if the models have reached equilibrium. At equilibrium, the generator creates realistic data, and the discriminator can barely tell the difference.
7. If not at equilibrium, adjust the weights of both networks through backpropagation. The discriminator learns to judge better. The generator learns to create better fakes.
8. Return to step 2 and continue until the generator produces high-quality synthetic data.

This is just the big picture. The finer details will click into place as you work through the coding exercises at the end of this chapter.

Go through the flowchart and colour it using a colour code. Blue for process steps, orange for decision diamonds, green arrows for “yes” paths, and red arrows for “no” paths. Purple for special elements like aggregations or distinctions. For output types, use green for classification and yellow for regression. Or if you would like to use a coding system of your own, please do so. Colouring as you learn helps your brain build stronger connections with the material.

Call to Play:

Draw Parallels



How does a student learn to draw realistically?

Say a drawing teacher is teaching a student to draw a realistic elephant. The student starts with a rough attempt. The proportions are off; the ears are too small. The teacher glances at it and immediately says, "Not realistic." The student studies what went wrong. It is the proportions. She tries again. Better proportions this time, but the skin texture looks flat. The teacher catches it, and the student improves. Round after round, the student gets better at drawing. But the teacher also raises the bar, pointing out finer and finer details. Each pushes the other. Over many rounds, the student draws elephants so realistically that even the teacher has to look twice.

In the space provided, draw this process and compare it with how GANs work.

Do not worry about matching every box one-to-one. Instead, find where your journey follows similar methods. Our goal here is simply to help you connect new information to your existing mental frameworks. This helps improve comprehension and retention.

Here is one mapping to get you started: the student drawing an elephant is Generator Creates Synthetic Data. Now find the rest yourself. What does the teacher evaluating the drawing represent? Where is Discriminator Evaluates Each Sample? What are the two losses being calculated? When do both the student and teacher Update? What does Converged look like in this learning journey?

make your own

If you are confused about how the algorithm works, watch the videos suggested in the "**Explore More**" section first, and then come back to this activity.

Call to Play:

Code Generation

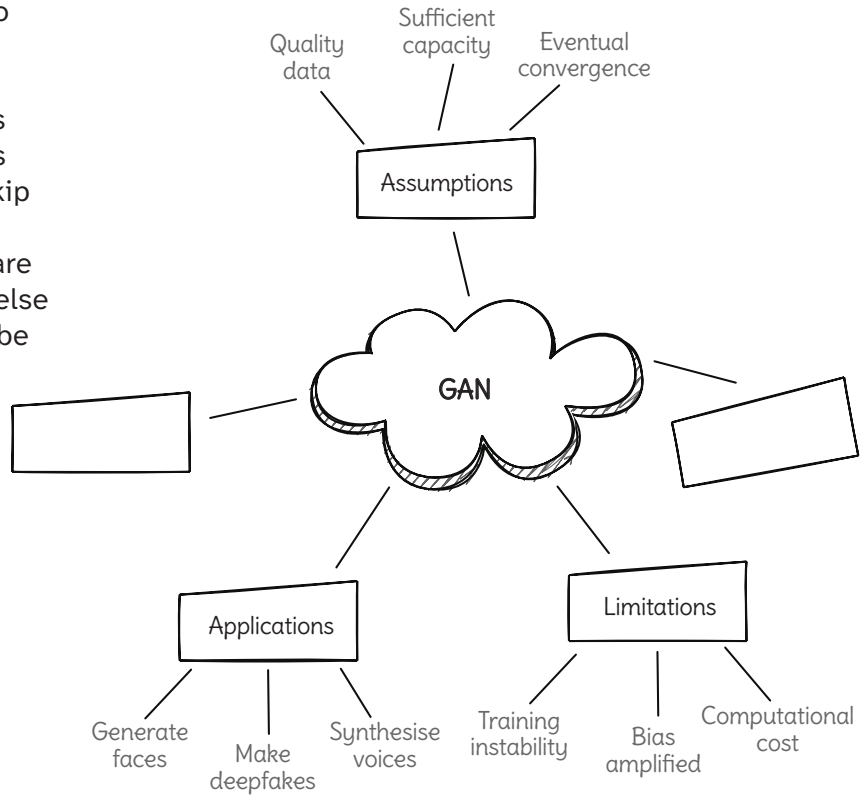
Let us see GAN in action using generated code and play with it. Open a new Google Colab notebook and enter the prompt below. First, run each cell one by one. Pay attention to the print statements. Then, take a step back and trace the flow from start to finish. Finally, play. Change parameters. Restructure the code. Try a different dataset. This is your sandbox.

Help me understand GANs through code. Build a generator and a discriminator to produce data matching a bell curve. Show how generated data improves over training and plot loss curves for both networks. Add beginner-friendly print statements.

Call to Play:

Build the Mind Map

Let us use the space below to create a mind map based on what you have learned so far. Colour-code the three groups we have filled in. Pick colours that make sense to you. Or skip colours entirely if that is not your style. The empty boxes are yours to name and fill. What else stands out about GANs? Maybe you want a group for "Definitions", "Key Ideas", or something we have not thought of.



Make this map your own.

Explore More

True understanding comes from the cycle of thinking and doing. The projects in later sections will teach you what no amount of reading can. In the meantime, if you need more explanation about how this concept works, the curated sources below will help. Grab your notebook and take active notes through sketches, concept maps, the slip-box method, or whatever works for you.

For the Visual Learner	Search for "The Math Behind Generative Adversarial Networks Clearly Explained!" by Normalized Nerd on YouTube. This video walks through how the generator and discriminator compete against each other, with clear explanations of the mathematics that make GANs work.
For the Interactive Learner	Explore GAN Lab (poloclub.github.io/ganlab). Watch the generator and discriminator train in real-time as they compete. One creates fake samples, the other tries to spot them. Adjust the learning rate and see how it affects the balance between the two networks.
For the Reader	Read "Generative Adversarial Networks" by Goodfellow et al. (2014). This is the original paper that introduced GANs and sparked a revolution in generative AI. Upload the PDF to Google NotebookLM and generate a podcast episode.

Call to Play:

Project 1

Use the Credit Card Fraud Detection dataset from Kaggle. Build a GAN that generates synthetic transaction data to balance the dataset. Create 100 synthetic fraud transactions, compare their feature distributions with real fraud cases using histograms, and explain how this synthetic data could help train better fraud detection models. Share your project on GitHub.



Call to Play:

Project 2

Build a DCGAN (Deep Convolutional GAN) that generates anime faces or handwritten digits. Use the Anime Faces dataset from Kaggle or MNIST. Train your GAN and save generated images every 100 epochs to create a time-lapse that shows how output quality improves during training. Document the challenges you faced, such as mode collapse, training instability, and how you addressed them. Share your project and training time-lapse on GitHub.



Call to Play:

Teach What You Learned

Draw a four to six-panel comic illustrating the generator and discriminator as characters in competition. Maybe they are rival artists or competing chefs. Show their initial adversarial relationship gradually coming to an equilibrium. One gets better at creating while the other gets better at judging. Eventually, both reach excellence. Use your comic to explain the technical aspects of GANs.

Share as a comic, slideshow, or animated video.

To share your creation with us, tag us @rowrowlearn or DM us on Instagram.

Frequently Asked Interview Questions

1. What is mode collapse in GANs, and how can you address it?
2. Why are GANs difficult to train, and what are some techniques to stabilise training?
3. Explain the difference between GANs and VAEs (Variational Autoencoders) for generating data.