# Deep Learning Bootcamp

2026

GSA

IGP GSC

ISE GRADUATE STUDENT CLUB

CCDS GRADUATE STUDENT CLUB

MAE GRADUATE STUDENT CLUB

SPMS Graduate Student Club

NTU EEE GSC

# Agenda

1. Introduction to Python
2. Basic numerical libraries for ML

**Break 15:30**

1. Linear Regression
2. Exploratory data analysis
3. Linear regression with PyTorch (deep learning library)

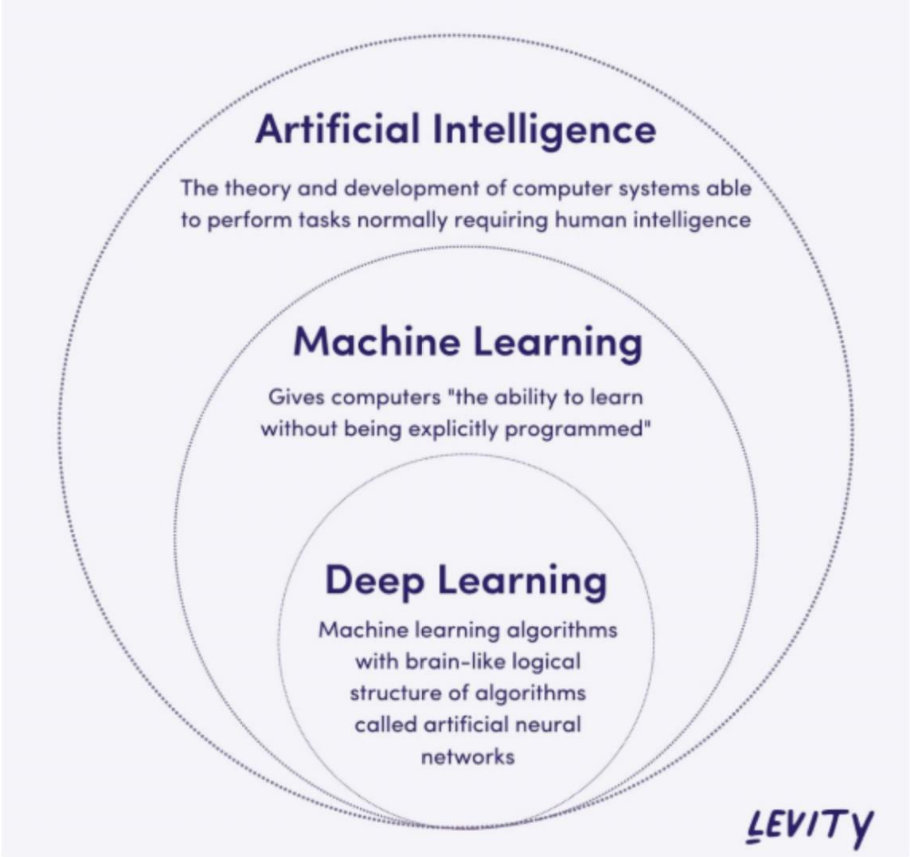
# Schedule

| Event No. | Details |
|---|---|
| **Event 1**<br>**24th Oct 2025** | *Deep Learning Essentials*<br>Covers the basics of Python and necessary packages required for Deep Learning such as numpy, scipy, pandas etc. |
| **Event 2**<br>**10th Nov 2025** | *Deep Learning for Regression and Classification*<br>Will cover the basics of PyTorch, as well as how to use PyTorch for performing regression and classification tasks. |
| **Event 3**<br>**17th Nov 2025** | *Deep Learning for Images*<br>In this event, we will extend the classification using deep learning, specifically focusing on datasets involving images. |
| **Event 4**<br>**TBD – Sem 2 – 2026** | *Deep Learning for Sequence Data (text and time series)*<br>In this event, we will focus on using Deep Learning models for datasets involving sequences or temporal relations. We plan to cover examples from both text and time-series datasets. |
| **Event 5**<br>**TBD – Sem 2 – 2026** | *Reinforcement Learning*<br>This session will introduce Deep Reinforcement Learning techniques with some practical applications. |

# What is Deep Learning?

- Machine learning: learn from examples or data
- Deep Learning: use artificial neural networks: weight * variable + some non-linear layer

$$y = \sigma\left(W_L \ldots \sigma\left(W_2 \sigma(W_1 x)\right)\right)$$



**Artificial Intelligence**

The theory and development of computer systems able to perform tasks normally requiring human intelligence

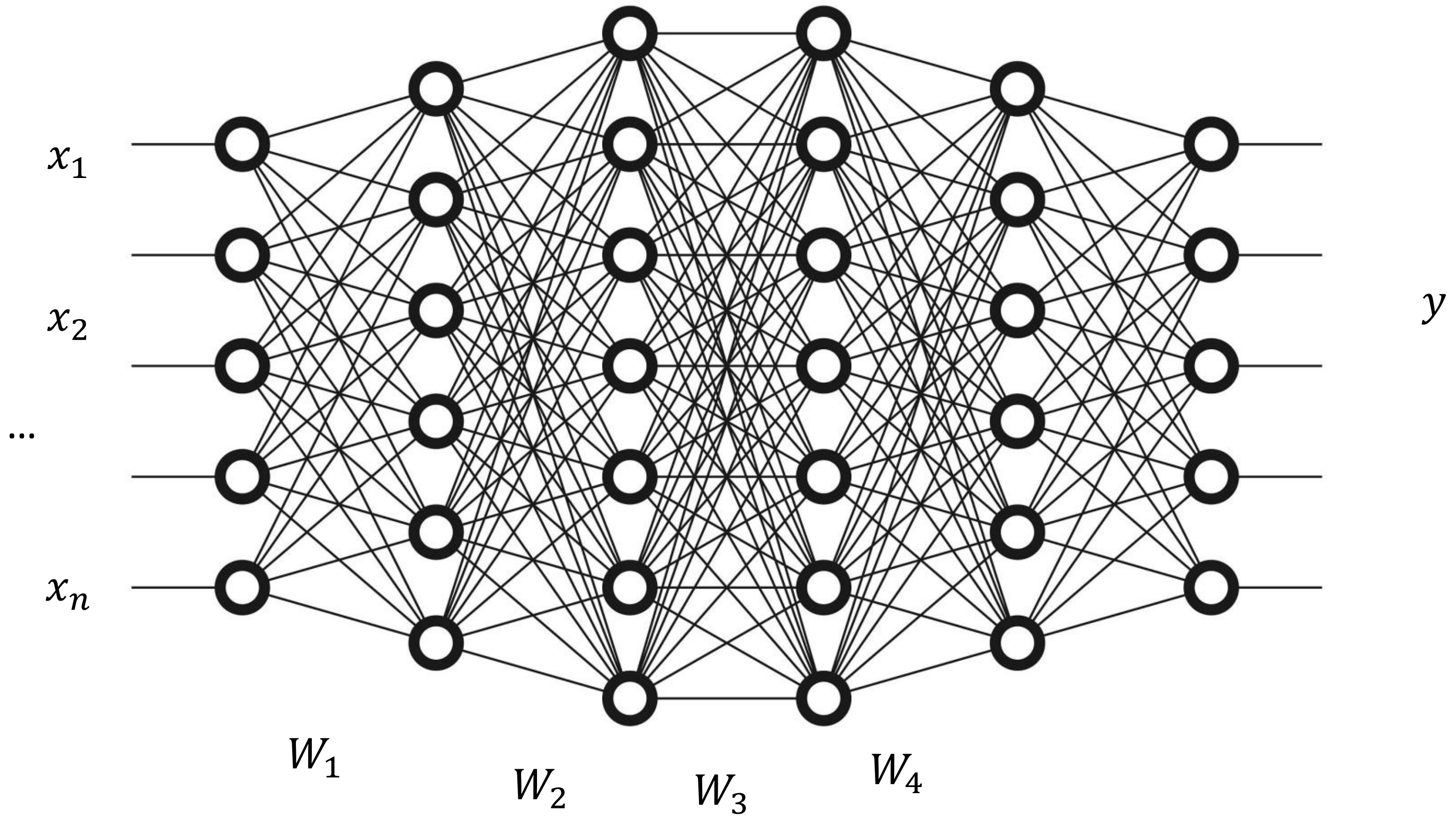
**Machine Learning**

Gives computers "the ability to learn without being explicitly programmed"

**Deep Learning**

Machine learning algorithms with brain-like logical structure of algorithms called artificial neural networks

# Deep Learning is powered by deep neural networks



Deep as in depth

i.e., many layers

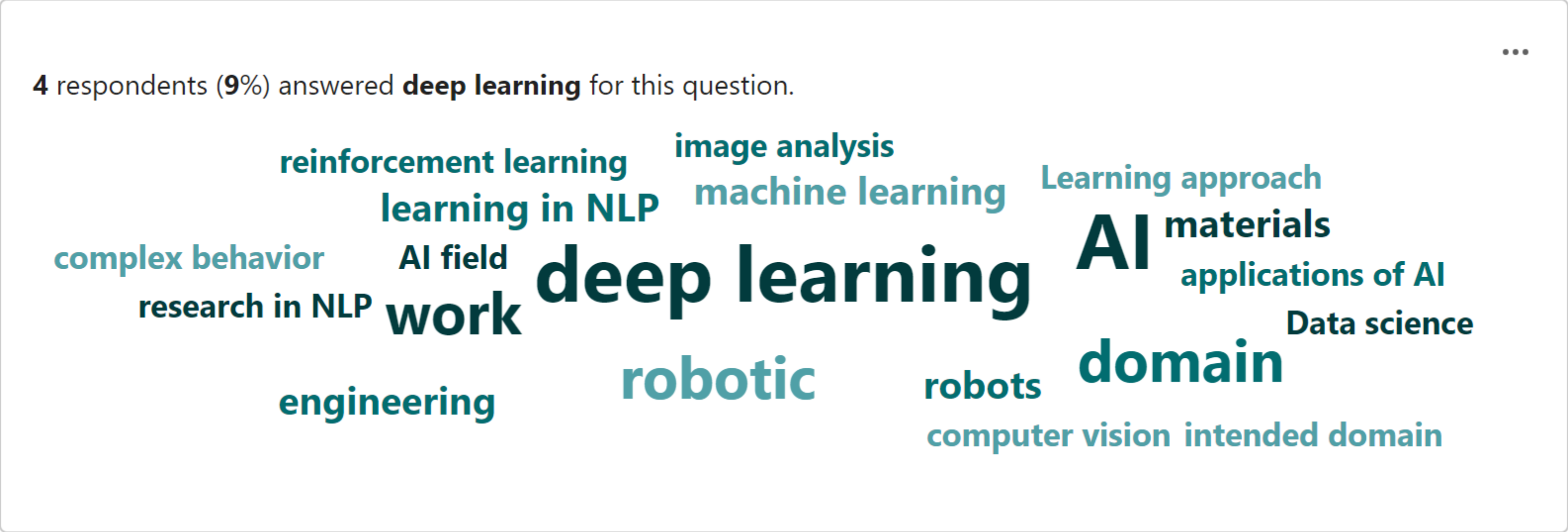
https://cdn-images-1.medium.com/max/2400/1*1mpE6fsq5LNxH31xeTWi5w.jpeg

# Why Deep Learning?

- Find hidden patterns in a lot of data (high-dimensional, complex, etc.)
- Many applications, open models
  - Material properties prediction
  - Protein folding: see AlphaFold
  - Text generation/Natural Language Processing: ChatGPT
  - Image generation: DALLE, MidJourney, etc.
  - Computer Vision: ImageNet, image segmentation, detect face in phone, etc.
  - …

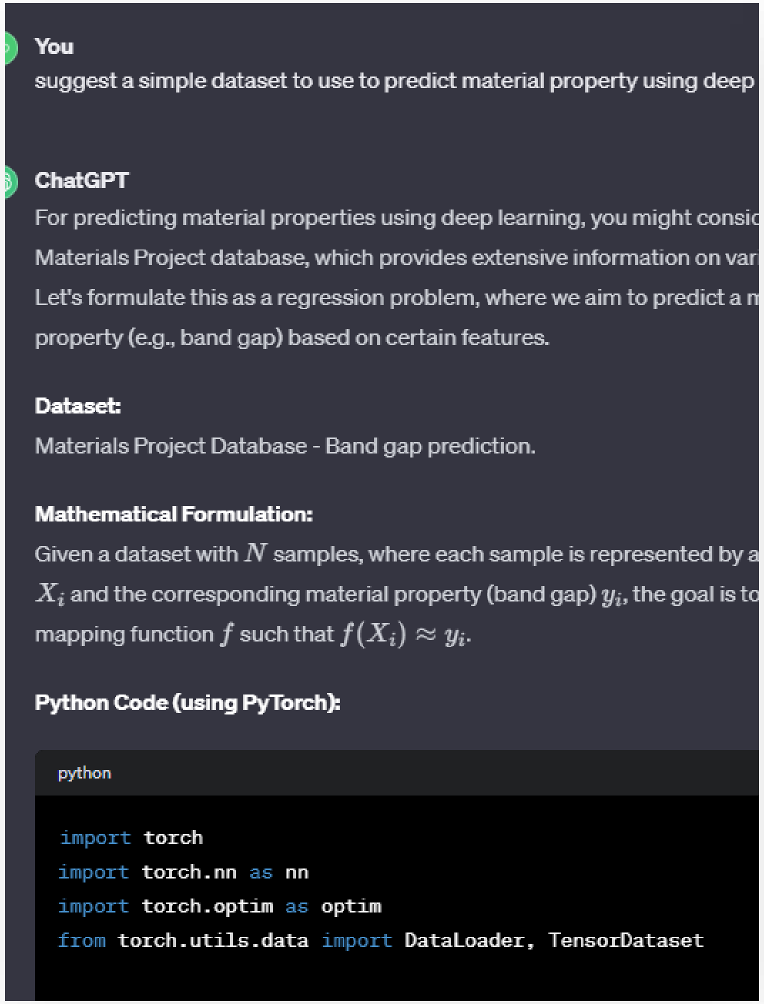
# Deep Fakes…

# Your interest



**4** respondents (**9**%) answered **deep learning** for this question.

# Examples



**You**

suggest a simple dataset to use to predict material property using deep

**ChatGPT**

For predicting material properties using deep learning, you might consic Materials Project database, which provides extensive information on var Let's formulate this as a regression problem, where we aim to predict a n property (e.g., band gap) based on certain features.

**Dataset:**

Materials Project Database - Band gap prediction.

**Mathematical Formulation:**

Given a dataset with $N$ samples, where each sample is represented by a $X_i$ and the corresponding material property (band gap) $y_i$, the goal is to mapping function $f$ such that $f(X_i) \approx y_i$.

**Python Code (using PyTorch):**

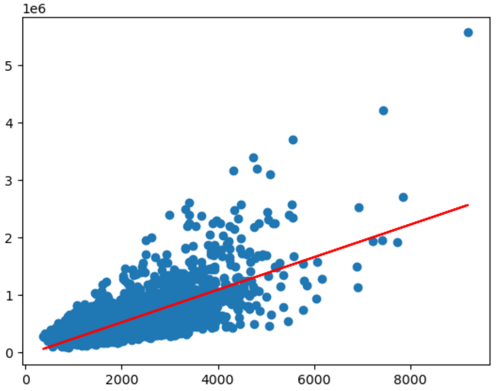
python

```
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import DataLoader, TensorDataset
```

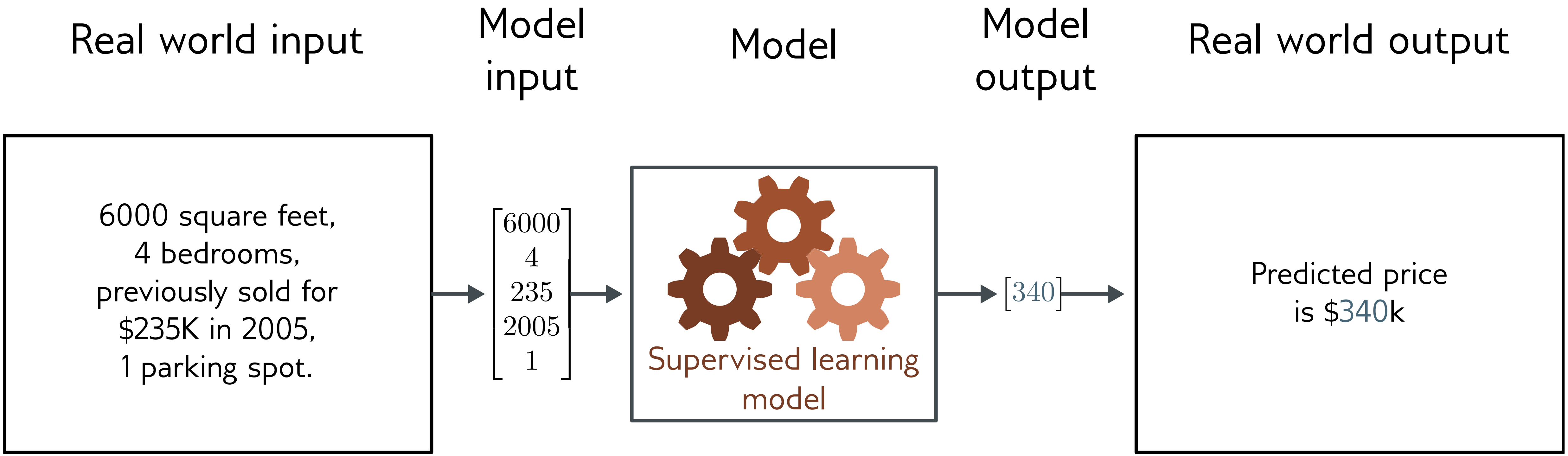
# ML 101

- Approximate a function $y = f(x;\theta)$
  - $x$: image pixels; $y$: classes (dog, cat)
  - $x$: house size; $y$: house price
  - $\theta$ are the parameters

- Two typical problems
  - Classification: discrete categories
  - Regression: continuous categories



# Regression



Real world input

Model input

Model

Model output

Real world output

6000 square feet,
4 bedrooms,
previously sold for
$235K in 2005,
1 parking spot.

$$\begin{bmatrix} 6000 \\ 4 \\ 235 \\ 2005 \\ 1 \end{bmatrix}$$

Supervised learning model

$$\begin{bmatrix} 340 \end{bmatrix}$$

Predicted price
is $340k

- Univariate regression problem (one output, real value)
- Fully connected network

# Classification

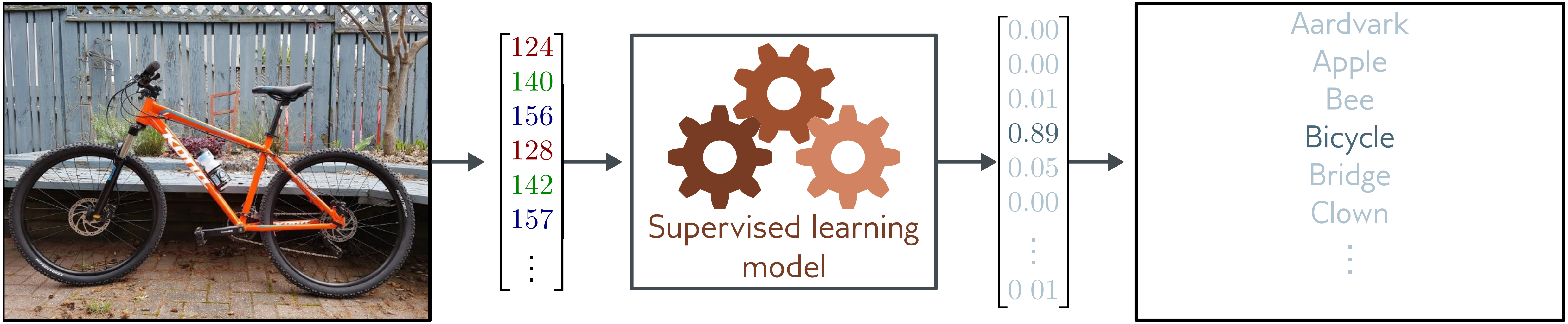
Real world input

Model input

Model

Model output

Real world output



$$\begin{bmatrix} 124 \\ 140 \\ 156 \\ 128 \\ 142 \\ 157 \\ \vdots \end{bmatrix}$$

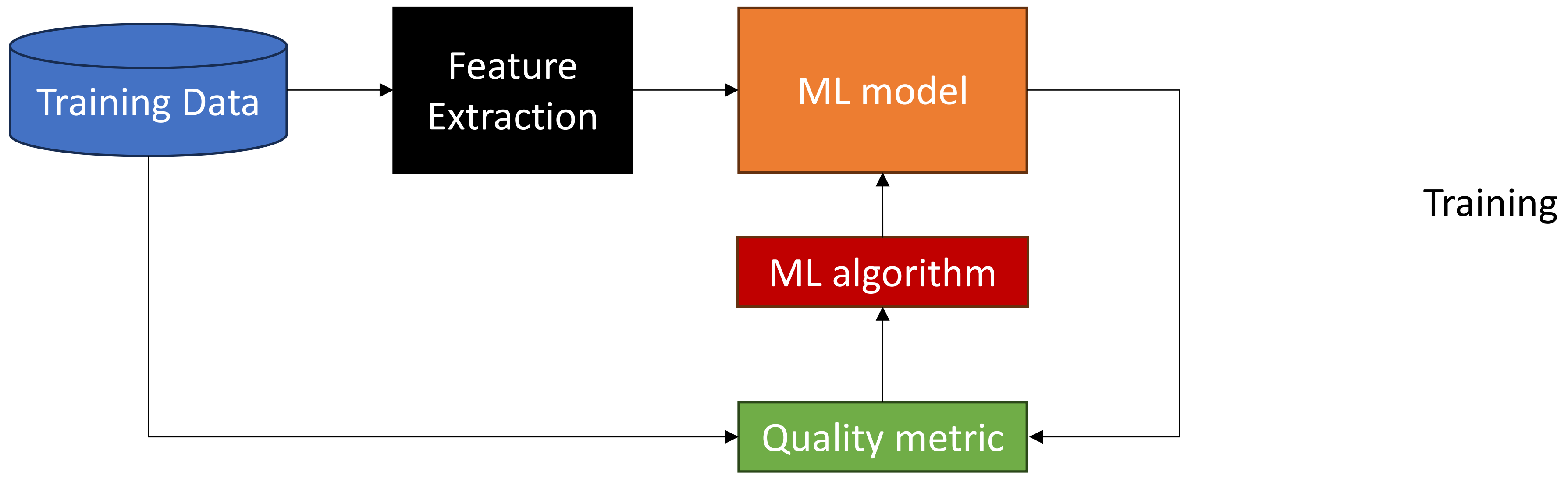
Supervised learning model

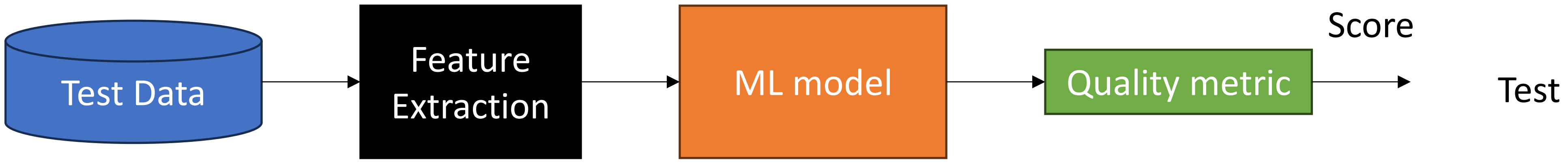
$$\begin{bmatrix} 0.00 \\ 0.00 \\ 0.01 \\ 0.89 \\ 0.05 \\ 0.00 \\ \vdots \\ 0.01 \end{bmatrix}$$

Aardvark
Apple
Bee
Bicycle
Bridge
Clown
⋮

- Multiclass classification problem (discrete classes, >2 possible classes)
- Convolutional network

# ML workflow





https://www.coursera.org/learn/ml-foundations?specialization=machine-learning

# Feature Extraction 101

- Machine learning needs numerical data as input:

$$\mathrm{x} \in \mathbb{R}^n$$

- Data normalization: assumption of data with mean 0, stddev 1

$$x' = \frac{x - \mu}{\sigma}$$

```
$ x_std = StandardScaler().fit_transform(x)
```

- Typical representations

| **Data Type** | **Common Representation** |
|---|---|
| Images | Pixel intensity arrays |
| Text | Word embeddings (Word2Vec, GloVe, BERT) |
| Audio | Spectrogram or MFCC features |

# Model Selection 101

ChatGPT for everything XGBoost

- **Many ML models to choose from:**
    - **Linear models:** simple, interpretable, good baseline
    - **Decision Trees / XGBoost:** handle complex patterns, less linear assumptions
    - **Neural Networks:** powerful but need large data and tuning

- **Not every problem needs Deep Learning!**
    - NNs can overfit on small datasets
    - Many hyperparameters, hard to tune
    - Require more computation and time

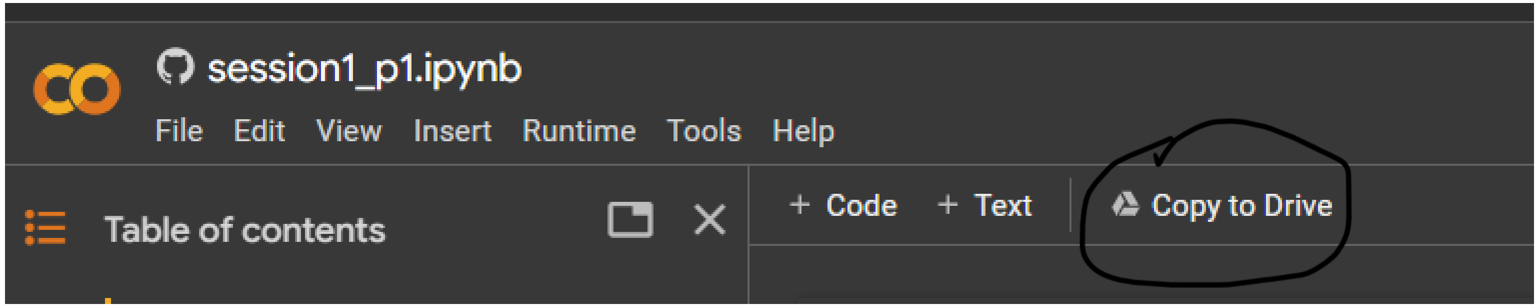
- **Model selection = experimentation**
    - Try multiple models
    - Tune hyperparameters
    - Compare results using validation metrics (e.g., accuracy, RMSE, F1)
    - Pick the model that's **accurate, simple, and practical**

# Pro & Cons of DL

| Pro | Cons |
|---|---|
| • Can learn features & representations easily<br>• Ability to process a lot of data<br>• Flexible framework<br>• Maps well into parallel hardware (GPU and others) | • Hard to understand and build intuition of why the model works: explainability and interpretability<br>• Requires a lot of data and expensive hardware. |

# Work session 1.0

# Google Colab: Our tool for today



# Handling numbers & data: numpy

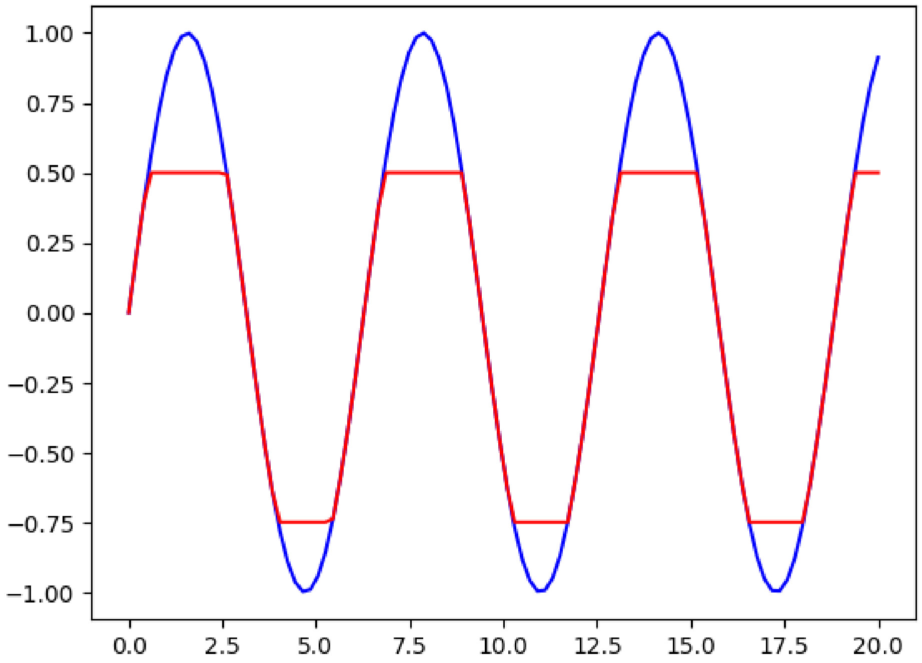
- Numpy is library for handling array, vector & matrices

- Provide all keys operations in arrays
    - Creation : *a = np.array([[1, 2, 3], [4, 5, 6]])*
    - Add/subtract/multiply: *a + b, a – b, a * b, …*
    - Dot product and matrix multiplication: *a @ b*
    - Slicing: *a[1]=> [4,5,6]*

# Plotting: matplotlib

- Matplotlib is the Python library used to plot data

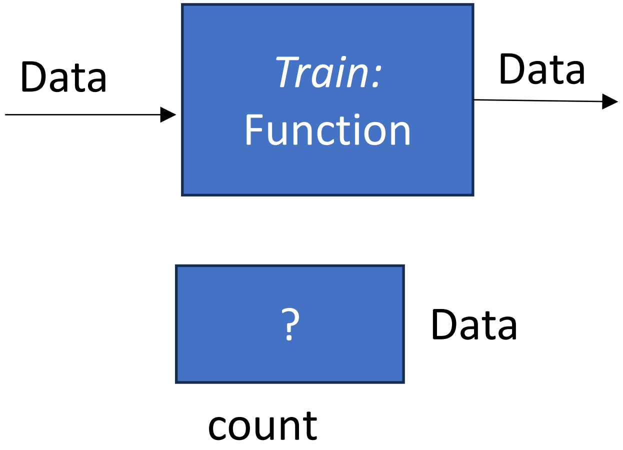
- You can create easily different types of plots (scatter plot, histogram, X-Y plot), add legend and different details.



# Python 101: Why Python?

- A lot of libraries for ML/DL: PyTorch, scikit-learn, pandas, numpy
- Easy to learn, simple syntax
- Interactive notebooks: Jupyter Notebook, Kaggle, Google Collab
- Free & open-source

# Python 101: Basic features



```
def vowel_count(word):
    vowels = ["a", "e", "i", "o", "u"]
    count = 0
    for char in word: # loops
        if char in vowels: # conditions
            count += 1
    return count

vowel_count("hello")
```

*MyModel: Class*

*my_model: Data1*
*params: Data2*
…

*Train: Function1*
*Test: Function2*
…

# Linear Regression

**Linear regression** makes predictions about the linear relationship between the input variable $x$ (contrast) and the output variable $y$ (neural response).

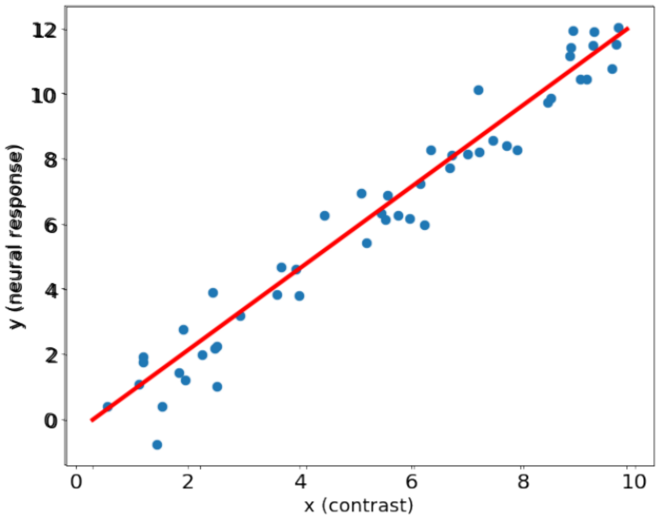
$$y = \theta_1 \times x + \theta_0$$

neural response ↑ $y$

linear weight ↑ $\theta_1$

contrast ↑ $x$

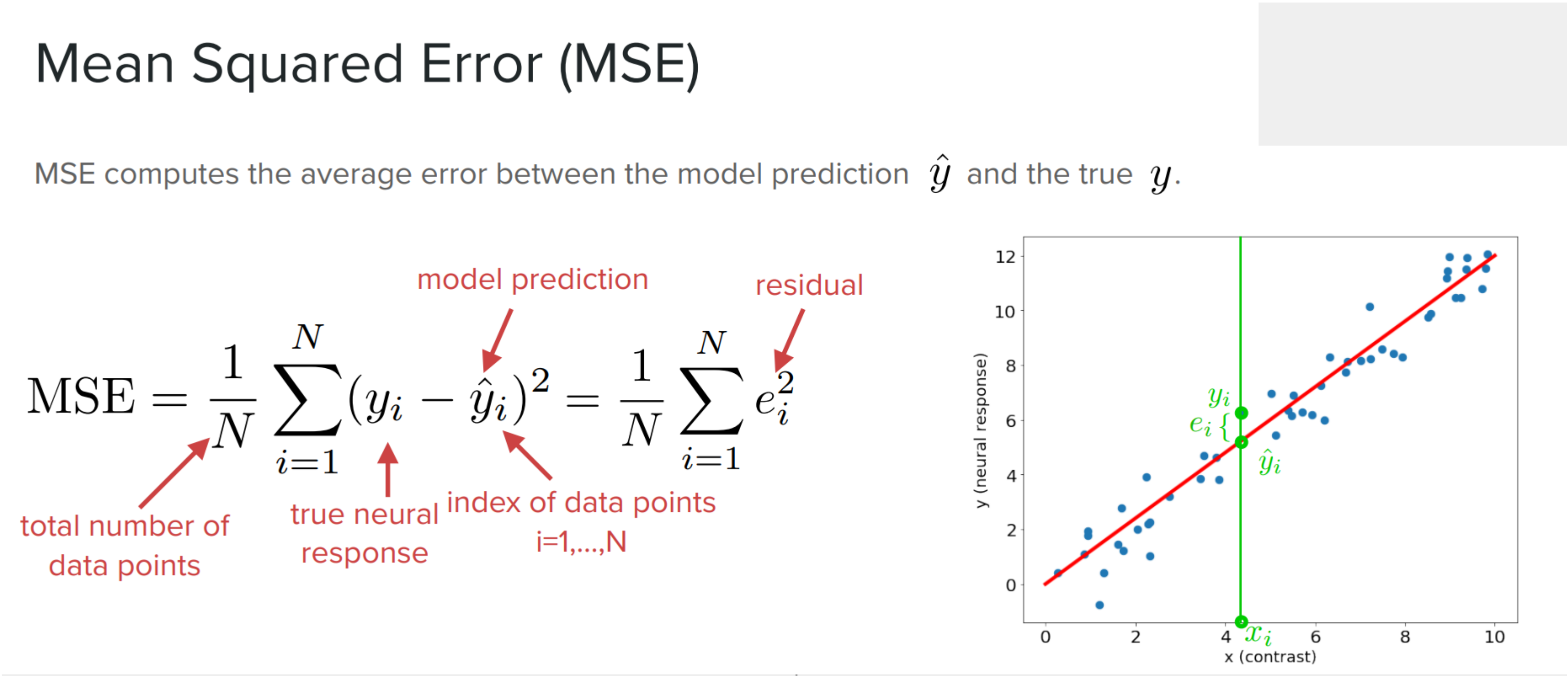
Intercept ↑ $\theta_0$



We are not considering the intercept for simplicity, resulting in a one-parameter model.

# Linear Regression: MSE

$$\min_{\theta} \frac{1}{N}\sum_{i=1}^{N}(y_i - \theta x_i)^2$$



## Mean Squared Error (MSE)

MSE computes the average error between the model prediction $\hat{y}$ and the true $y$.

model prediction

residual

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \frac{1}{N}\sum_{i=1}^{N} e_i^2$$

total number of data points

true neural response

index of data points i=1,...,N

# Work session 2

How to Evaluate a Machine Learning Model (and Why It Matters)

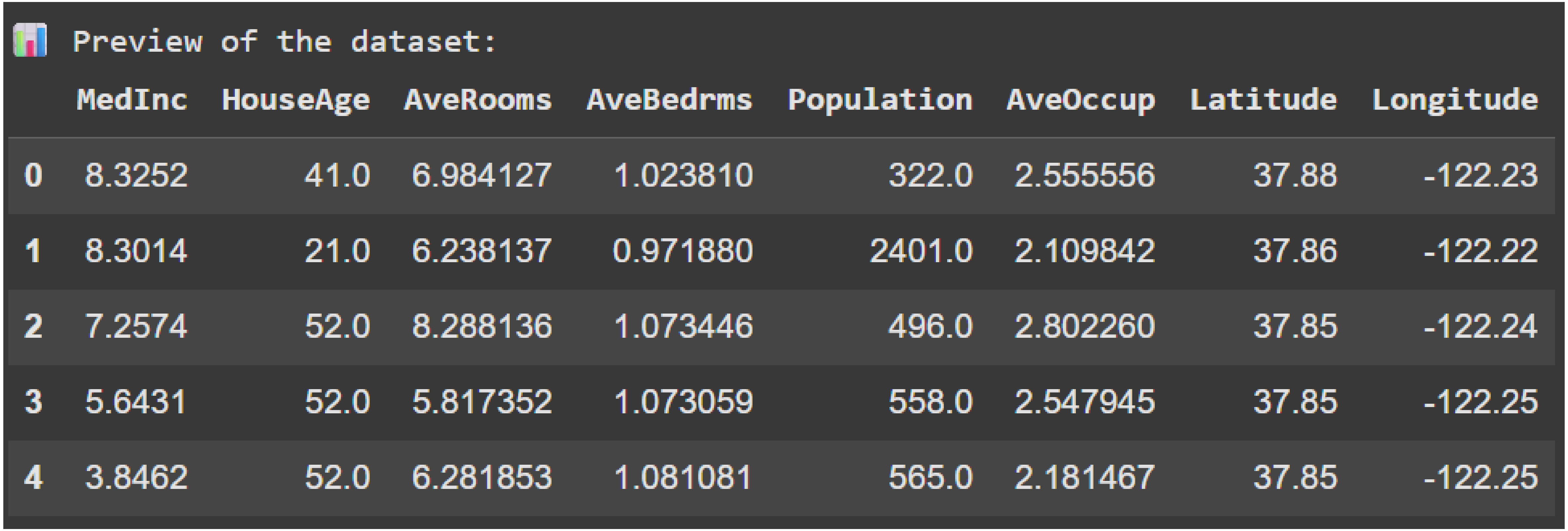
# Learning Objectives

- Preparing data (inspection, visualization, train/test splits)
- Building a machine learning model (Decision Tree)
- Choosing metrics to assess model performance
- Using the concept of bias–variance trade-off to tune hyperparameters
- **BONUS.** Reducing variance using ensembling

**Link to Colab:** https://tinyurl.com/dlbs1p2

# Preparing Data

California Housing Dataset

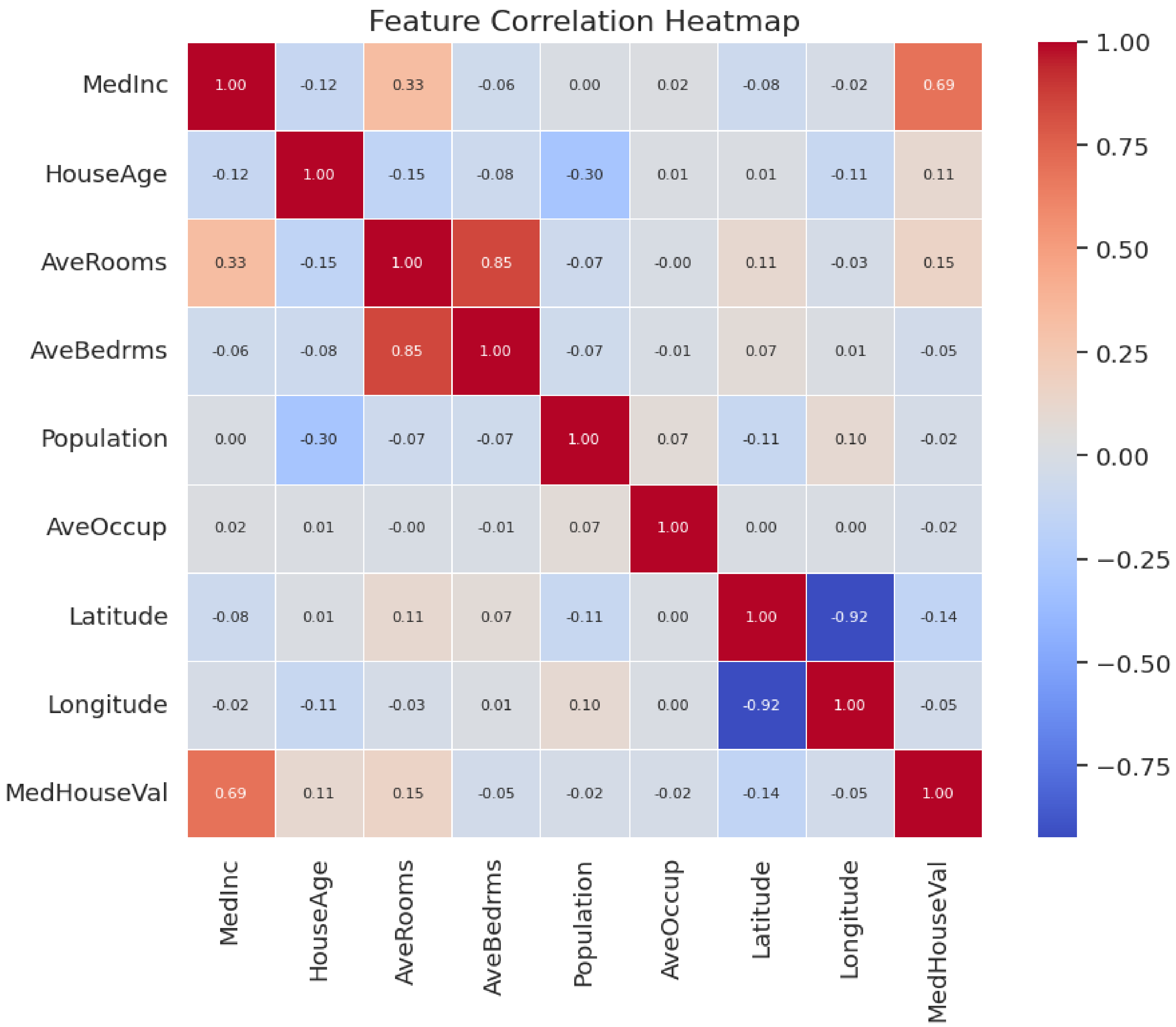


```
📊 Preview of the dataset:
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  Longitude
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88    -122.23
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86    -122.22
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85    -122.24
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85    -122.25
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85    -122.25
```

# Activity 1

**Q1.** Which features are most correlated with the target?

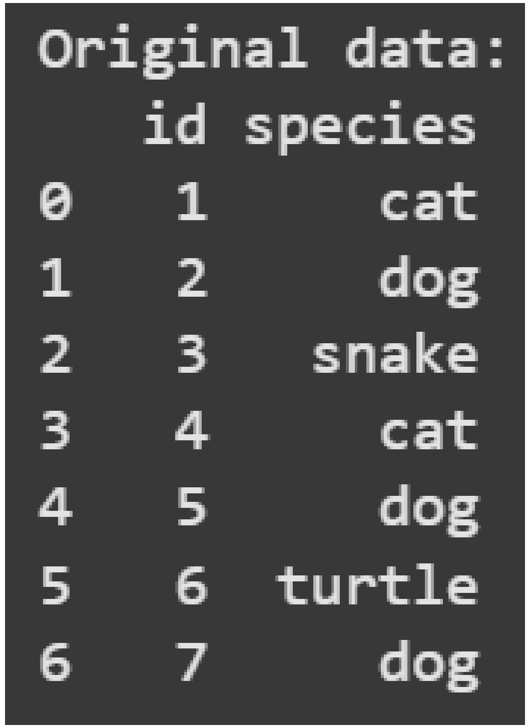
**Q2.** Which features might be redundant or overlapping?



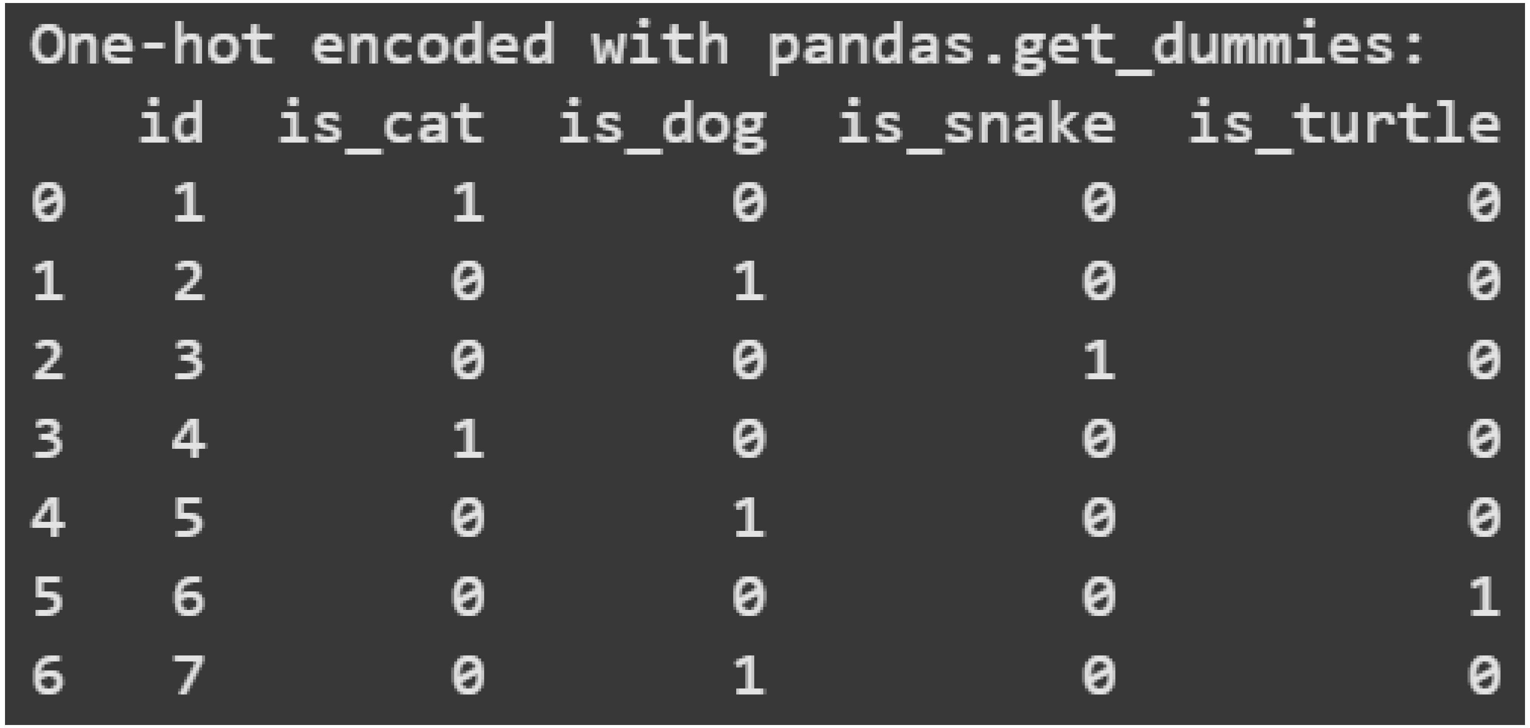
Feature Correlation Heatmap

| | MedInc | HouseAge | AveRooms | AveBedrms | Population | AveOccup | Latitude | Longitude | MedHouseVal |
|---|---|---|---|---|---|---|---|---|---|
| MedInc | 1.00 | -0.12 | 0.33 | -0.06 | 0.00 | 0.02 | -0.08 | -0.02 | 0.69 |
| HouseAge | -0.12 | 1.00 | -0.15 | -0.08 | -0.30 | 0.01 | 0.01 | -0.11 | 0.11 |
| AveRooms | 0.33 | -0.15 | 1.00 | 0.85 | -0.07 | -0.00 | 0.11 | -0.03 | 0.15 |
| AveBedrms | -0.06 | -0.08 | 0.85 | 1.00 | -0.07 | -0.01 | 0.07 | 0.01 | -0.05 |
| Population | 0.00 | -0.30 | -0.07 | -0.07 | 1.00 | 0.07 | -0.11 | 0.10 | -0.02 |
| AveOccup | 0.02 | 0.01 | -0.00 | -0.01 | 0.07 | 1.00 | 0.00 | 0.00 | -0.02 |
| Latitude | -0.08 | 0.01 | 0.11 | 0.07 | -0.11 | 0.00 | 1.00 | -0.92 | -0.14 |
| Longitude | -0.02 | -0.11 | -0.03 | 0.01 | 0.10 | 0.00 | -0.92 | 1.00 | -0.05 |
| MedHouseVal | 0.69 | 0.11 | 0.15 | -0.05 | -0.02 | -0.02 | -0.14 | -0.05 | 1.00 |

# **Transforming Data** – *One-Hot Encoding*

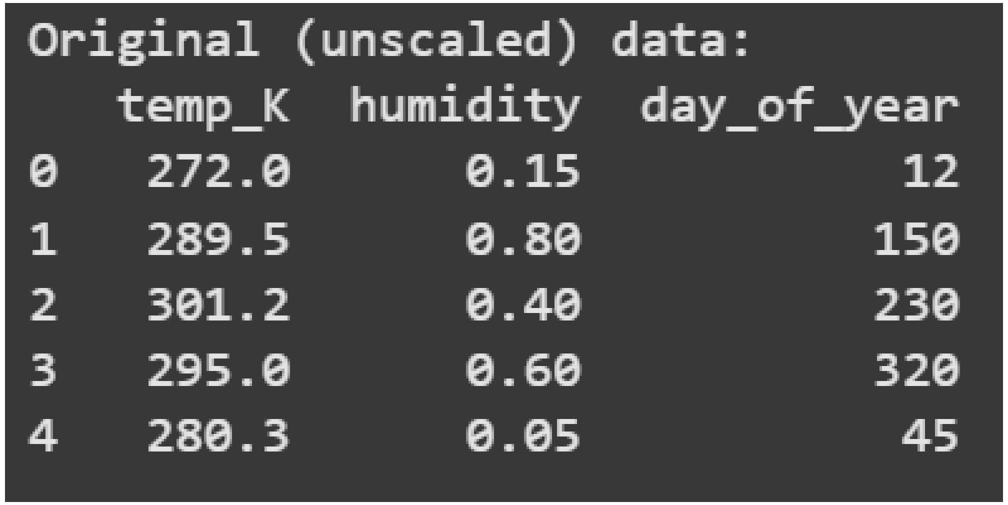


```
Original data:
   id species
0   1     cat
1   2     dog
2   3   snake
3   4     cat
4   5     dog
5   6  turtle
6   7     dog
```

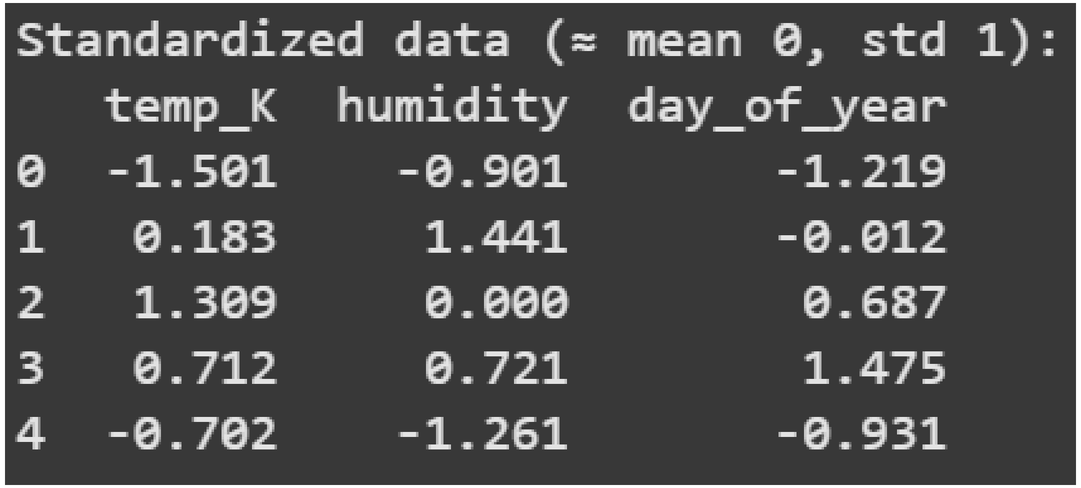


```
One-hot encoded with pandas.get_dummies:
   id  is_cat  is_dog  is_snake  is_turtle
0   1       1       0         0          0
1   2       0       1         0          0
2   3       0       0         1          0
3   4       1       0         0          0
4   5       0       1         0          0
5   6       0       0         0          1
6   7       0       1         0          0
```

# Transforming Data – *Feature Scaling*

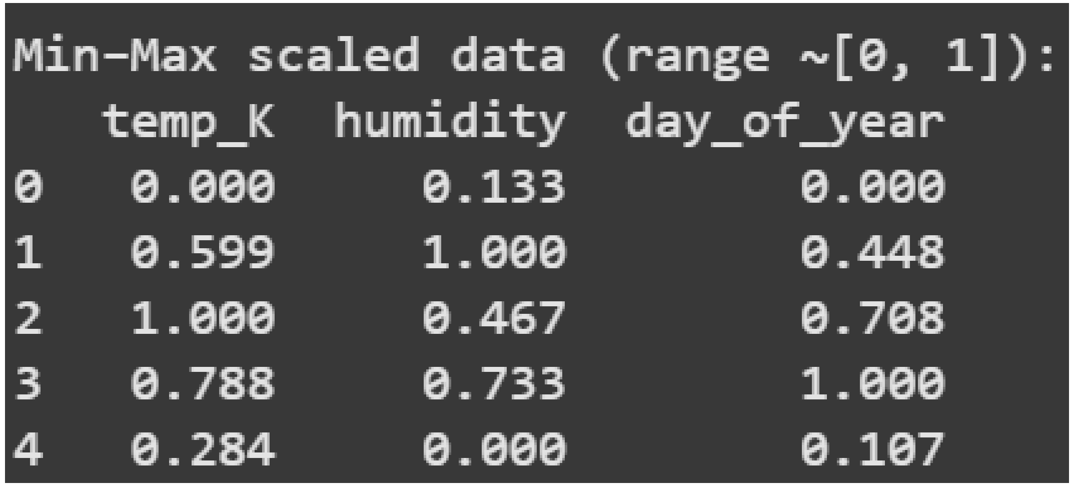


```
Original (unscaled) data:
   temp_K  humidity  day_of_year
0   272.0      0.15           12
1   289.5      0.80          150
2   301.2      0.40          230
3   295.0      0.60          320
4   280.3      0.05           45
```



```
Standardized data (≈ mean 0, std 1):
   temp_K  humidity  day_of_year
0  -1.501    -0.901       -1.219
1   0.183     1.441       -0.012
2   1.309     0.000        0.687
3   0.712     0.721        1.475
4  -0.702    -1.261       -0.931
```

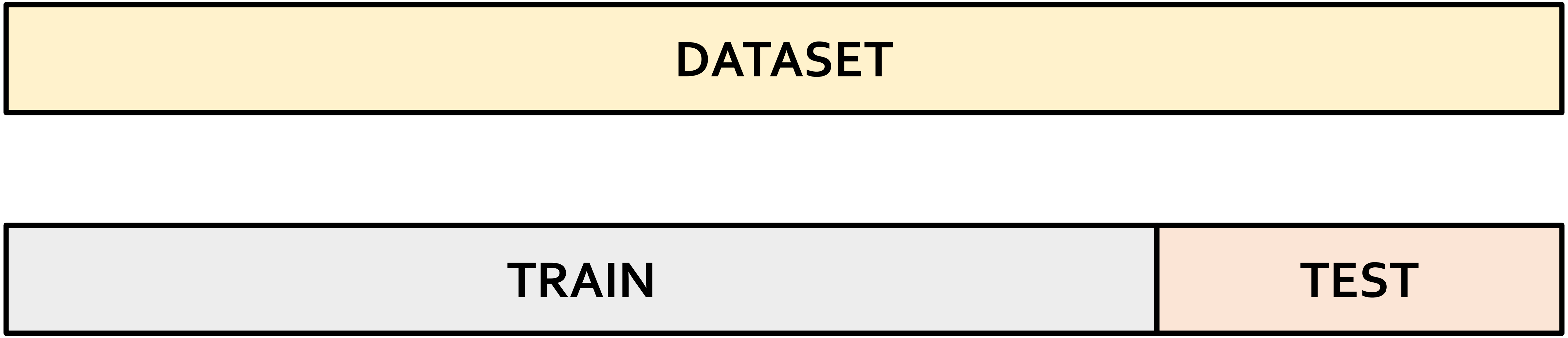
Standardization



```
Min-Max scaled data (range ~[0, 1]):
   temp_K  humidity  day_of_year
0   0.000     0.133        0.000
1   0.599     1.000        0.448
2   1.000     0.467        0.708
3   0.788     0.733        1.000
4   0.284     0.000        0.107
```

Min-Max Scaling

# Splitting Data for Training and Testing

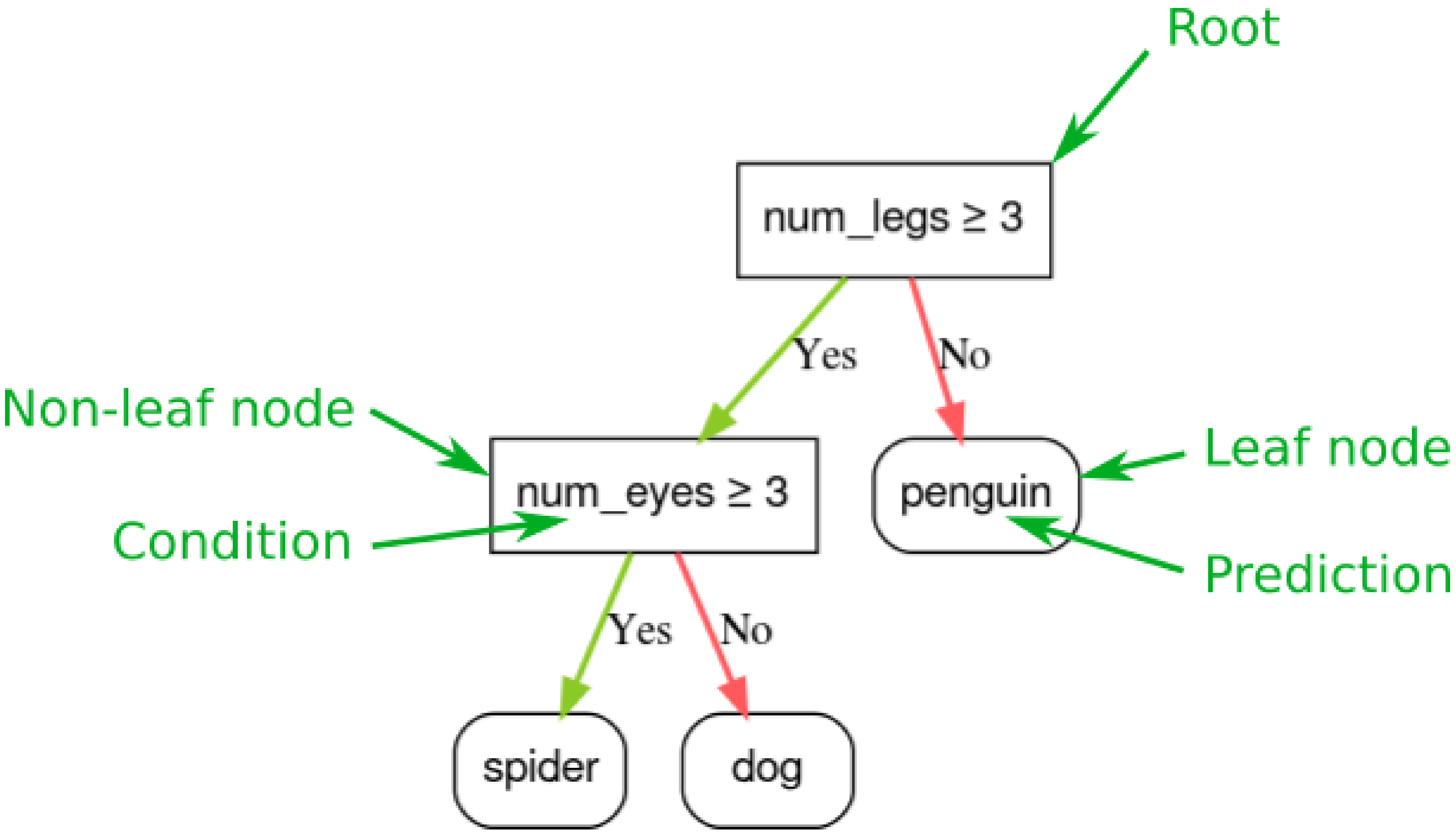


DATASET

TRAIN

TEST

# Choosing a Machine Learning Model – *Decision Tree*



https://developers.google.com/machine-learning/decision-forests/decision-trees

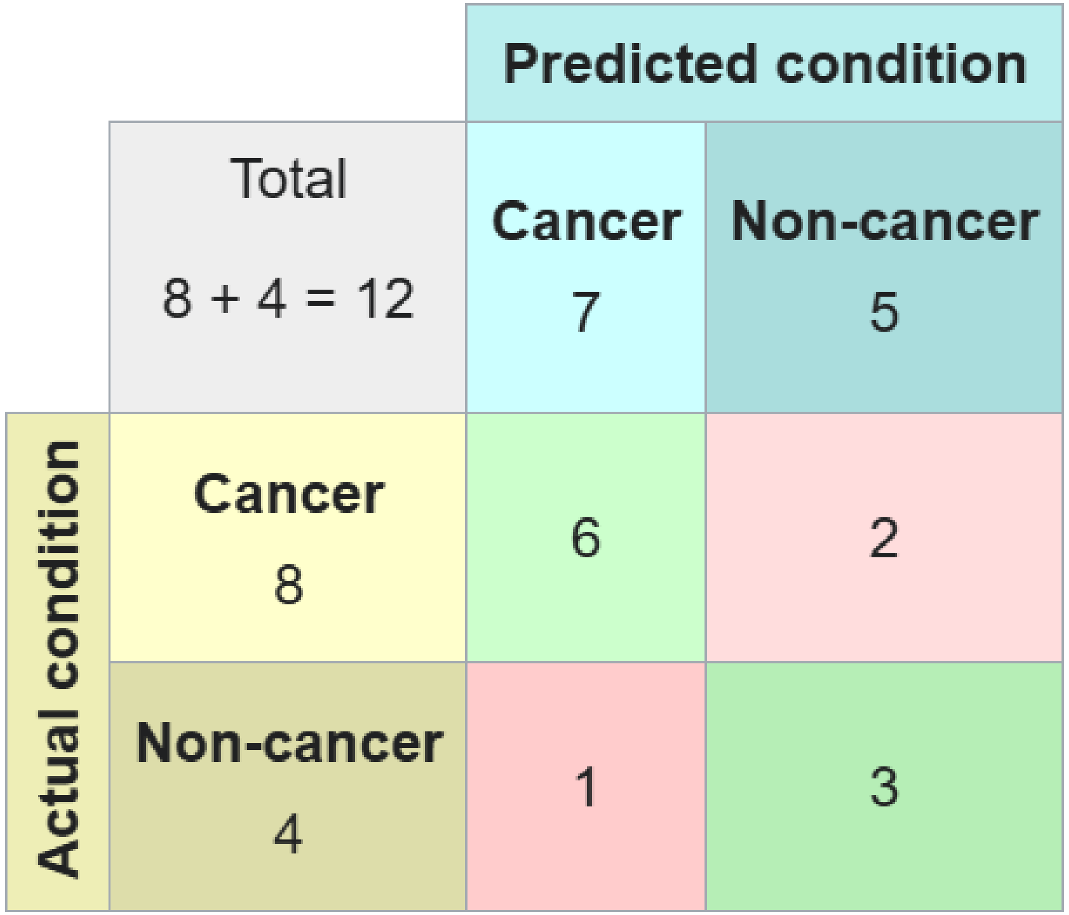
# Model Evaluation Metrics – *Regression*

$$\text{RMSE}(y_{\text{true}}, y_{\text{pred}}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

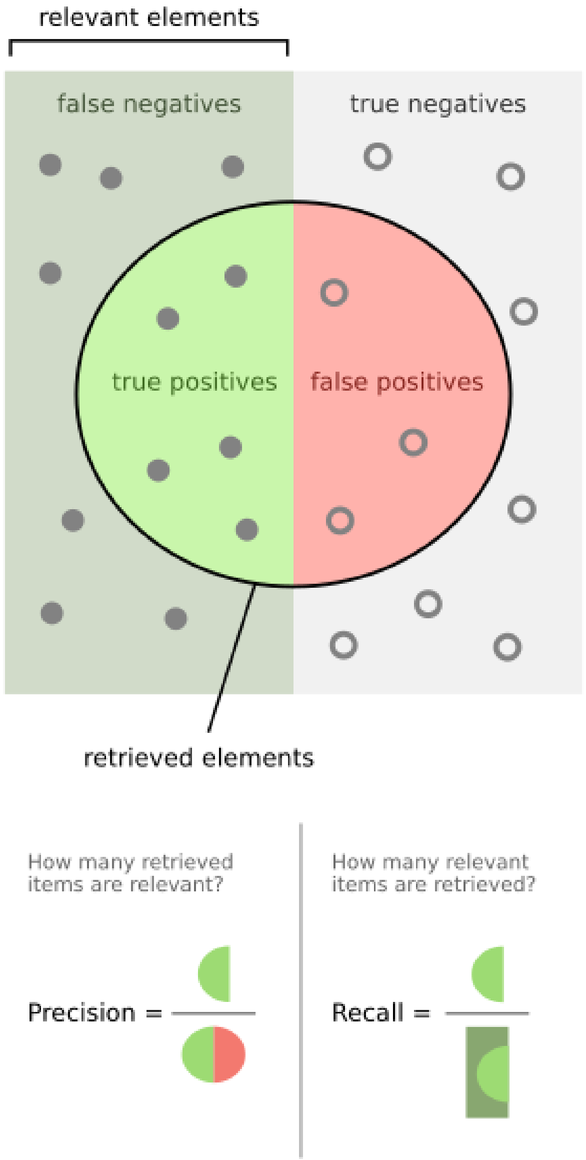
$$R^2(y_{\text{true}}, y_{\text{pred}}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

**Activity 2.** Write a function to compute RMSE for train and test splits

# Model Evaluation Metrics – *Classification*



| | | Predicted condition | |
|---|---|---|---|
| | Total<br>8 + 4 = 12 | **Cancer**<br>7 | **Non-cancer**<br>5 |
| **Actual condition** | **Cancer**<br>8 | 6 | 2 |
| | **Non-cancer**<br>4 | 1 | 3 |

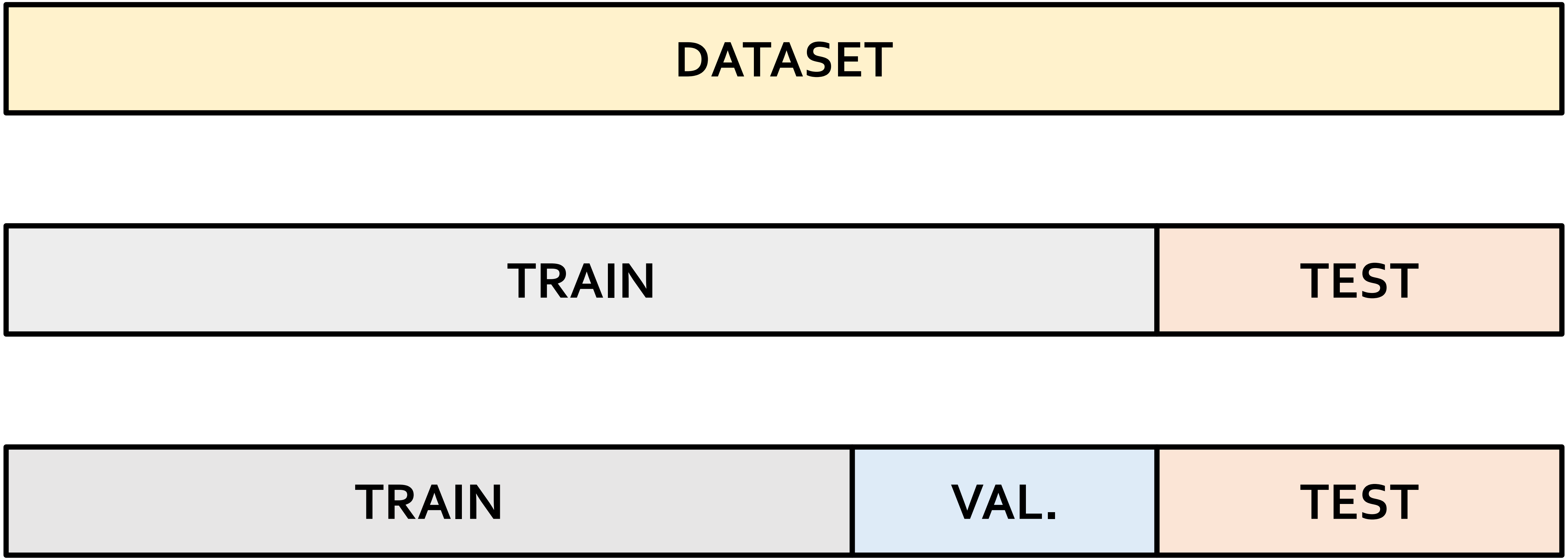


# Hyperparameter Tuning

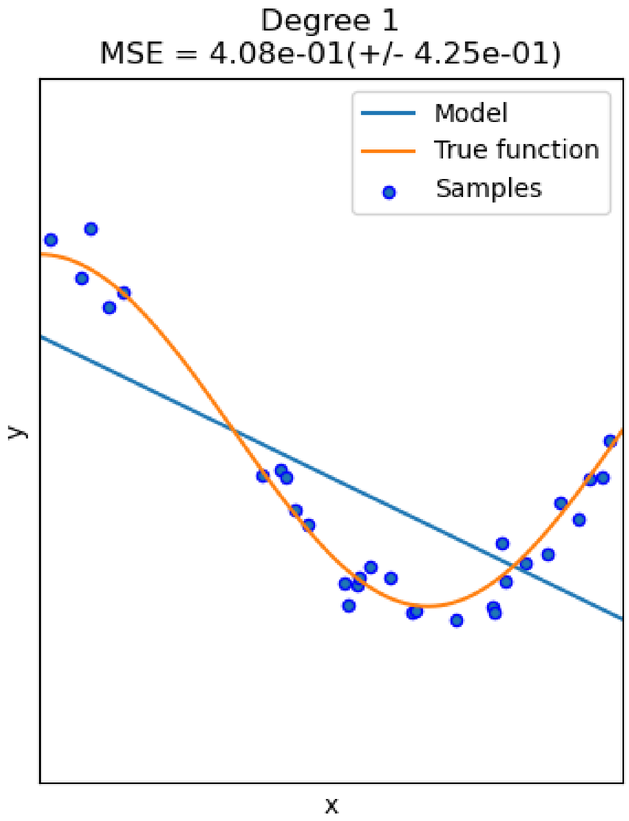
- **Hyperparameter.** A setting you choose before training that controls how the model learns. It's not *learned* from the data.
- Decision tree hyperparameters:
  - `max_depth` → limits how many "levels" the tree can grow
  - `min_samples_split` → minimum samples required to split a node
  - `min_samples_leaf` → minimum samples required in any leaf criterion

**Activity 3.** Which `max_depth` is the best?

# Splitting Data – *Enter Validation Split*

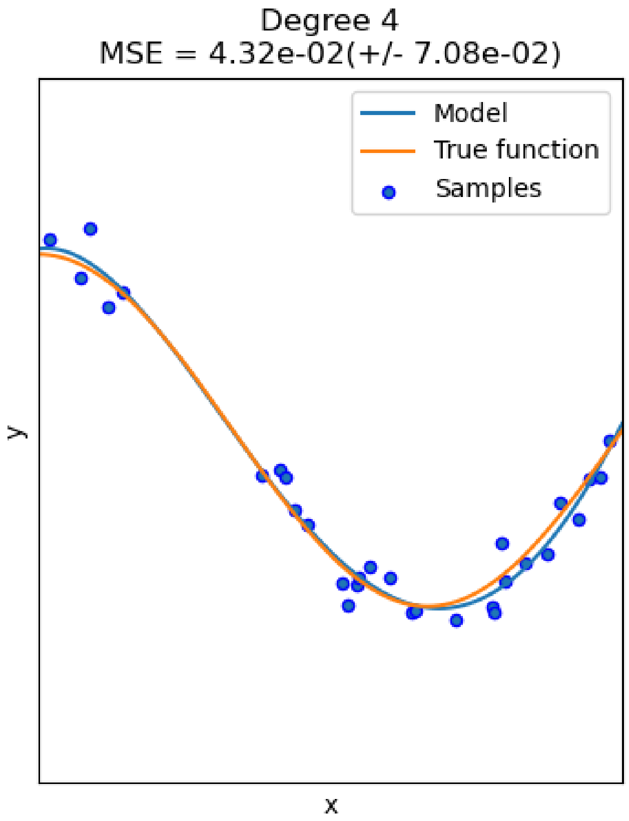


# Bias-Variance Tradeoff

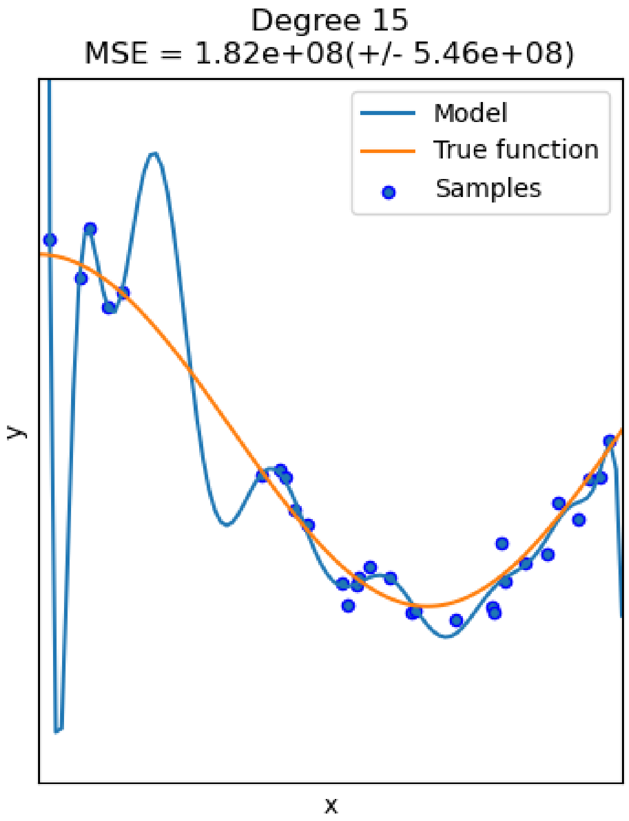


**High bias**

(Underfitting)



Good Fit

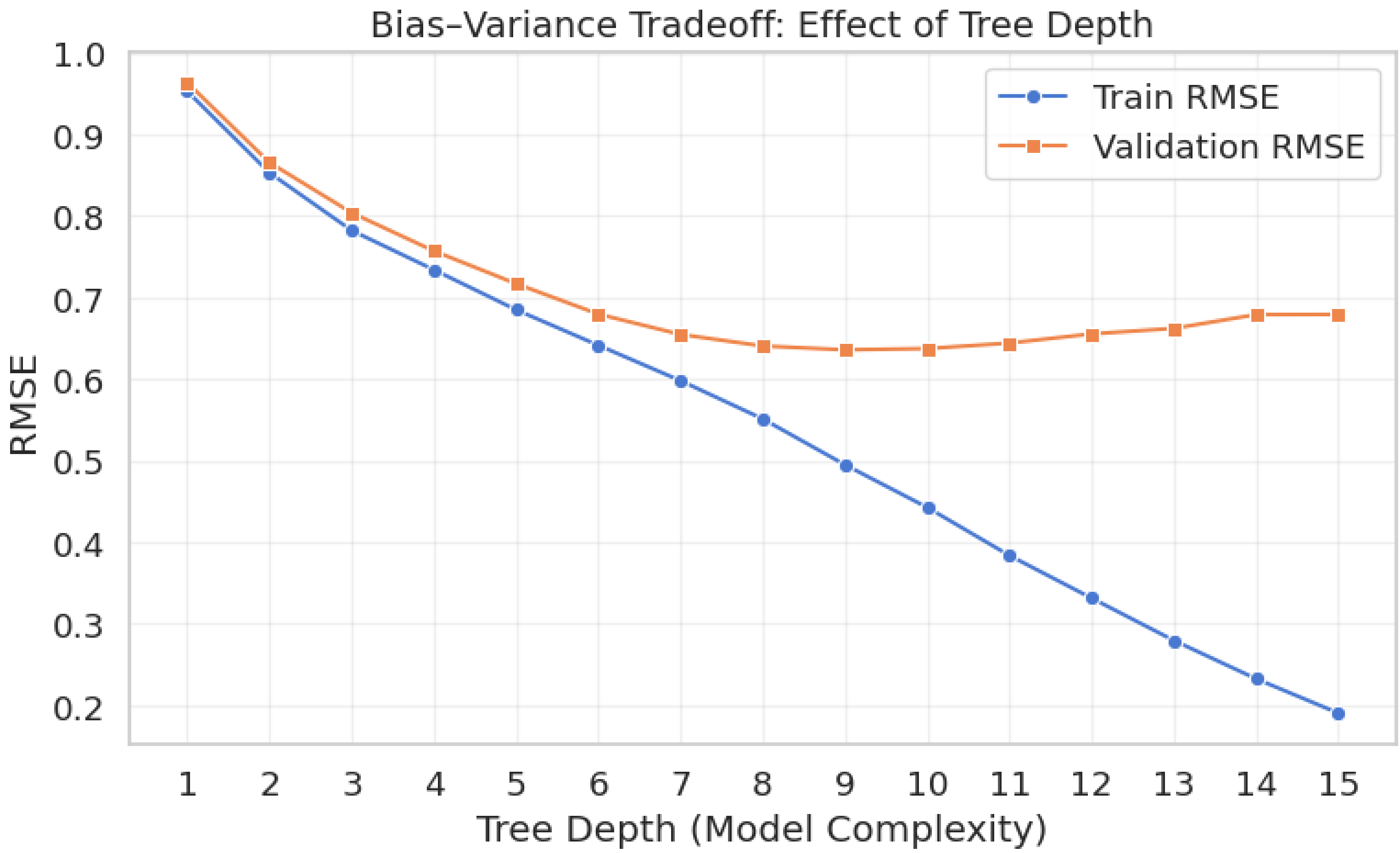


**High Variance**

(Overfitting)

https://scikit-learn.org/stable/modules/learning_curve.html

# Validation Curves



**Activity 4.** Which `max_depth` is the best?

# Summary

**Blueprint of a Machine Learning Project**

1. Define your prediction goal (classification/regression)
2. Explore and understand your dataset
3. Split the data into train/test/validation
4. Pick a model, train it, and measure its performance
5. Tune hyperparameters and optimize model through the lens of bias–variance trade-off.

# We'd love to hear your feedback

Help us make the next session even better!

3rd DL Bootcamp (2025-26) - Session 1 - Post Session Feedback

# References / Reading

- Python introduction: https://swcarpentry.github.io/python-novice-inflammation/
- More on scientific python: https://lectures.scientific-python.org/
- https://deeplearning.neuromatch.io/
- NeuroMatch Academy: https://deeplearning.neuromatch.io/tutorials/W1D1_BasicsAndPytorch/chapter_title.html
- Exploratory computing w/ Python: https://mbakker7.github.io/exploratory_computing_with_python/
- https://udlbook.github.io/udlbook/
  - Reuse slides from there

See you in the next session

DEEP LEARNING BOOTCAMP

# DEEP LEARNING ESSENTIALS

## BOOTCAMP SESSION-2

Deep Learning for Regression and Classification with PyTorch

**SRIRAM RANGA**

**SANTHISENAN AJITH**

02:30 PM, NOV 10TH, 2025

THE ARC TR+30

BRING YOUR LAPTOP !!!!

Scan Me

| Upcoming Sessions | Date | Speakers |
|---|---|---|
| Deep Learning for Images | 17 Nov 2025 | Micheal Yuhas & Amashi Niwarthana |
| Natural Language Processing (NLP) | To be decided | To be decided |