DATASCI207-007
Applied Machine Learning

Vilena Livinsky, PhD(c)

School of Information, UC Berkeley

Week 1: 01/09/2025

Introductions: Welcome to the class!

- Current:
  - Lecturer in the School of Information at UC Berkeley
  - Full-time AI Data scientist for a leading U.S. financial services institution
    - Risk modelling
    - NLP/LLM modelling
    - Cybersecurity modelling
  - Researcher in speech science
    - PhD Candidate in Speech Science (CUNY)
- Fun fact:
    - Love travelling
    - Love very hot weather!

About You

- Undergraduate/ Graduate Major
- Current position (if any)/ Would like to transition to?
- DATASCI207: Topic most interested in exploring in this class?
- Something you’d like to share about yourself?

Live Sessions

- Each session is 90 minutes
  - Review of/ further deep dive into the week’s material
  - Code demonstration/ Breakout room exercises
  - Q&A about the topic of the week

Course Resources

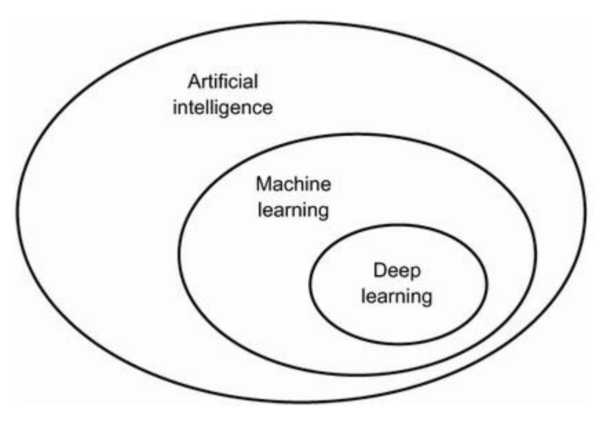
Async material	• in bCourses: https://bcourses.berkeley.edu/
Live session	• Live session material will be posted on Slack • Live lecture recording will be synced into each calendar invite in bCourses
Slack Channels	• General Slack Channel: #datasci-207-2025-spring • Section Slack Channel: #datasci-207-007-w25
Schedule	• Live session: Th, 6:30 pm - 7:59PM PST • Office hour: Wed, 8:00 - 9:00AM PST

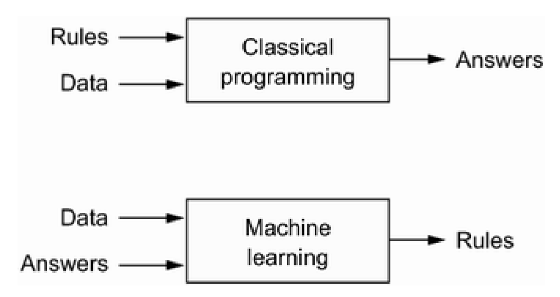
Today’s Agenda

- Introductions
- General concepts of Machine Learning
  - Typical workflow for ML predictive modeling
- Walkthrough
  - NumPy Arrays
    - 1-D, 2-D, and n-D arrays (tensors)
- Walkthrough
  - a model build

AI | Machine Learning | Deep Learning

- AI: automation of human intellectual tasks





- ML: ML learning system is trained (learning a function) given some data
  - Requires:
    - Input (exposure to examples)
    - Output
    - Metric for success (algo's current output vs expected output = feedback)
- Deep Learning: "layered representations learning"
  - Learning successive layers of increasingly meaningful representations (of data)
    - Neural network models

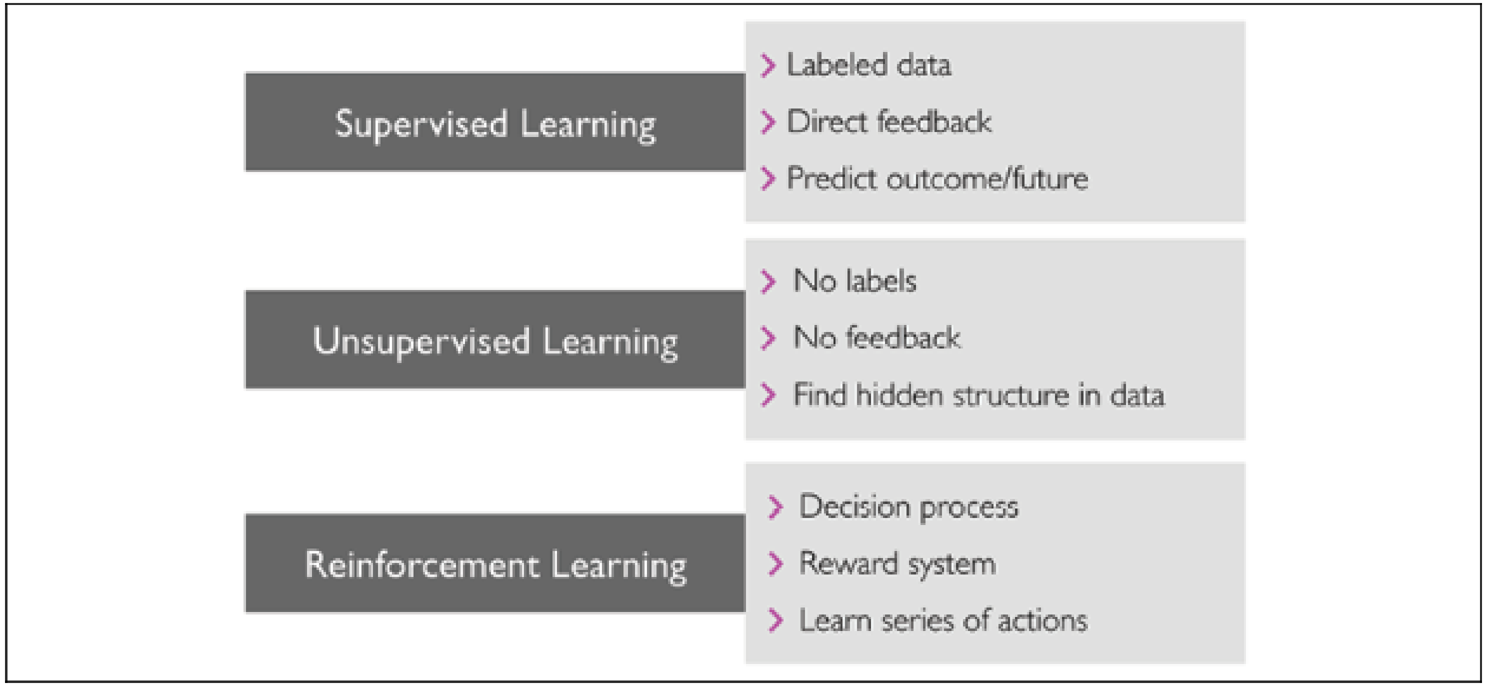
Chollet, F. (2021). Deep learning with Python / François Chollet. (Second edition.). Manning Publications Co. LLC

Types of Machine Learning

- What types of Machine Learning come to mind?

Raschka, S., & Mirjalili, V. (2019). Python machine learning : machine learning and deep learning with python, scikit-learn, and tensorflow 2 / Sebastian Raschka, Vahid Mirjalili. (Third edition.). Packt.

Types of Machine Learning



Supervised Learning

- Labeled data
- Direct feedback
- Predict outcome/future

Unsupervised Learning

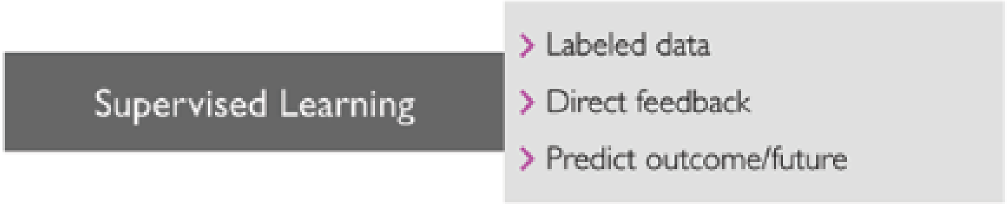
- No labels
- No feedback
- Find hidden structure in data

Reinforcement Learning

- Decision process
- Reward system
- Learn series of actions

Raschka, S., & Mirjalili, V. (2019). Python machine learning : machine learning and deep learning with python, scikit-learn, and tensorflow 2 / Sebastian Raschka, Vahid Mirjalili. (Third edition.). Packt.

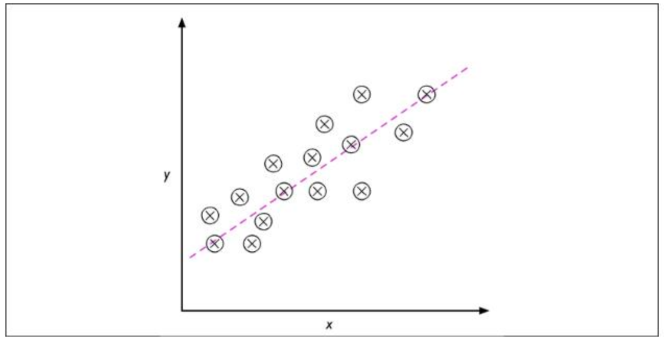
Data & Learning a Function (Model)



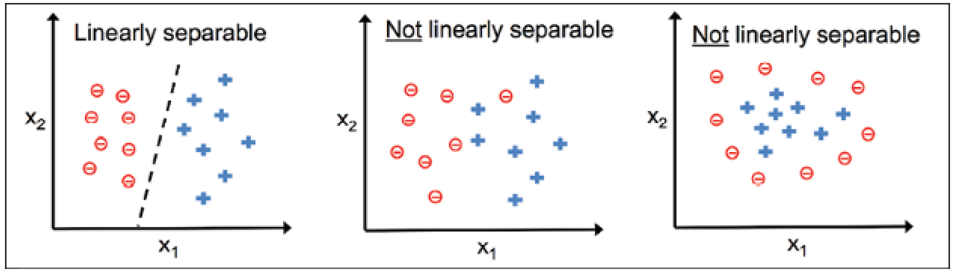
Supervised Learning

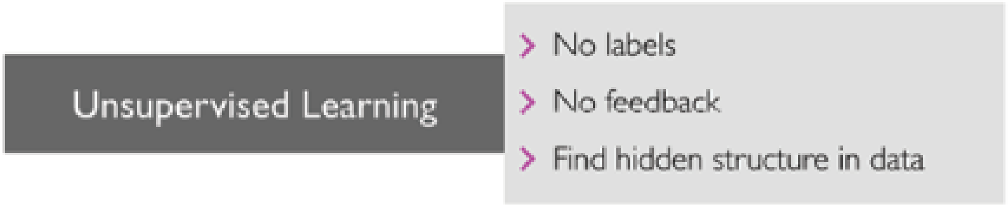
- Labeled data
- Direct feedback
- Predict outcome/future

- Regression
  - Predicting continuous outcomes



- Classification
  - Predicting class labels

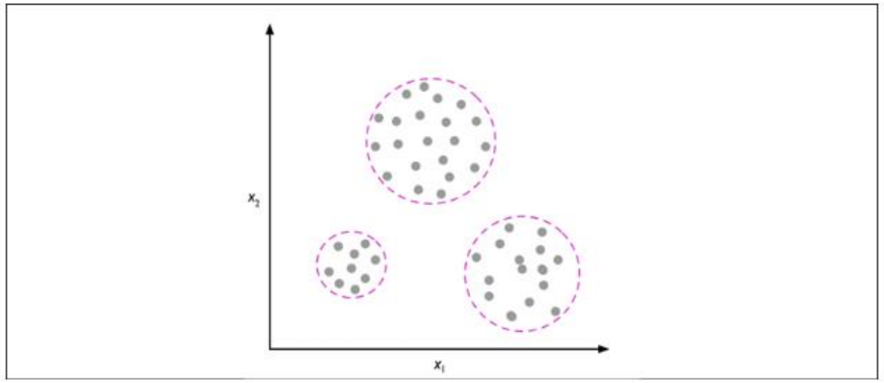




Unsupervised Learning

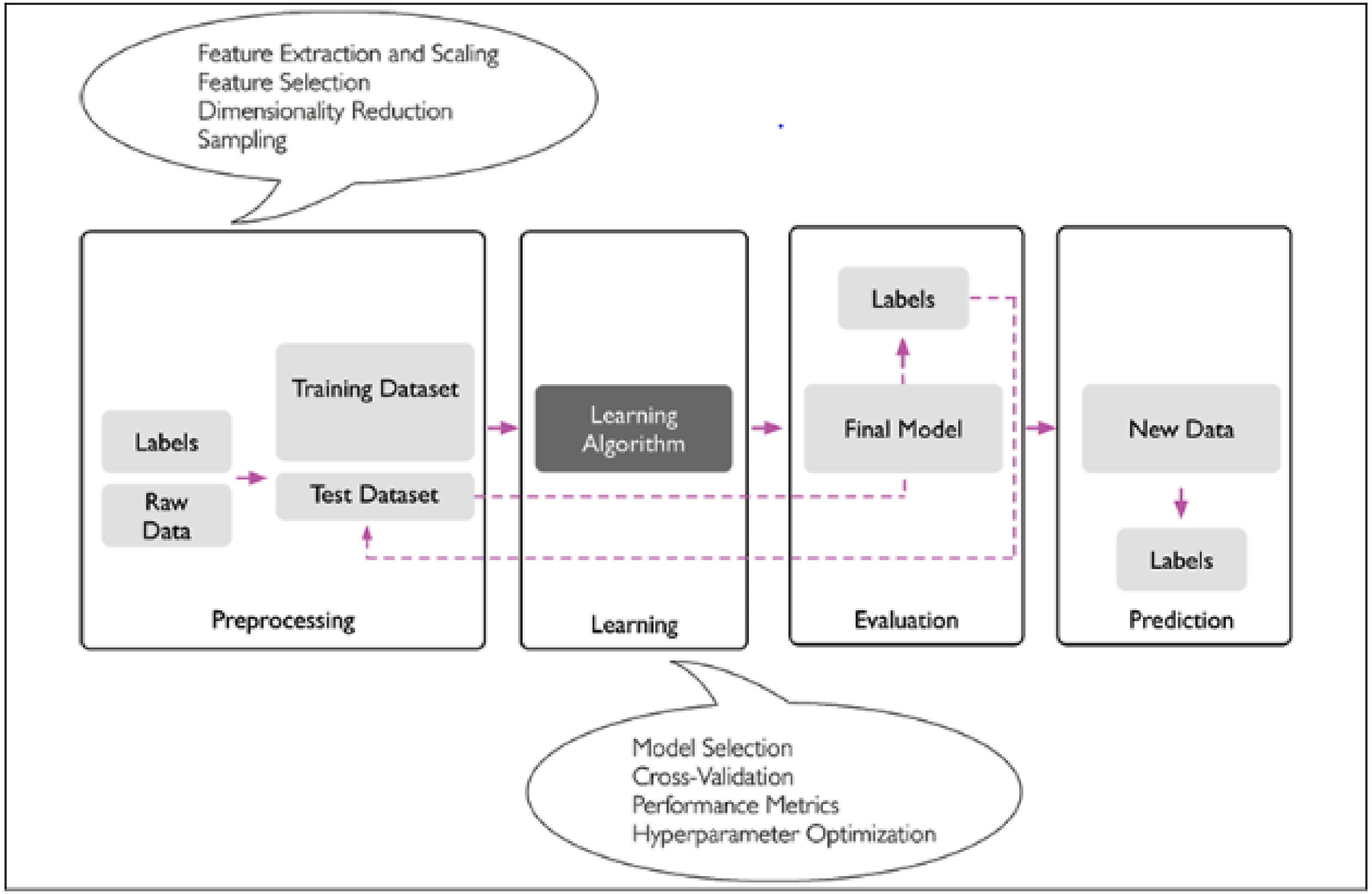
- No labels
- No feedback
- Find hidden structure in data

- Clustering
  - Finding subgroups in data



Raschka, S., & Mirjalili, V. (2019). Python machine learning : machine learning and deep learning with python, scikit-learn, and tensorflow 2 / Sebastian Raschka, Vahid Mirjalili. (Third edition.). Packt.

Machine Learning Workflow



Q. Why do we need a train-test split?

Raschka, S., & Mirjalili, V. (2019). Python machine learning : machine learning and deep learning with python, scikit-learn, and tensorflow 2 / Sebastian Raschka, Vahid Mirjalili. (Third edition.). Packt.

Tools

- ML/ AI APIs:
- CPU = general purpose processors
- GPUs/ TPUs = optimized to accelerate ML

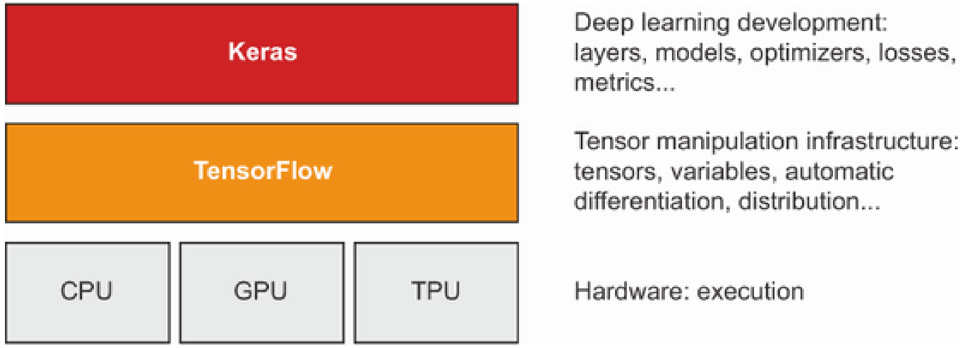

Keras
Deep learning development: layers, models, optimizers, losses, metrics...
TensorFlow
Tensor manipulation infrastructure: tensors, variables, automatic differentiation, distribution...
CPU
GPU
TPU
Hardware: execution

Figure 3.1 Keras and TensorFlow: TensorFlow is a low-level tensor computing platform, and Keras is a high-level deep learning API

- scikit-learn
  - will be used occasionally, especially for data preprocessing and model evaluation

- Other APIs:

- 1. NumPy:
  - adds support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays

- 2. Pandas:
  - useful for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables

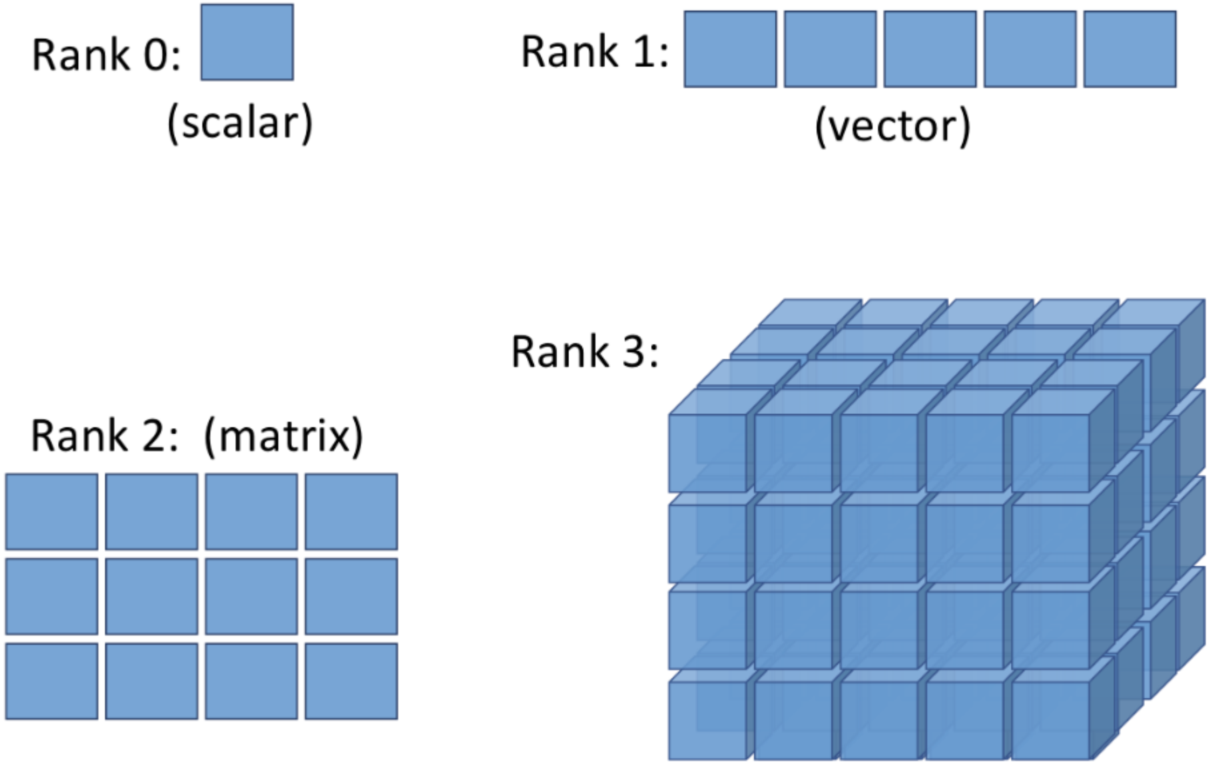
- 3. Matplotlib:
  - a plotting library

Raschka, S., & Mirjalili, V. (2019). Python machine learning : machine learning and deep learning with python, scikit-learn, and tensorflow 2 / Sebastian Raschka, Vahid Mirjalili. (Third edition.). Packt.

Walkthrough

01a_Introduction.ipynb

Tensors



https://github.com/rasbt/python-machine-learning-book-3rd-edition/blob/master/ch13/ch13_part1.ipynb

Tensors are *multi-dimensional* arrays with a *uniform type* (called a dtype).

- tensors are (kind of) like np.arrays

  "Scalar" or "rank-0" tensor .
  A scalar contains a single value, and no "axes".

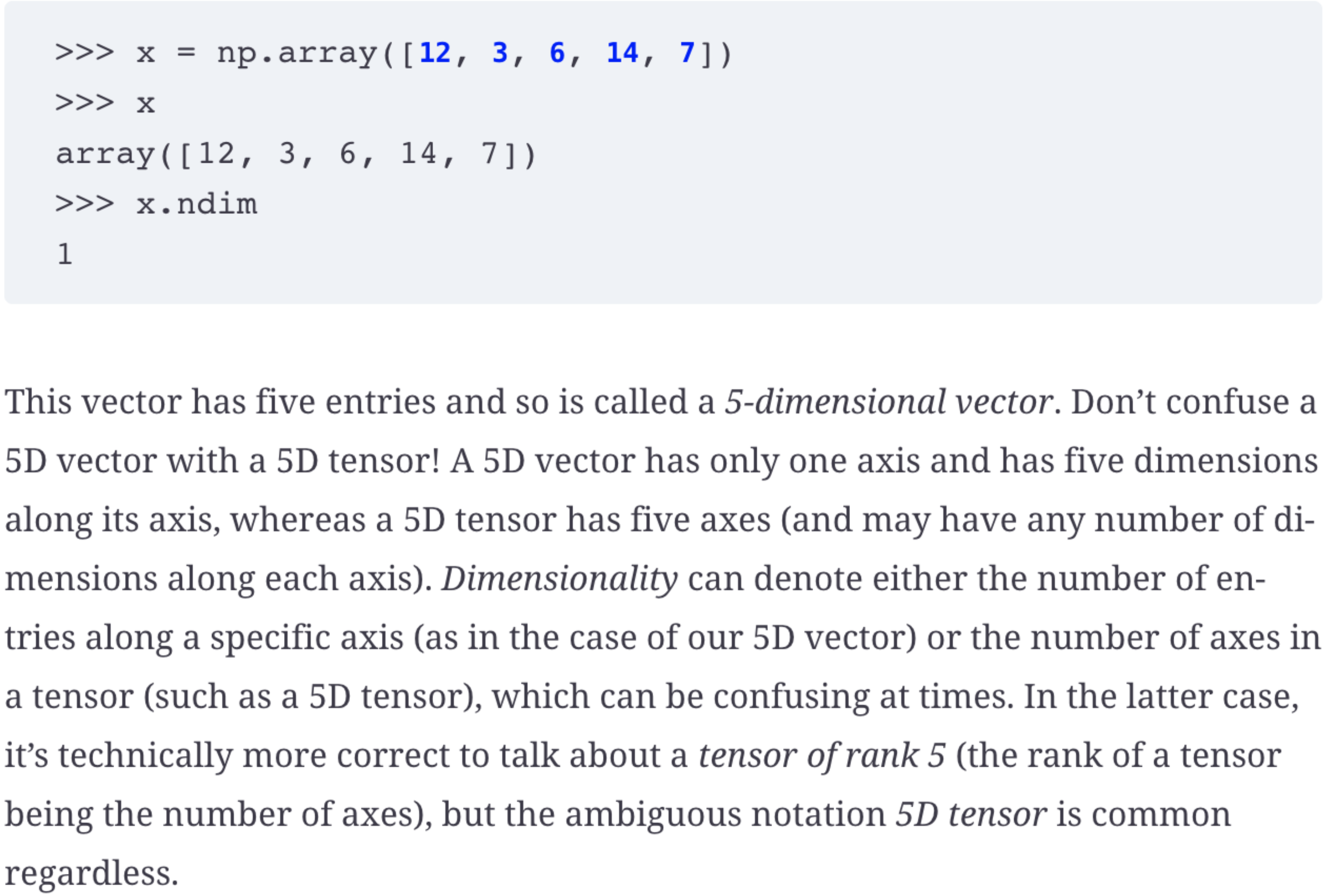
  A "vector" or "rank-1" tensor is like a list of values. A vector has one axis.

  A "matrix" or "rank-2" tensor has two axes.

TensorFlow

- https://www.tensorflow.org/guide/tensor

Tensors: Note!

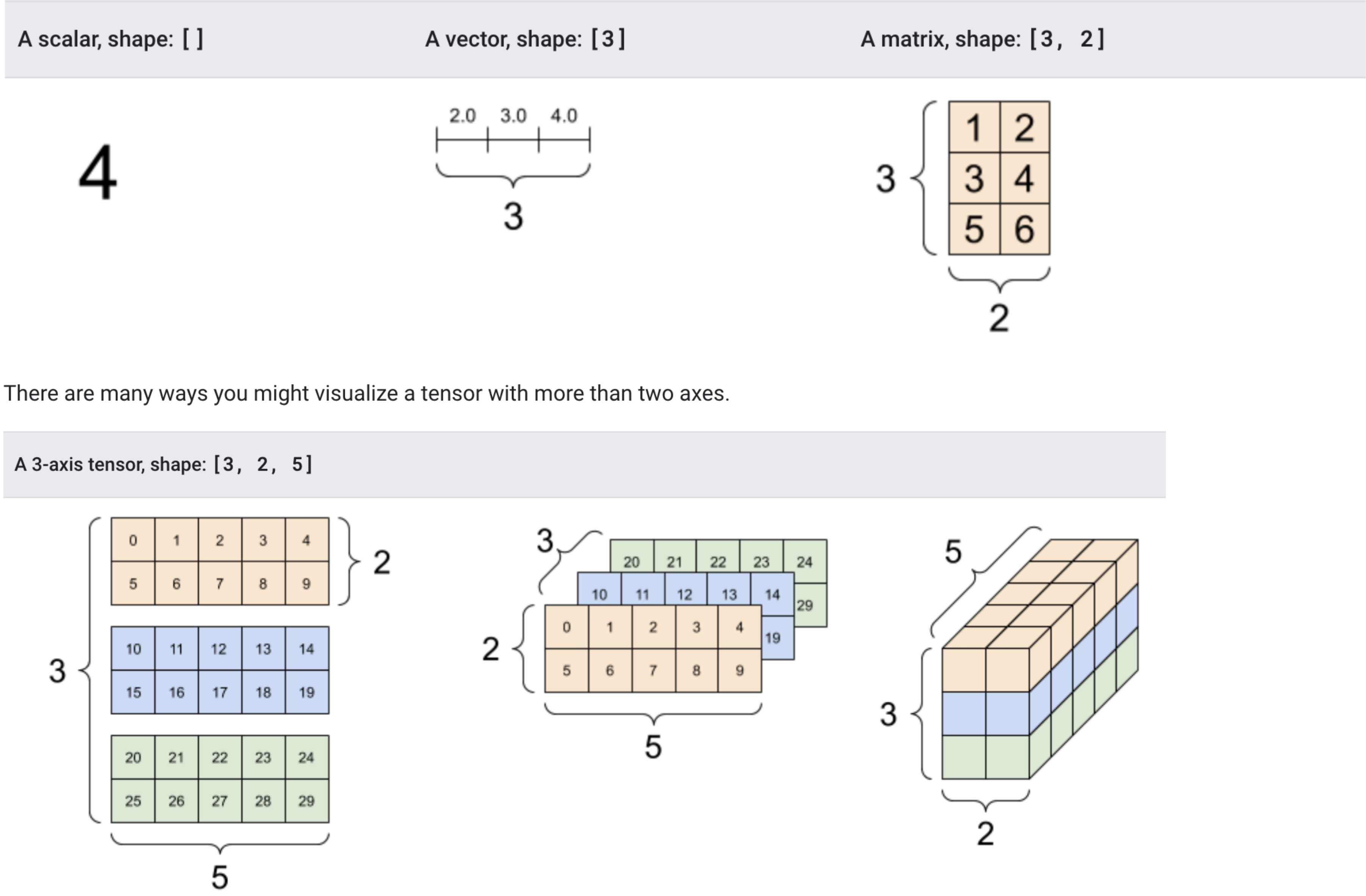


```
>>> x = np.array([12, 3, 6, 14, 7])
>>> x
array([12, 3, 6, 14, 7])
>>> x.ndim
1
```

This vector has five entries and so is called a *5-dimensional vector*. Don't confuse a 5D vector with a 5D tensor! A 5D vector has only one axis and has five dimensions along its axis, whereas a 5D tensor has five axes (and may have any number of dimensions along each axis). *Dimensionality* can denote either the number of entries along a specific axis (as in the case of our 5D vector) or the number of axes in a tensor (such as a 5D tensor), which can be confusing at times. In the latter case, it's technically more correct to talk about a *tensor of rank 5* (the rank of a tensor being the number of axes), but the ambiguous notation *5D tensor* is common regardless.

Ref.: https://learning.oreilly.com/library/view/deep-learning-with/9781617296864/Text/02.htm#heading_id_7

Tensors



A scalar, shape: [ ]

A vector, shape: [3]

A matrix, shape: [3, 2]

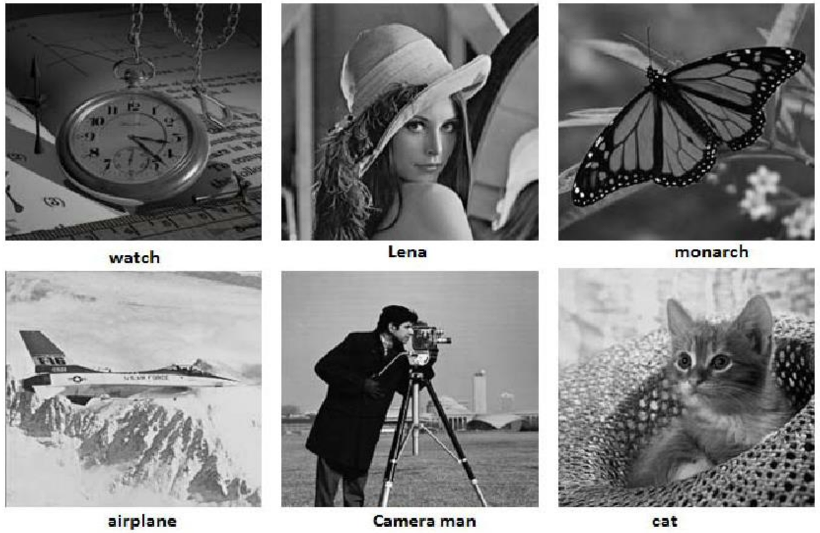
There are many ways you might visualize a tensor with more than two axes.

A 3-axis tensor, shape: [3, 2, 5]

Ref.: https://www.tensorflow.org/guide/tensor

Tensors: Examples

- A batch of 128 *grayscale* images of size 256 × 256 could be stored in a tensor of shape (128, 256, 256, 1)
- A batch of 128 *color* images could be stored in a tensor of shape (128, 256, 256, 3)



https://learning.oreilly.com/library/view/deep-learning-with/9781617296864/Text/02.htm#heading_id_15

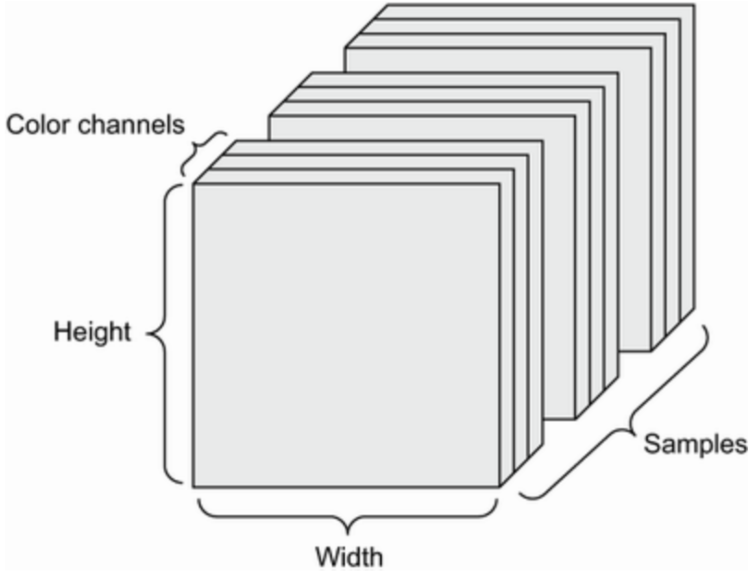


Figure 2.4 A rank-4 image data tensor

Walkthrough

01b_Framing.ipynb