

Study Delight Publications



Big
Ideas
Made
Simple

AI & ML

Learn
Practice
Build
Grow

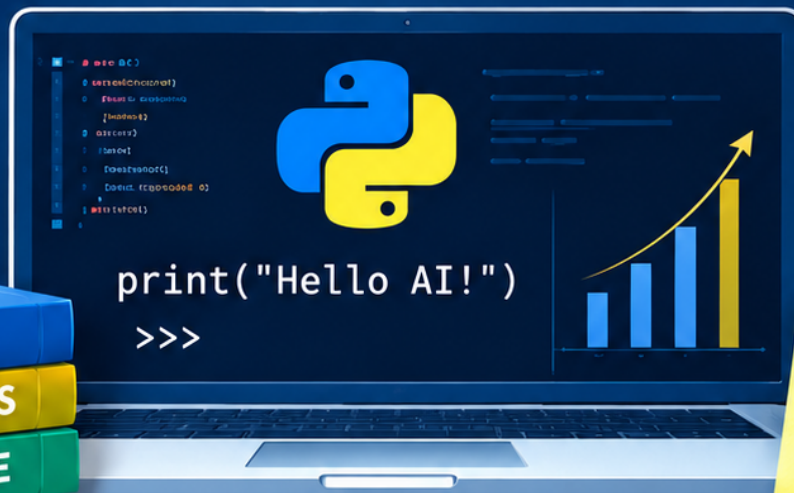
Simplified

PART 1

Foundations & Essentials



AI
Basics



Think
Like a
Data Scientist

Small
Steps
Big Future



Clear
Explanations



Practical
Examples



Real-World
Use Cases



Exam Ready



Build Strong
Fundamentals

pythonforaiml.com

PART 1 • AI & ML FOUNDATIONS

Understand the language, workflow and thinking behind machine learning.

30-PAGE ROADMAP



01 What is Artificial Intelligence?

16 Exploratory Data Analysis

02 AI, Machine Learning & Deep Learning

17 Correlation

03 Where AI is Used

18 Bias & Variance

04 Traditional Programming vs ML

19 Overfitting

05 Types of Machine Learning

20 Underfitting

06 Supervised Learning

21 Model Evaluation

07 Unsupervised Learning

22 Regression vs Classification

08 Reinforcement Learning

23 Loss / Cost / Objective

09 Data: The Raw Material

24 Parameters vs Hyperparameters

10 Features & Target

25 From Problem to ML Project

11 Dataset Anatomy

26 ML Workflow

12 Train / Validation / Test

27 Ethics, Bias & Fairness

13 Data Leakage

28 Foundation Map

14 Data Preprocessing

29 Final Review

15 Handling Missing Values

30 Master Checklist

LEARN • PRACTICE • BUILD • GROW

Use this index as your quick navigation map while studying.

PART 1 • AI & ML FOUNDATIONS

Start here: understand the ideas before the algorithms

ARTIFICIAL
INTELLIGENCE

MACHINE
LEARNING

DEEP
LEARNING

THEORY

Artificial Intelligence sounds complex because it covers many technologies. But the central idea is simple: build systems that perform tasks that normally require some form of human intelligence. Machine learning is one way to build such systems by learning patterns from examples instead of writing every rule by hand.

EXAMPLE / DIAGRAM

AI — Machine Learning — Deep Learning — Generative AI This book moves through this hierarchy step by step.

OUTPUT / RESULT

AI is the broad field. ML is a major approach inside AI. Deep learning is a family of ML methods based on neural networks.

CONCEPT NOTE

Write one everyday example that you think uses AI.

PRACTICE

Understand the map ☒ learn one concept ☒ build one example

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

WHAT IS ARTIFICIAL INTELLIGENCE?

AI is about intelligent behavior, not magic

THEORY

Artificial Intelligence is the field of creating computer systems that can perform tasks involving abilities such as perception, language understanding, reasoning, planning, prediction or decision-making. An AI system does not need to think exactly like a person. It needs to accomplish a useful task using computational methods.

EXAMPLE / DIAGRAM

Examples: • Voice assistants understand speech. • Recommendation systems rank content. • Computer vision systems analyze images. • Planning systems choose actions. • Generative models create text or images.

OUTPUT / RESULT

The word "intelligence" describes the capability being reproduced, not necessarily human consciousness.

CONCEPT NOTE

List three AI systems you interact with and identify what task each performs.

PRACTICE

Task ☒ capability ☒ system

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

AI IS NOT THE SAME AS ROBOTICS

AI is software intelligence; robotics adds a physical body

THEORY

Robotics deals with machines that sense and act in the physical world. AI can be part of a robot, but AI also exists entirely in software. A recommendation engine has no physical body, yet it can use AI. Separating these ideas helps you understand the scope of the field.

EXAMPLE / DIAGRAM

Robot vacuum: Sensors → perception → decision → motor action
Recommendation system: User data → model → ranking → screen

OUTPUT / RESULT

Both can involve intelligent decision-making, but only one necessarily has a physical machine.

CONCEPT NOTE

Give one example of AI without robotics and one example of robotics without advanced AI.

PRACTICE

Separate the task from the hardware

STUDY LOOP: Understand → code → test → explain

AI • ML • DEEP LEARNING

Three terms you must keep separate

THEORY

AI is the umbrella term. Machine learning focuses on algorithms that learn patterns from data. Deep learning uses multi-layer neural networks and has become especially important for images, speech, language and generative models. These terms overlap, but they are not interchangeable.

EXAMPLE / DIAGRAM

Traditional program: Rules + data → output
Machine learning: Data + desired answers → learned model
Deep learning: Data + neural network training → learned representation/model

OUTPUT / RESULT

A model is not simply "AI code." It is a learned computational object produced through a training process.

CONCEPT NOTE

Explain AI, ML and deep learning in your own words without using the definitions above.

PRACTICE

Define ☒ compare ☒ explain

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

RULE-BASED PROGRAMS

Not every intelligent-looking program is machine learning

THEORY

A rule-based program follows instructions written by a developer. If the rules cover the problem well, this approach can be excellent. But some problems contain too many exceptions or depend on patterns that are difficult to describe manually. ML becomes useful when examples can teach the system those patterns.

EXAMPLE / DIAGRAM

```
if temperature > 30: advice = "Drink water" else: advice = "Normal conditions" print(advice)
```

OUTPUT / RESULT

The developer explicitly wrote the decision rule.

CONCEPT NOTE

Ask: can I describe the behavior reliably with rules, or would examples be better?

PRACTICE

Rules ✕ limits ✕ decide

STUDY LOOP: Understand ✕ code ✕ test ✕ explain

WHY MACHINE LEARNING?

Learn patterns when rules are difficult to write

THEORY

Machine learning is useful when the relationship between inputs and outputs is complex, changing, or difficult to encode manually. We provide examples and an algorithm searches for parameters that make predictions useful. The quality of learning depends on data, problem formulation, model choice and evaluation.

EXAMPLE / DIAGRAM

Email text → ML model → spam / not spam
Historical examples: text + known label → training process → learned model

OUTPUT / RESULT

The model learns a relationship from examples; it does not automatically understand the world like a human.

CONCEPT NOTE

Choose a task where manually writing thousands of rules would be inconvenient.

PRACTICE

Problem → examples → learning

STUDY LOOP: Understand → code → test → explain

TYPES OF MACHINE LEARNING

Three major learning styles



THEORY

The three classic categories are supervised learning, unsupervised learning and reinforcement learning. The distinction is based largely on what kind of feedback the learning system receives. Modern systems can combine ideas from these categories, but these foundations remain essential.

EXAMPLE / DIAGRAM

Supervised: inputs + known answers Unsupervised: inputs without known answers Reinforcement: actions + rewards/penalties

OUTPUT / RESULT

Supervised learning learns from labeled examples. Unsupervised learning looks for structure. Reinforcement learning learns through interaction and feedback.

CONCEPT NOTE

Match: customer grouping, house-price prediction, game-playing agent.

PRACTICE

Identify the feedback type

STUDY LOOP: Understand ✕ code ✕ test ✕ explain

SUPERVISED LEARNING

Learning from labeled examples

Features
Area, rooms

Known label
Price

Learning
Model

New input
Prediction

THEORY

In supervised learning, each training example contains inputs and a known target. The algorithm learns a mapping from inputs to targets. Regression predicts numbers; classification predicts categories. Many practical business prediction problems fit this pattern.

EXAMPLE / DIAGRAM

Example: Area = 1200, Rooms = 2 → Price = 60 lakh
Area = 1800, Rooms = 3 → Price = 90 lakh
Features → Target

OUTPUT / RESULT

The target is the answer the model learns to predict.

CONCEPT NOTE

Identify features and target for predicting whether a customer will renew a subscription.

PRACTICE

Inputs → target → model

STUDY LOOP: Understand → code → test → explain

REGRESSION

Predict a number

THEORY

Regression problems have numeric targets. The predicted value may be a price, temperature, demand, revenue or score. A regression model does not promise an exact future value; it estimates a value based on patterns in training data.

EXAMPLE / DIAGRAM

Features: area, bedrooms, location score Target: house price Model
→ predicted price

OUTPUT / RESULT

If the actual price is 75 and the model predicts 72, the prediction error is 3 units.

CONCEPT NOTE

Write two more real-world regression problems.

PRACTICE

Numeric target → prediction → error

STUDY LOOP: Understand → code → test → explain

CLASSIFICATION

Predict a category

THEORY

Classification assigns an example to one of a set of classes. Binary classification has two classes, while multiclass classification has more than two. The labels can be numbers internally, but the meaning of each class must be clear.

EXAMPLE / DIAGRAM

Message → model → spam Customer → model → low / medium / high risk
Image → model → cat / dog / other

OUTPUT / RESULT

Classification is about categories, even if the computer represents those categories using numbers.

CONCEPT NOTE

Create one binary and one multiclass classification example.

PRACTICE

Classes ☒ examples ☒ decision

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

UNSUPERVISED LEARNING

Find structure without known answers

THEORY

In unsupervised learning, the training data does not provide a target label for every example. Algorithms can search for groups, patterns or compact representations. Clustering is a common example. The challenge is interpreting whether the discovered structure is actually useful.

EXAMPLE / DIAGRAM

Customer data: age, spending, visits Clustering → Group A Group B Group C

OUTPUT / RESULT

The algorithm can discover groups, but a human still needs to interpret what those groups mean.

CONCEPT NOTE

Imagine grouping customers. List three features you might use.

PRACTICE

Data ☒ structure ☒ interpretation

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

REINFORCEMENT LEARNING

Learn through actions and feedback

THEORY

Reinforcement learning involves an agent interacting with an environment. The agent chooses actions and receives rewards or penalties. Over many interactions, it learns a policy that aims to achieve better long-term outcomes. This is different from simply learning a fixed labeled dataset.

EXAMPLE / DIAGRAM

Agent → action → environment ↑ ↓ ┌── reward ──┐

OUTPUT / RESULT

Examples include game-playing, control and sequential decision problems.

CONCEPT NOTE

Describe a simple game where an agent could receive rewards for useful actions.

PRACTICE

State ⊠ action ⊠ reward

STUDY LOOP: Understand ⊠ code ⊠ test ⊠ explain

DATA • THE RAW MATERIAL

Machine learning starts with information

THEORY

Data may come from databases, files, sensors, applications, surveys, images, text or APIs. Before choosing an algorithm, understand the data. Ask where it came from, what each column means, how it was collected and whether the examples represent the real situation where predictions will be used.

EXAMPLE / DIAGRAM

Possible dataset: `customer_id age monthly_spend visits renewed`
Rows = examples Columns = variables

OUTPUT / RESULT

Good modeling cannot rescue a fundamentally unsuitable dataset.

CONCEPT NOTE

Pick a prediction problem and list five data fields you would want.

PRACTICE

Source ☒ meaning ☒ quality

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

FEATURES

What the model uses as input

THEORY

Features are measurable attributes supplied to the model. They can be numeric, categorical, text, image-derived or other representations. A good feature should be available at the time a prediction is made and should carry useful information about the target.

EXAMPLE / DIAGRAM

To predict house price: $X = [\text{area}, \text{bedrooms}, \text{age}, \text{location_score}]$
These are features.

OUTPUT / RESULT

Features are often represented as X in machine-learning notation.

CONCEPT NOTE

Give four possible features for predicting restaurant demand.

PRACTICE

Input $\rightarrow$ representation $\rightarrow$ usefulness

LABEL / TARGET

What the model tries to learn

THEORY

The target is the answer we want to predict in supervised learning. It is commonly written as y . Defining the target precisely is critical. If the target is vague or measured incorrectly, even a technically excellent model may solve the wrong problem.

EXAMPLE / DIAGRAM

$X = [\text{area}, \text{bedrooms}, \text{age}]$ $y = \text{price}$ $X \rightarrow \text{model} \rightarrow \text{predicted } y$

OUTPUT / RESULT

A target can be numeric for regression or categorical for classification.

CONCEPT NOTE

Define X and y for employee-attrition prediction.

PRACTICE

Define the answer before choosing the algorithm

TRAINING DATA

Examples used to learn

THEORY

Training data is the portion of labeled data used to fit model parameters. During training, the algorithm adjusts parameters to reduce an objective or loss. Training accuracy alone is not enough: a model can memorize training examples and perform poorly on new data.

EXAMPLE / DIAGRAM

Training examples: 1. inputs → known answer 2. inputs → known answer 3. inputs → known answer ... Model learns parameters from these examples.

OUTPUT / RESULT

Training is the learning phase, not the final proof that the model works.

CONCEPT NOTE

Explain why more training examples can help a model generalize.

PRACTICE

Learn ☒ don't memorize

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

TEST DATA

The final exam for the model

Dataset

ALL EXAMPLES

TRAIN • 80%

TEST • 20%

THEORY

Test data is held back from model fitting and used to estimate performance on unseen examples. Keeping it separate protects the evaluation from becoming another training exercise. In a careful project, the test set is touched only after the modeling decisions are largely finalized.

EXAMPLE / DIAGRAM

Training set → fit model Test set → evaluate model The test examples should behave like future unseen cases.

OUTPUT / RESULT

A good test set represents the deployment situation as realistically as possible.

CONCEPT NOTE

Explain why repeatedly tuning a model on the test set can make the test score unreliable.

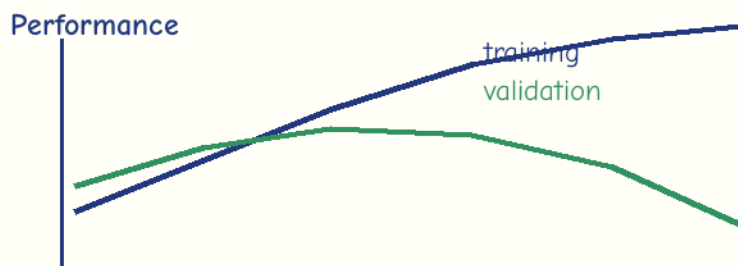
PRACTICE

Hold out ☒ predict ☒ measure

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

VALIDATION DATA

A second layer for model selection



THEORY

When many model choices or hyperparameters are tested, a validation set can help compare them before the final test. Cross-validation is another common strategy. The main idea is to avoid using the final test set as a tuning playground.

EXAMPLE / DIAGRAM

Train → learn parameters Validation → choose settings Test → final estimate

OUTPUT / RESULT

The exact splitting strategy depends on dataset size and problem type.

CONCEPT NOTE

Draw a train/validation/test diagram and explain each role.

PRACTICE

Tune ☒ freeze ☒ test

OVERFITTING

When a model learns the training data too closely

THEORY

Overfitting happens when a model captures noise or accidental details in the training examples and therefore performs poorly on new examples. A large gap between training and validation performance can be a warning sign. More data, simpler models, regularization and careful validation can help.

EXAMPLE / DIAGRAM

Training performance: 99% Validation performance: 68% Possible warning: the model has learned details that do not generalize.

OUTPUT / RESULT

High training performance is not automatically good machine learning.

CONCEPT NOTE

Give one analogy for overfitting, such as memorizing practice questions instead of learning the topic.

PRACTICE

Train score $\nrightarrow$ real-world score

STUDY LOOP: Understand $\nrightarrow$ code $\nrightarrow$ test $\nrightarrow$ explain

UNDERFITTING

When the model is too simple to capture useful patterns

THEORY

Underfitting occurs when a model fails to learn enough structure from the data. Both training and validation performance may be poor. The solution may involve better features, a more expressive model, less restrictive regularization or improved training.

EXAMPLE / DIAGRAM

Training: 62% Validation: 60% Both are weak → investigate underfitting or data quality.

OUTPUT / RESULT

The goal is not maximum complexity. The goal is useful generalization.

CONCEPT NOTE

Compare underfitting and overfitting in two sentences.

PRACTICE

Too simple ☒ too complex

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

GENERALIZATION

The real goal of machine learning

THEORY

Generalization means performing well on new examples that were not used to fit the model. This is the central reason we separate training and evaluation data. A model that works only on historical examples is not useful for future decisions.

EXAMPLE / DIAGRAM

Past examples → learn pattern New example → apply pattern Good model: works reasonably well on both.

OUTPUT / RESULT

Generalization is more important than memorization.

CONCEPT NOTE

Describe what “generalizes well” means without using the word memorize.

PRACTICE

Learn pattern ☒ handle new cases

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

LOSS FUNCTION

Turn prediction mistakes into a number

THEORY

A loss function measures how wrong a model's predictions are during training. The learning algorithm uses this signal to adjust model parameters. Different tasks use different losses. Understanding the concept is more important at first than memorizing formulas.

EXAMPLE / DIAGRAM

Actual: 100 Predicted: 92 A loss function converts the difference into a numerical training signal.

OUTPUT / RESULT

Lower loss generally means predictions are better according to the chosen objective, but the loss must be appropriate for the problem.

CONCEPT NOTE

Why does an ML algorithm need a numerical objective to optimize?

PRACTICE

Predict → measure error → adjust

STUDY LOOP: Understand → code → test → explain

MODEL PARAMETERS

What training learns

THEORY

Parameters are values learned from data. Their meaning depends on the model. In a simple linear model, parameters can represent coefficients and an intercept. Training searches for parameter values that produce useful predictions according to the objective.

EXAMPLE / DIAGRAM

$y = w_1 * x + b$ w_1 and b are learned parameters. Training searches for useful values.

OUTPUT / RESULT

Parameters are learned; hyperparameters are settings chosen before or around training.

CONCEPT NOTE

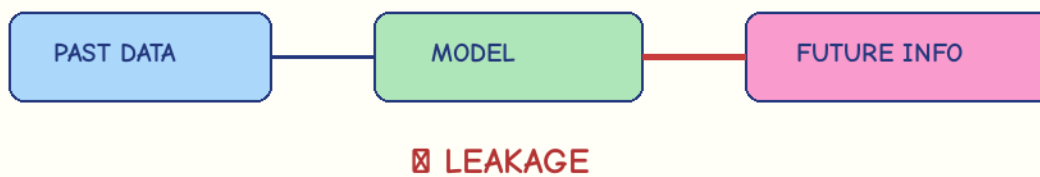
Explain the difference between a parameter and a hyperparameter.

PRACTICE

Learned values vs chosen settings

HYPERPARAMETERS

Settings chosen outside the learning process



THEORY

Hyperparameters control aspects of model training or structure. Examples include tree depth, learning rate, number of neighbors or regularization strength. They are not learned in the same direct way as model parameters. Validation or cross-validation can help select them.

EXAMPLE / DIAGRAM

Example: Decision tree depth = 5 K in KNN = 7 Learning rate = 0.001
These are configuration choices.

OUTPUT / RESULT

Changing a hyperparameter can change both performance and generalization.

CONCEPT NOTE

Name two hyperparameters you have heard of and research what each controls.

PRACTICE

Choose ❌ validate ❌ freeze

STUDY LOOP: Understand ❌ code ❌ test ❌ explain

DATA QUALITY

Garbage in can become unreliable predictions

THEORY

Data quality includes correctness, completeness, consistency, relevance and representativeness. Duplicates, incorrect labels, missing values, measurement errors and biased sampling can all affect a model. Data preparation is often one of the largest parts of a real ML project.

EXAMPLE / DIAGRAM

Check: • missing values • duplicates • impossible values • inconsistent categories • incorrect labels • unusual distributions

OUTPUT / RESULT

A model can discover patterns in bad data too. Learning does not automatically mean learning truth.

CONCEPT NOTE

Create a five-item checklist for inspecting a new dataset.

PRACTICE

Inspect ☒ clean ☒ validate

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

DATA LEAKAGE

A hidden shortcut that makes evaluation look better

THEORY

Data leakage occurs when information that would not be legitimately available at prediction time influences training. Leakage can produce excellent-looking evaluation scores that collapse in real use. Always ask when each feature becomes available and whether preprocessing was fitted using only appropriate data.

EXAMPLE / DIAGRAM

Bad example: Using `"final_payment_status"` to predict whether the customer will eventually pay. The feature contains future information.

OUTPUT / RESULT

A feature can be technically correlated with the target yet still be invalid if it is only known after the prediction time.

CONCEPT NOTE

Invent one possible leakage example in a loan or sales problem.

PRACTICE

Ask: was this information available then?

BIAS & FAIRNESS

Models can reproduce problems in data

THEORY

AI systems can reflect biases in historical data, labels, measurement processes and deployment decisions. Fairness is not solved by a single formula. Teams need to understand who is affected, what errors occur, whether important groups are represented and how decisions will be monitored.

EXAMPLE / DIAGRAM

Historical data ↓ patterns + missing representation ↓ model ↓ predictions
Data quality and decision context matter.

OUTPUT / RESULT

A model can be accurate on average while still performing differently across groups.

CONCEPT NOTE

Why should an AI project consider affected users, not only model accuracy?

PRACTICE

Measure ☒ inspect ☒ question

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

ETHICS & RESPONSIBLE AI

Technical skill includes responsible use

THEORY

Responsible AI involves privacy, safety, transparency, security, fairness and appropriate human oversight. The right question is not only “Can we build it?” but also “Should we use it this way?” and “What happens when it is wrong?” High-impact applications need stronger safeguards and review.

EXAMPLE / DIAGRAM

Before deployment ask: What data? Who is affected? What can go wrong? How are errors handled? Can humans review important decisions?

OUTPUT / RESULT

Responsible design is part of engineering, not an optional final paragraph.

CONCEPT NOTE

Write three questions you would ask before deploying an AI system that affects people.

PRACTICE

Build ☒ evaluate ☒ govern

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

THE ML WORKFLOW

A repeatable process beats random experimentation

THEORY

A useful machine-learning workflow is problem definition, data collection, exploration, cleaning, feature preparation, splitting, model training, evaluation and deployment/monitoring. The steps are iterative. Poor results may send you back to the data or problem definition rather than simply to a more complex algorithm.

EXAMPLE / DIAGRAM

Problem ↓ Data → Clean → Features ↓ Train → Validate → Test ↓
Deploy → Monitor → Improve

OUTPUT / RESULT

The workflow connects the concepts from the previous pages into one mental model.

CONCEPT NOTE

Draw this workflow from memory and explain each arrow.

PRACTICE

Problem ☒ data ☒ model ☒ evidence ☒ action

STUDY LOOP: Understand ☒ code ☒ test ☒ explain

FIRST AI/ML PROJECT

Start small and finish the loop

THEORY

Your first project should be small enough to understand completely. A student-score predictor, simple customer classifier or sales forecasting exercise can teach the full workflow. Focus on clean data, a clear target, a baseline, evaluation and an explanation of results rather than chasing the most advanced model.

EXAMPLE / DIAGRAM

Project plan: 1. Define question 2. Find/create dataset 3. Inspect data 4. Choose X and y 5. Split data 6. Train baseline 7. Evaluate 8. Explain limitations

OUTPUT / RESULT

A finished small project is more valuable for learning than an unfinished "AI super project."

CONCEPT NOTE

Choose your first project and write its question, features, target and success metric.

PRACTICE

Define ☒ build ☒ measure ☒ explain

STUDY LOOP: Understand ☒ code ☒ test ☒ explain