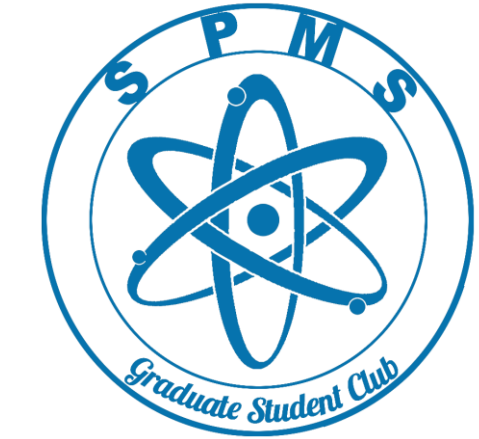
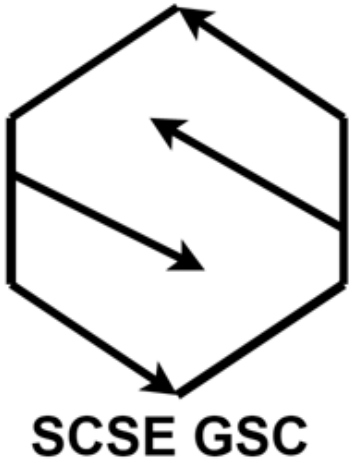
Deep Learning Essentials

Eduardo de Conto

Yasharth Yadav





Agenda

1. Quick intro to python data structures and control structures, functions (with hands on exercise) - 15-20 mins
2. Quick intro to NumPy and basic NumPy capabilities, with quick hands-on exercise – 20 mins
3. Linear Regression using NumPy (theory on Linear Regression with Hands-on exercise) - 30 mins
4. Exploratory Data Analysis on Housing Dataset (Basic plotting) - 30min
5. Closed form Linear Regression on Housing Dataset using scikit-learn - 30min
6. Gradient Descent Linear Regression on Housing Dataset using PyTorch (but without GPU support) - 10min

Plan for the afternoon

Time	Activity
14:00 – 14:30	Introduction (30’)
14:30 – 15:45	Work session in groups (1h15) - 45min intro - 30min hands-on linear regression
15:45 – 16:00	Break
16:00 – 17:00	Work session in groups (1h)

Future Schedule

Event No.	Details
Event 1 **26nd Jan 2024**	*Deep Learning Essentials* Covers the basics of Python and necessary packages required for Deep Learning such as numpy, scipy, pandas etc.
Event 2 **09th Feb 2024**	*Deep Learning for Regression and Classification* It should cover the basics of PyTorch, as well as how to use PyTorch for performing regression and classification tasks.
Event 3 **23rd Feb 2024**	*Deep Learning for Images* In this event, we will extend the classification using deep learning, specifically focusing on datasets involving images.
Event 4 **15th Mar 2024 - ??**	*Deep Learning for Sequence Data (text and time series)* 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 **28th Mar 2024**	*Hackathon* The purpose of this final event is to apply the skills acquired in previous sessions to solve a real-world problem within a specified time frame.

About Hackathon

Hackathon

Hackathon Prizes

- Winner : SGD 300
- Runner Up : SGD 150
- Honorable Mention : SGD 50
- Participation Prizes : 10 x Starbucks / FairPrice vouchers

Team Setup

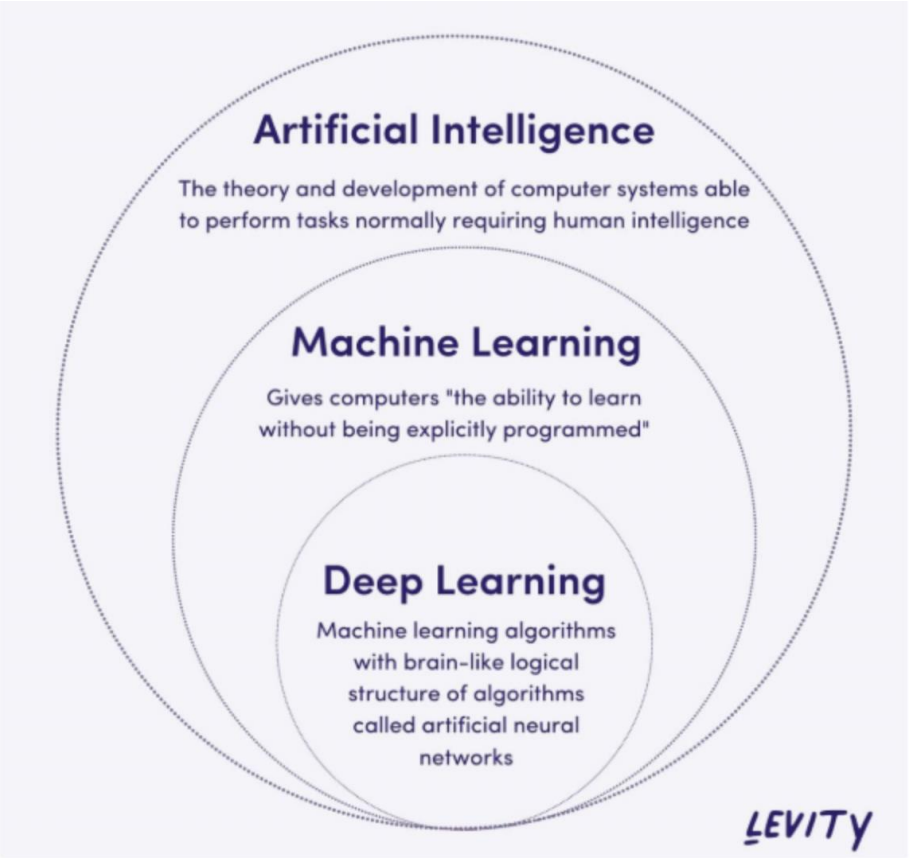
- The hackathon is open to individuals who have attended at least 3 out of 4 sessions of the DL Bootcamp.
- Teams can be formed on the spot or come prepared with a pre-established group.
- Each team must comprise of min 4 to max 5 members.

What is Deep Learning?

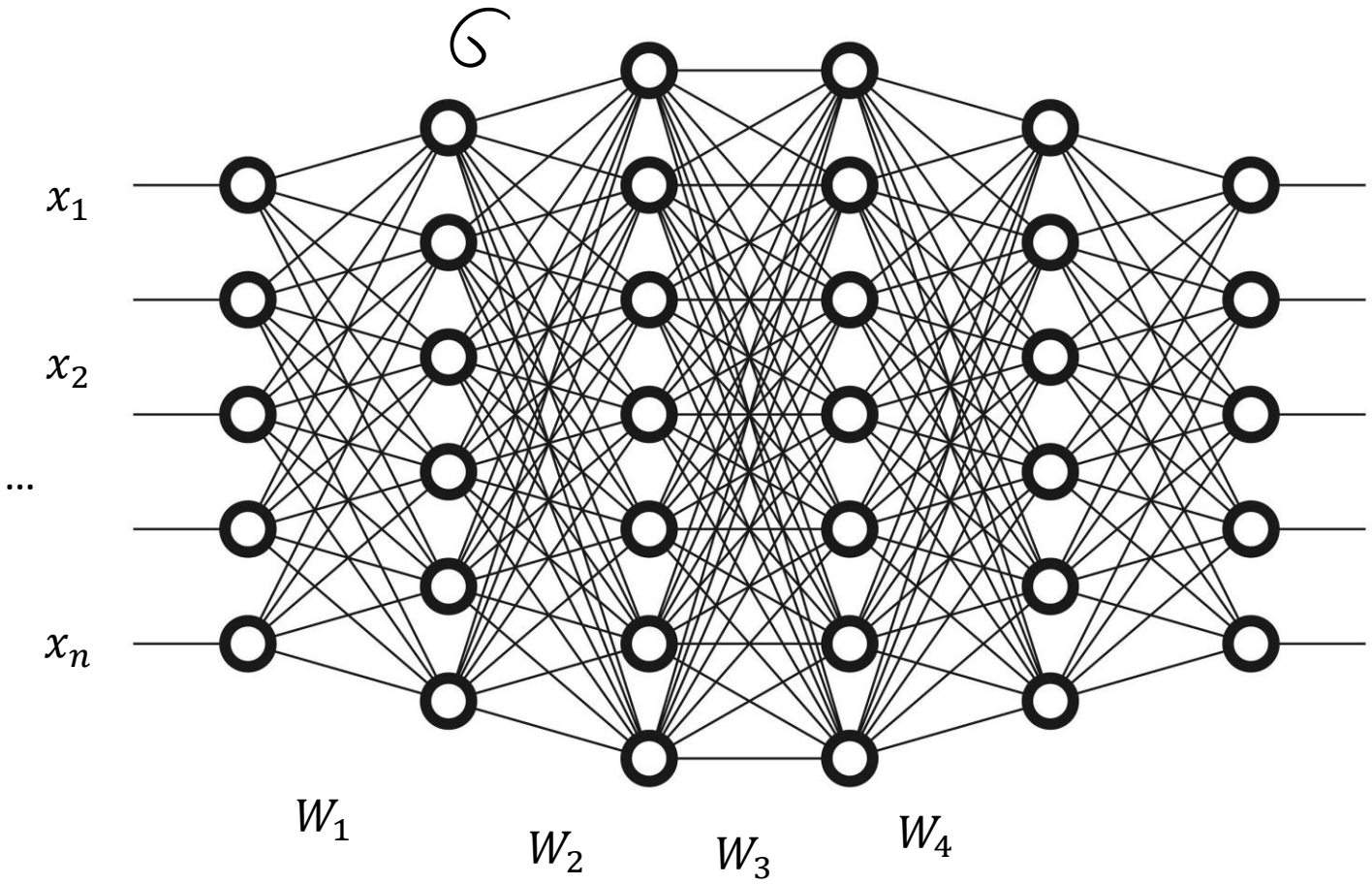
- Machine learning: learn from examples or data
- Deep Learning: use artificial neural networks: weight * variable + some non-linear layer

`

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



Deep Learning



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

y

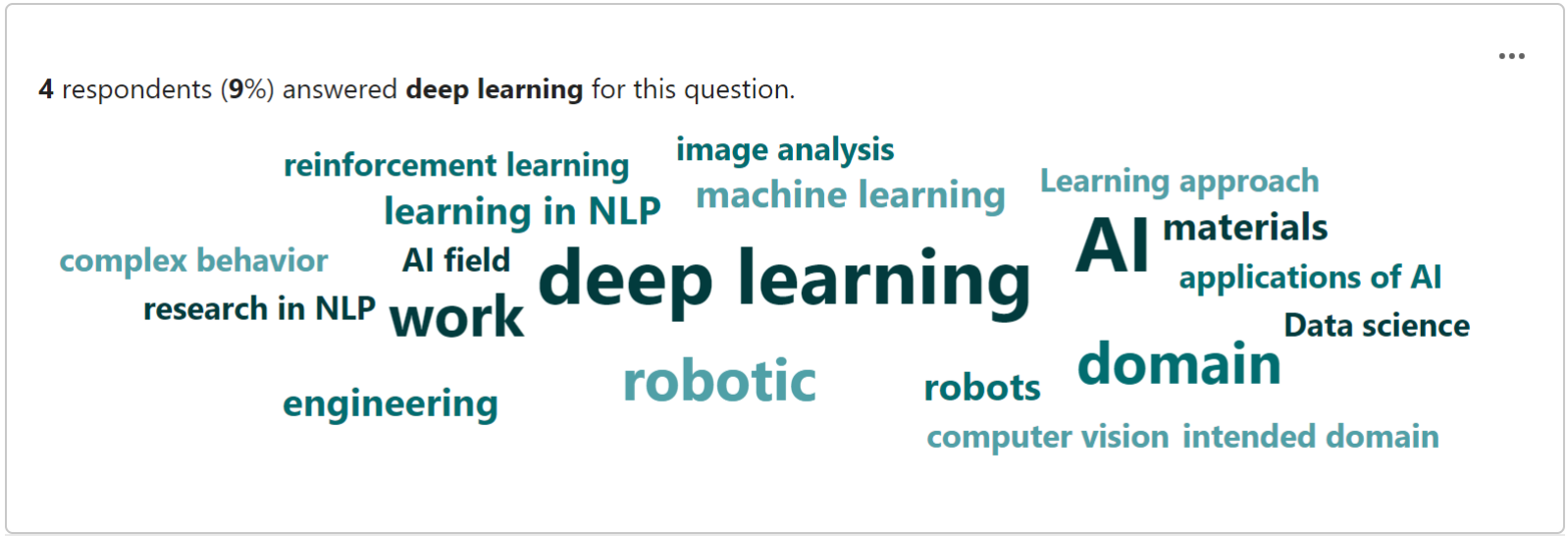
https://cdn-images-1.medium.com/max/2400/1*1mpE6fsq5LNxH31xeTWi5w.jpeg

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



Examples

You

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

ChatGPT

For predicting material properties using deep learning, you might consid
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):

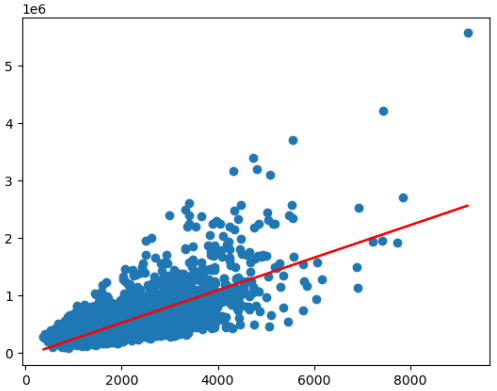
python

```
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
  - θ are the parameters

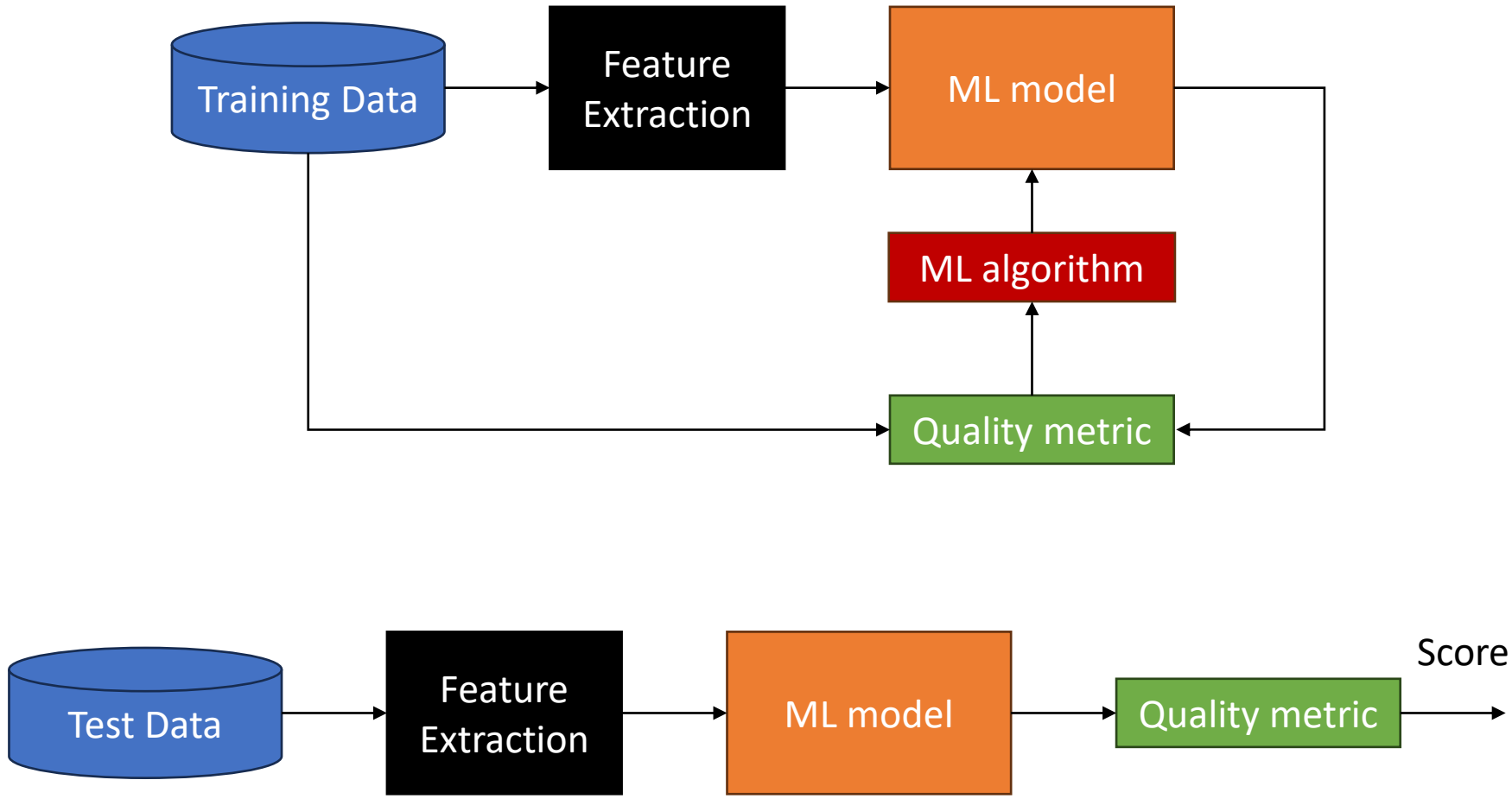
- Two typical problems
  - Classification: discrete categories
  - Regression: continuous categories



Pro & Cons of DL

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

ML workflow

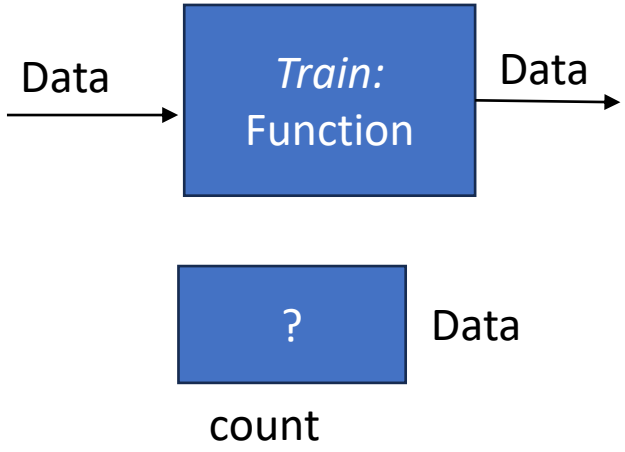


https://www.coursera.org/learn/ml-foundations?specialization=machine-learning

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

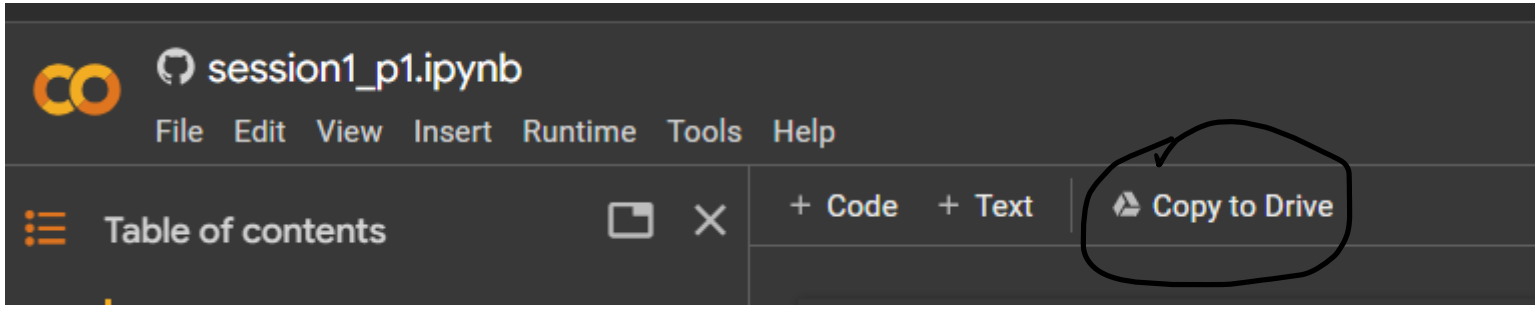



```
def vowel_count(word):
    vowels = ["a", "e", "i", "o", "u"]
    count = 0
    for char in word: # loops
        if char in vowels: # conditions
            count += 1
    return count

vowel_count("hello")
```

MyModel: Class
my_model: Data1 *params: Data2* …
Train: Function1 *Test: Function2* …

Google Colab: Our tool for today



Can you open the notebooks?

Work session 1.0

Handling numbers & data: numpy

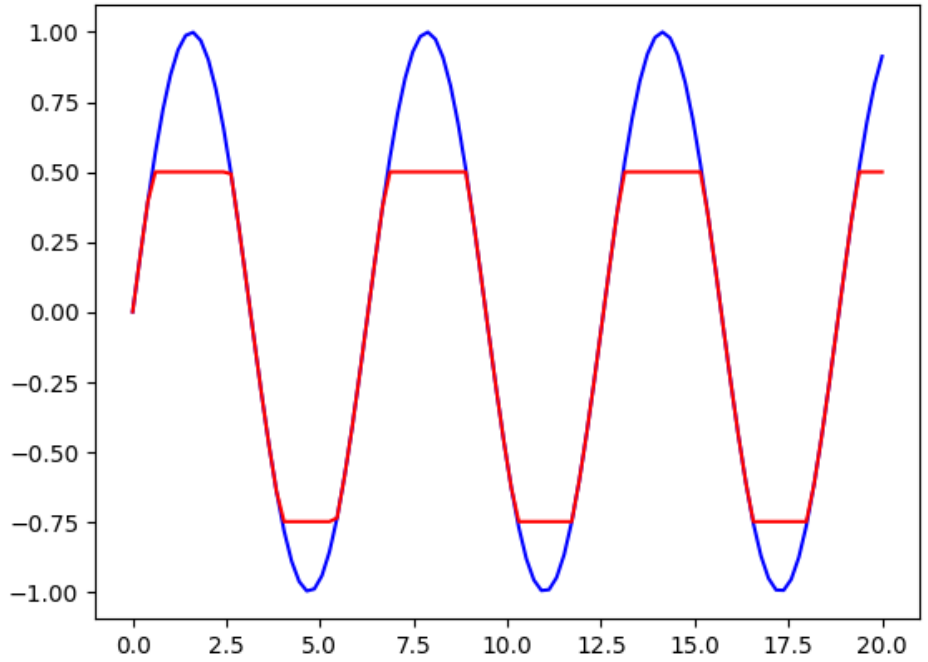
- Numpy is library for handling array, vector & matrices

- 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.



Work Session 1.1

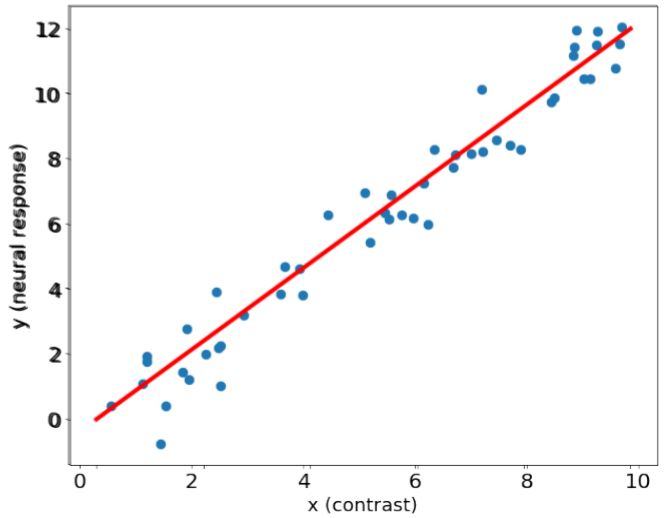
Linear Regression

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 — θ_1
contrast — x
Intercept — θ_0

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



Linear Regression: MSE

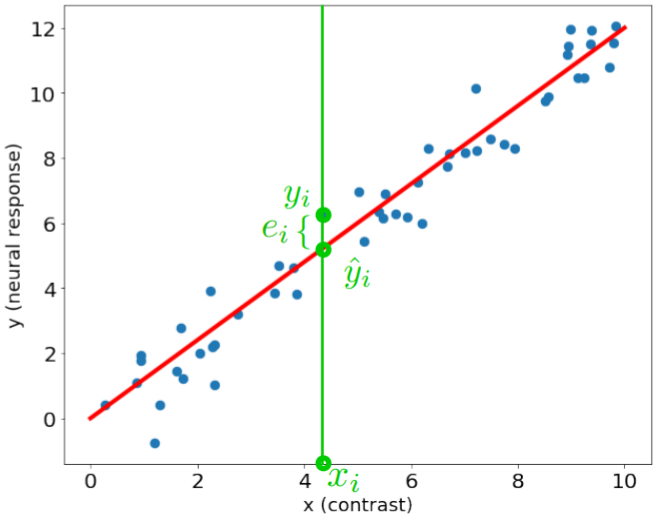
$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} (y_i - \theta x_i)^2$$

Mean Squared Error (MSE)

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

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

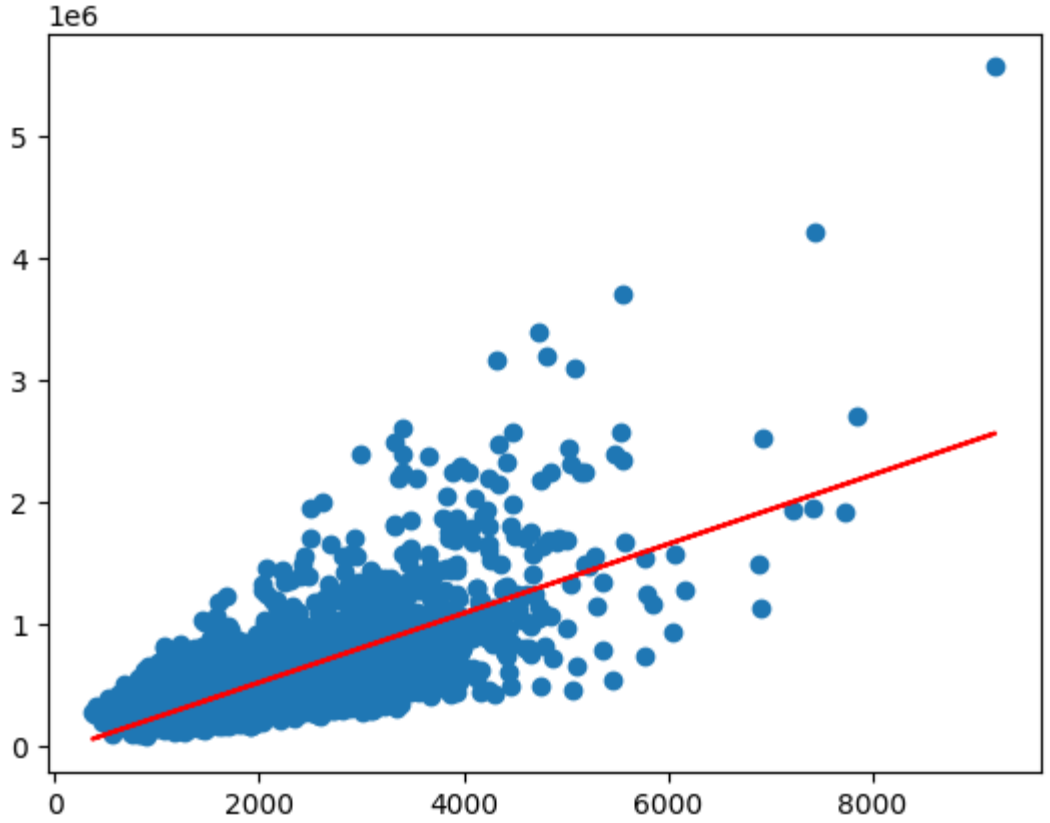
- total number of data points
- true neural response
- model prediction
- index of data points i=1,...,N
- residual



Work session 2

Housing data exploration

- Let’s explore a housing data
- Relevant plots
- Linear regression model
  - Scikit-learn
  - PyTorch



Deep Learning & Pytorch

- Pytorch one of the most popular libraries for deep learning

- Components of a DL model:
  - The model itself: the structure of the model
  - A loss function (error metric)
  - An optimization algorithm
  - The training loop

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/

Please provide feedbacks!