Introduction to Deep Learning (I2DL)

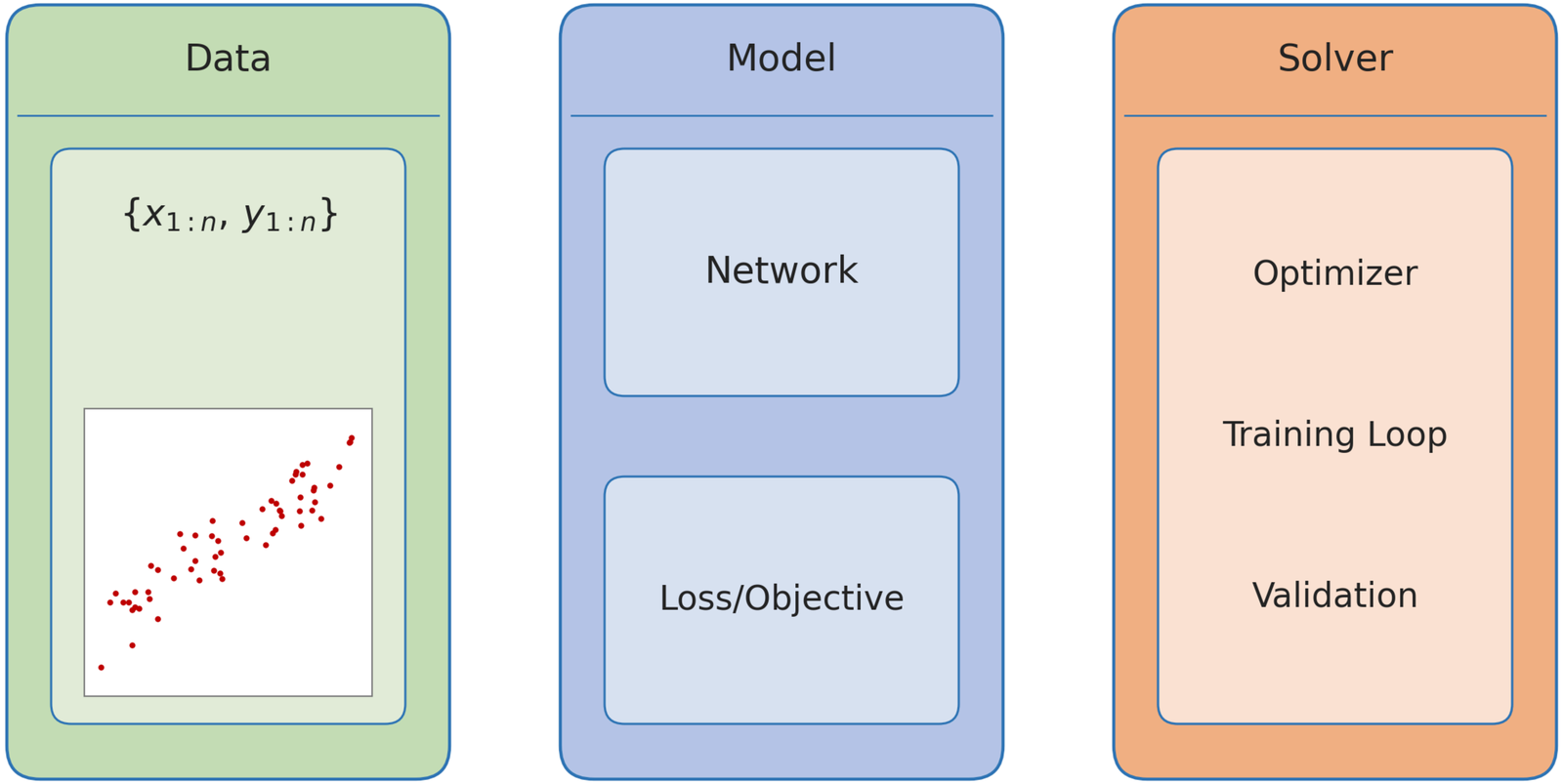
Tutorial 3: Data

Today's Outline

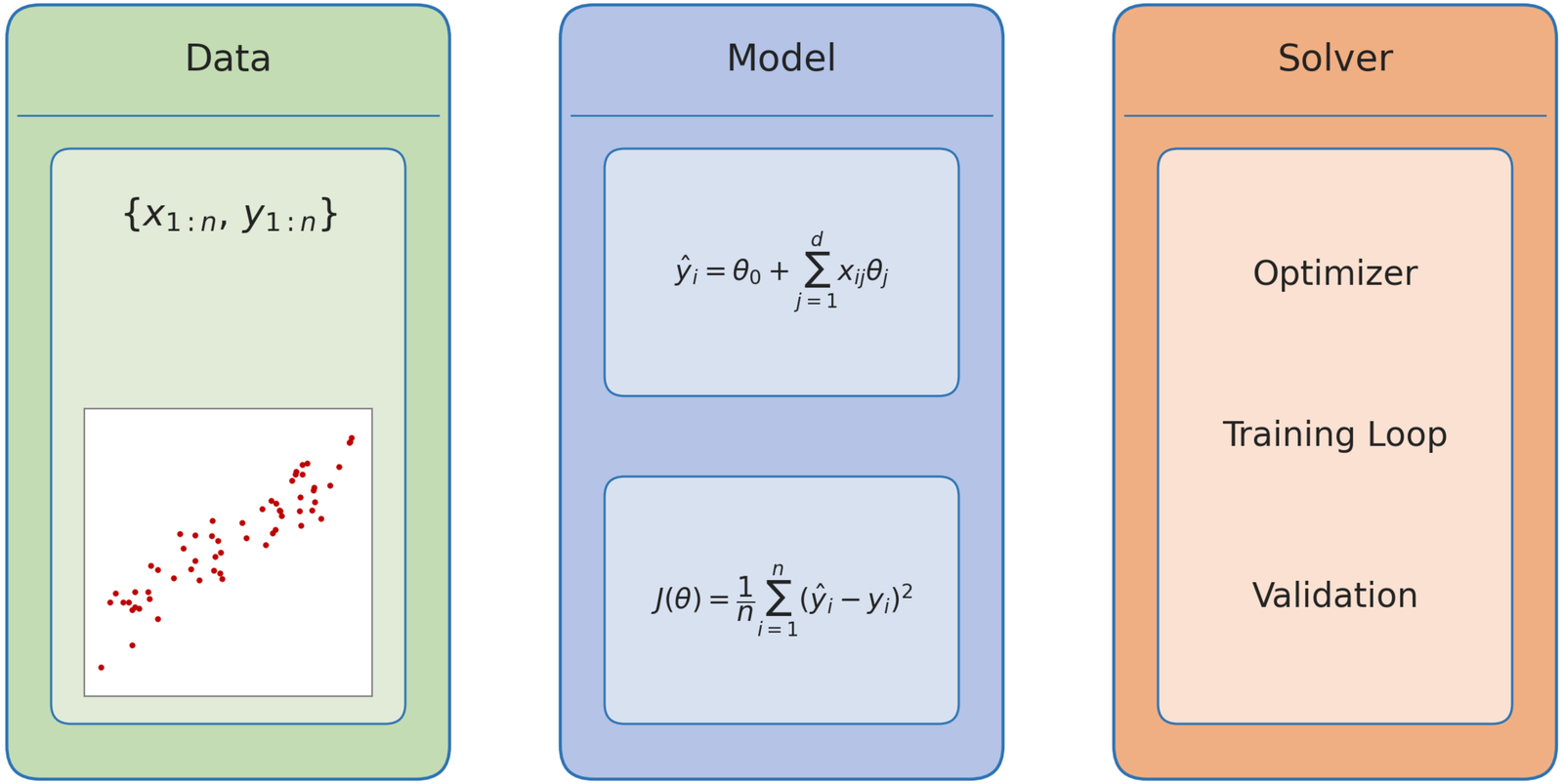
- **Exercise outline**
  - Pillars of Deep Learning
  - Reinvent the wheel
- **Contents of Exercise 3**
  - Example Datasets & -loader
  - Exercise 3 (bonus-related)

General Exercise Overview

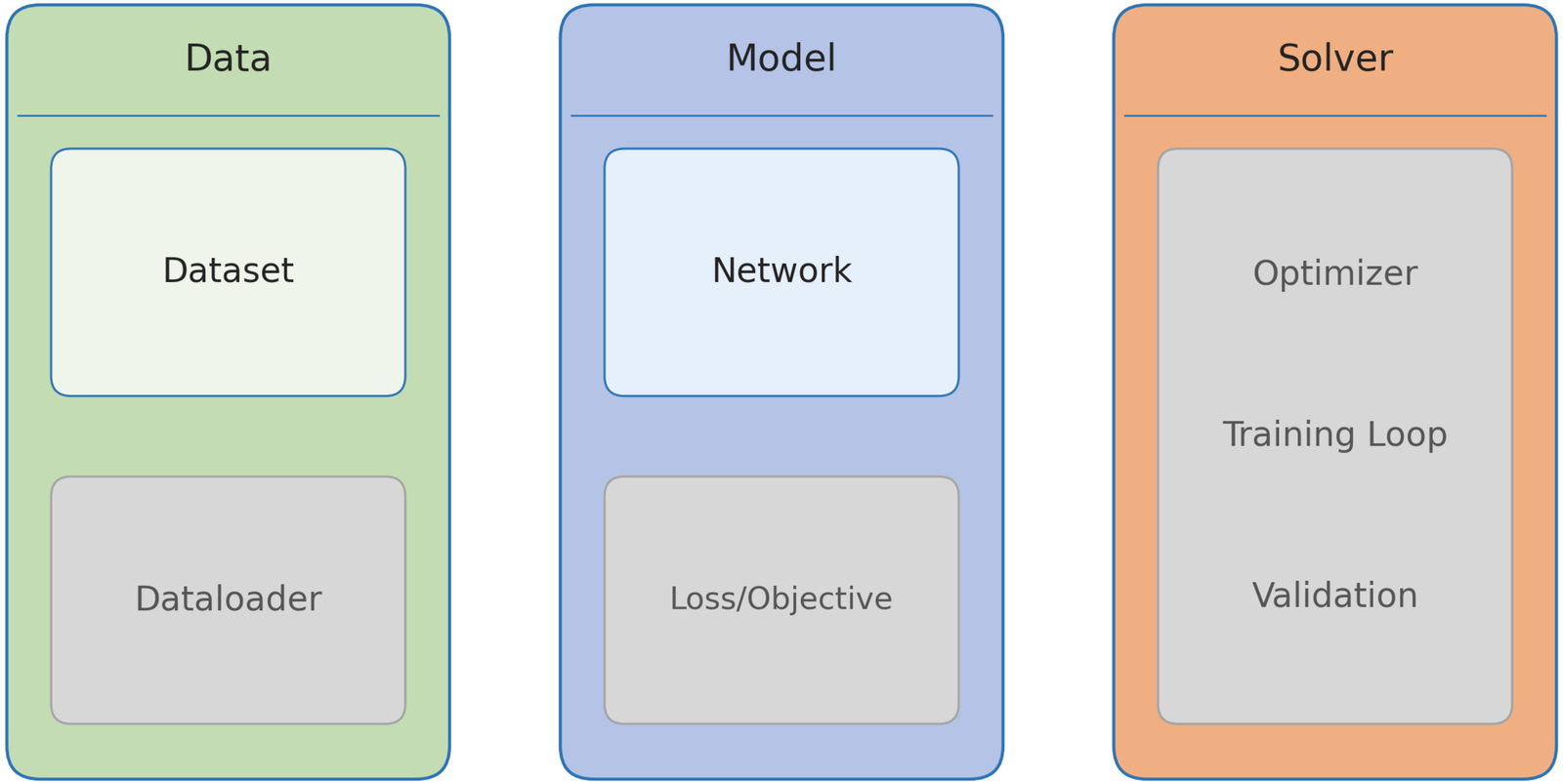
The Pillars of Deep Learning



The Pillars of Deep Learning



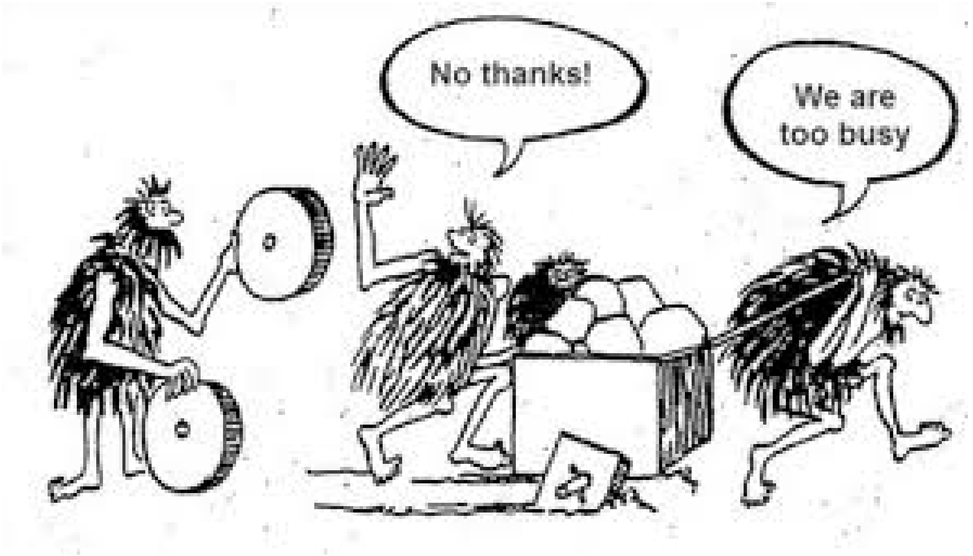
The Pillars of Deep Learning



Can be implemented once and used in multiple projects

Your task for exercises 3–5

- **Implement the basic building blocks in NumPy (no PyTorch yet)**
  - Exercise 3: Dataset and Dataloader
  - Exercise 4: Solver on a simple regression task
  - Exercise 5: Your first neural network
- Hands-on with the moving parts before we trust a framework with them



Exercise 3

Exercise 3: Dataset

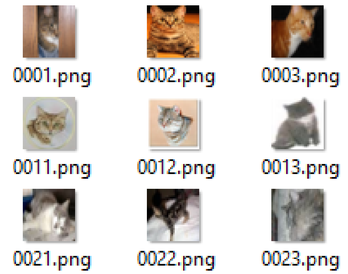
- Reads data and provides a simple way to access individual samples
- **Performs on-the-fly preprocessing and augmentations**
  - Preprocessing — e.g. scale image to a fixed size
  - Augmentations — e.g. random flips, random crops

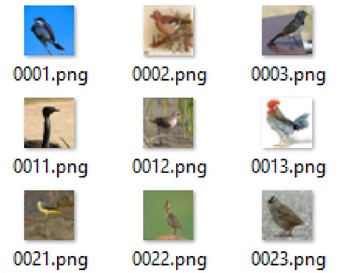
Example: Image Classification Dataset

- Given: Path to a folder with 10 sub-folders

<dataset_root>
|- cat
|- bird
|- car
|- ...

- Each folder contains X images of a specific category



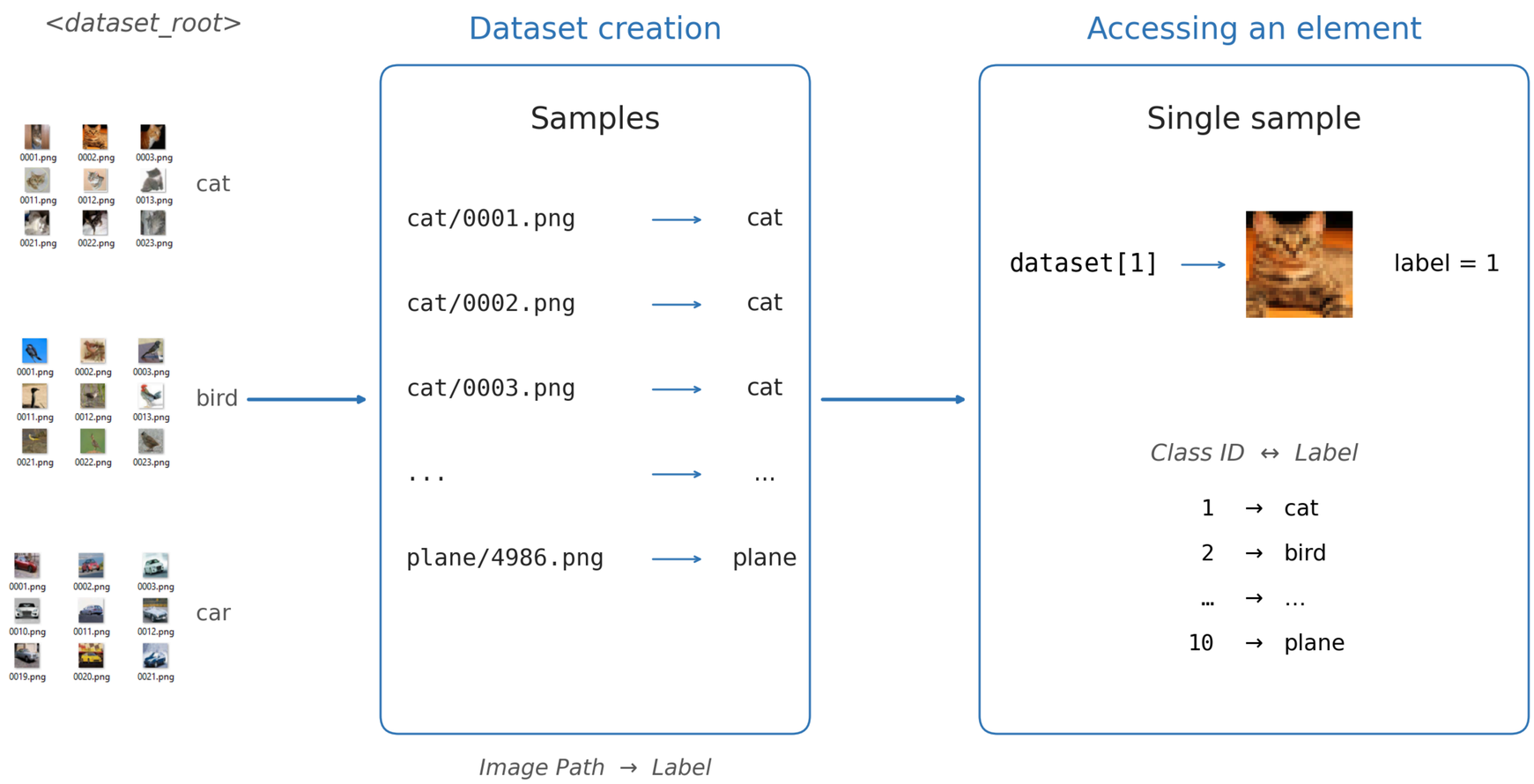




Example: Image Classification Dataset

- **Dataset class reads the folder structure on init**
  - ImageDataset(<dataset_root>)
  ➔ samples = [("cat/0001.png", 1), ..., ("plane/4986.png", 10)]
  - Usually it does NOT open the images yet
  - Defines a class-ID ↔ label mapping (e.g. 1 ↔ cat, 10 ↔ plane)
- **Calling dataset[i] gives one element**
  - Reads the image from disk
  - Performs on-the-fly preprocessing
  - Performs augmentations

Example: Image Classification Dataset

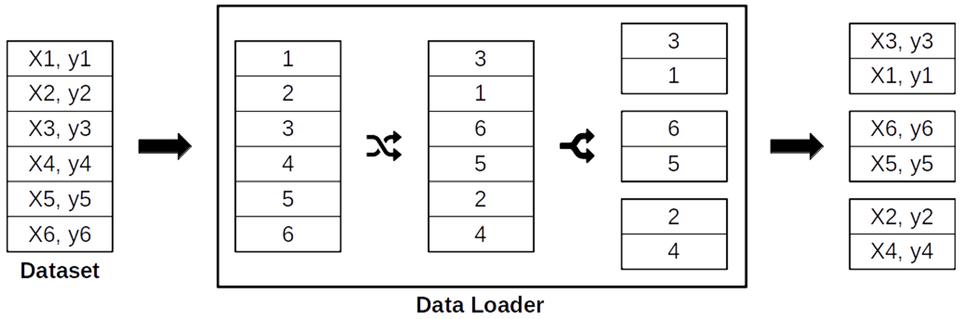


Exercise 3: Dataset

- **What we excluded**
  - Low-level scripting details (operating-system calls)
- **Reading every file from disk one-by-one vs loading the entire dataset into memory**
  - Datasets are usually too big to fit in RAM
  - When they DO fit, loading once into memory gives a huge training-time speed-up

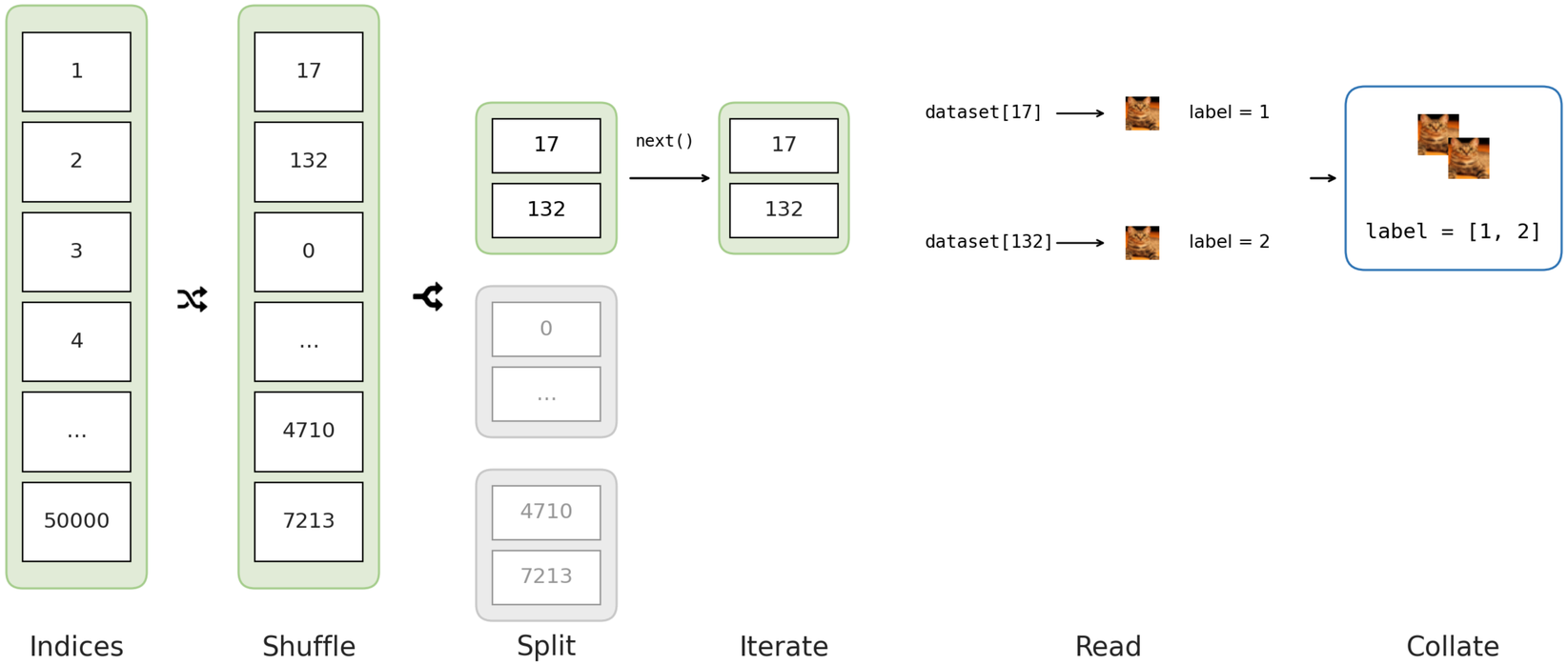
Exercise 3: Dataloader

- **Defines how the dataset is loaded during training**
  - Number of images per batch
  - Number of workers (parallel processes)
- **Shuffles** the dataset so the network doesn't see one class at a time
- **Splits** the data into small subsets — (mini-)batches



Exercise 3: Dataloader — Iterator & Batching

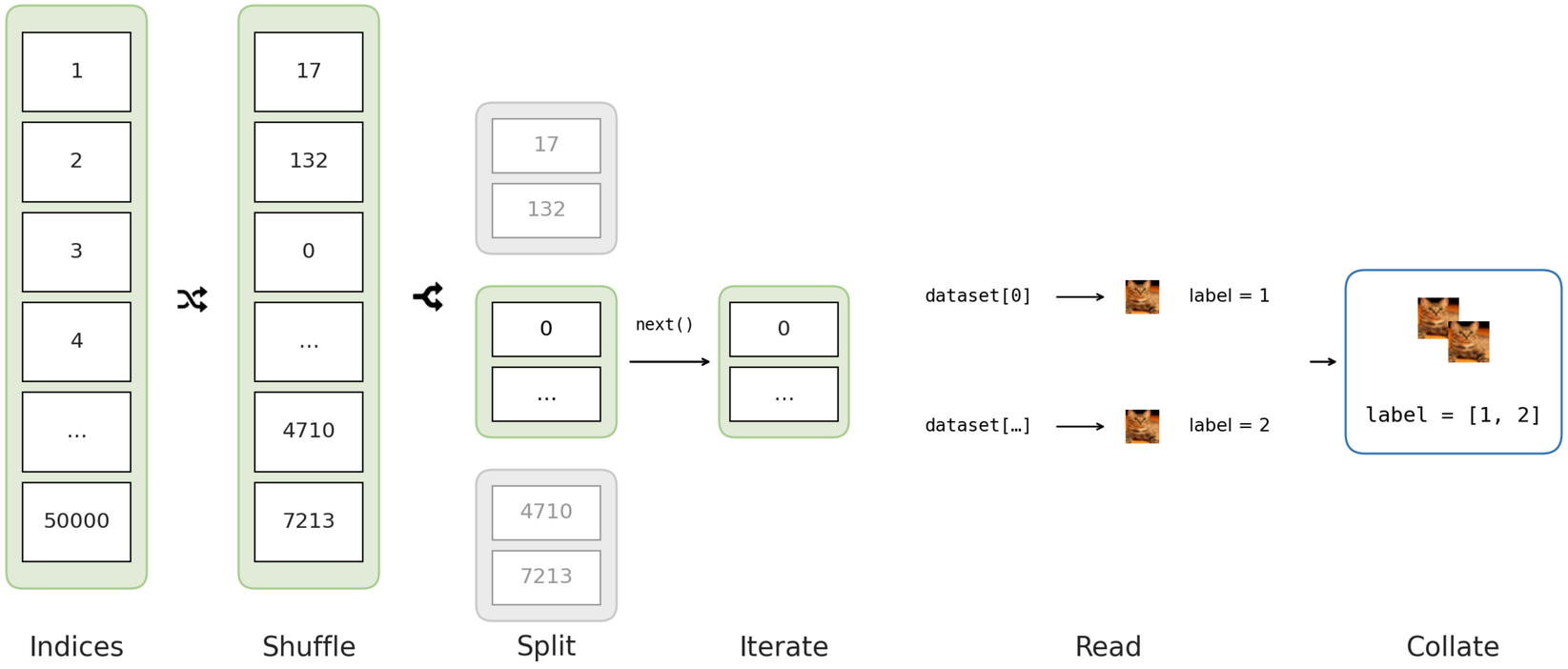
✗ `dataloader[i]` ✓ `next(dataloader)`



(Can be done in parallel, multiple workers)

Exercise 3: Dataloader — Iterator & Batching

✗ `dataloader[i]` ✓ `next(dataloader)`



(Can be done in parallel, multiple workers)

Overview — Exercise 3

- **Two notebooks**
  - Dataset (CIFAR-10)
  - Dataloader
- **Submission**
  - Implement solutions in BOTH notebooks
  - A single submission zip is generated in the dataloader notebook

See you next week!