

LAB

Python for Machine Learning II

link.jpw.info/6

Course Announcements

Project

Milestone 1 due 3/27

- Submit *Methodology* draft to Canvas

Code Demos next week

- Indicate preferred presentation date on [Teams Spreadsheet](#)

Project Check-Ins 4/1, 4/3

- Indicate date on [Team Spreadsheet](#)

Assignments

Assignments 1, 2 graded

No more assignments for a while!

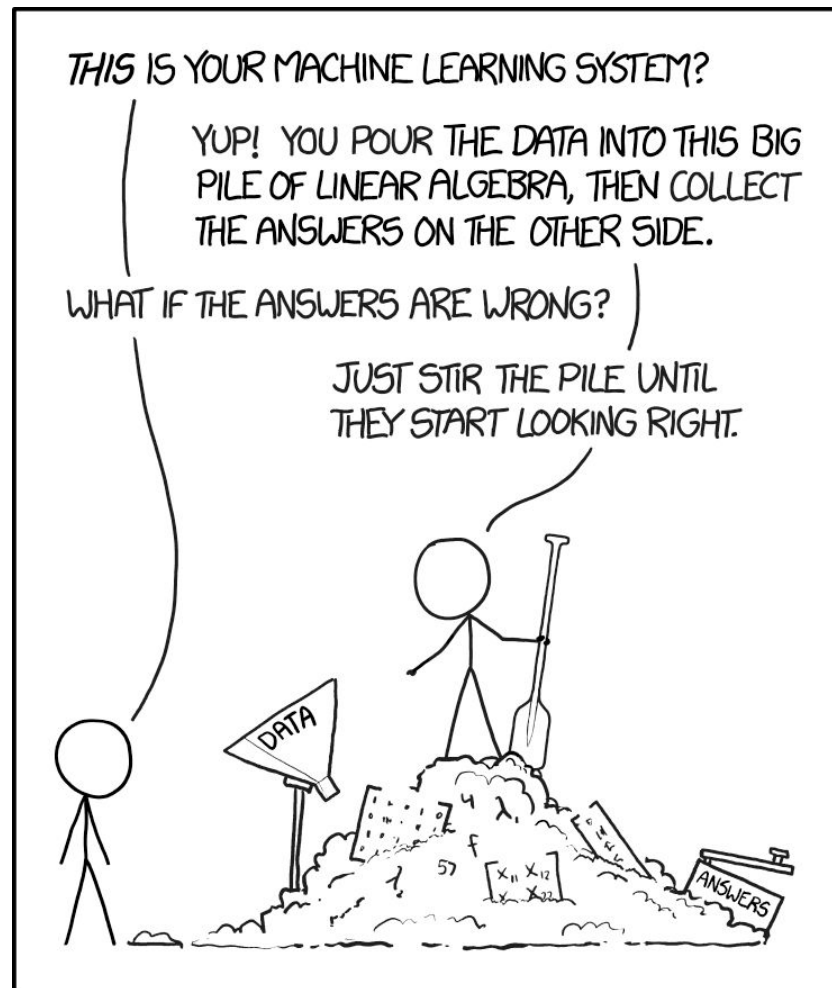
After Spring Break:

Assignment 3 due 4/17

ML Pipeline

Assignment 4 due 4/24

AI Ethics & Policy



The “Data Science Process” (Pt. II)

Explore

- **Get & import** your data
- **Understand** the data's structure, features
- **Exploratory Data Analysis**
 - *What relationships are worth investigating?*

Transform

- **Clean** the data
 - *How should I deal with missing data?*
- **Feature Extraction**
 - *What information is most valuable?*

Apply

- **Train model** on selected features
- **Cross-validate** on different data splits
- **Test** on unseen data
- **Evaluate** results and generalizability

But First...

Part 0

What Are Data?

0. Storing Data

What are data?

- Data are **information**
- Data are **information structured in a consistent manner**



Tables!

Tables consist of **arrays**
and associated **labels**

	Name	Dog?	Breed	Energy
0	Alfie	True	Labrador Retriever	9
1	Babbles	False	Domestic Short Hair	4
2	Banjo	True	Cattle Dog	10
3	Clay	True	German Pointer	7
4	Cookie	False	Domestic Short Hair	2
5	Milky Way	False	Domestic Short Hair	6
6	Moondust	True	Terrier	5
7	Oli	True	Beagle	3
8	Sam	True	Pit Bull	6
9	Pumpkin	False	Domestic Short Hair	5

0. Storing Data

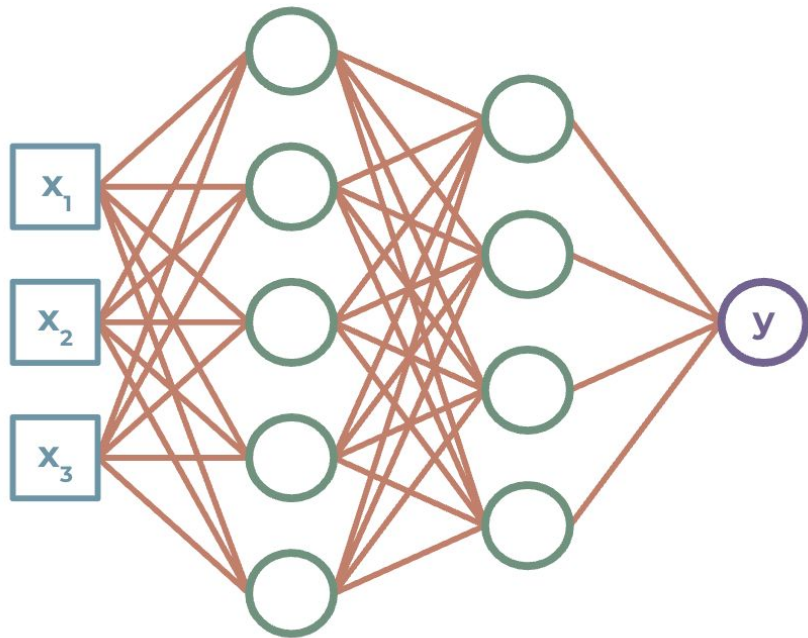
Are tables necessary?

- We're dealing with **a lot of data**
- We can leverage **consistency in data structure**



Tables!

Matrices



Part 1

Arrays and Tensors

Arrays and Tensors

numpy arrays

Store **values of the same type**

Can handle **arbitrary sizes and dimensions**

scalar → array

```
np.array([...])
```

Operations are **fast (ish)**

torch tensors

Store **values of the same type**

Can handle **arbitrary sizes and dimensions**

**everything
is a tensor**

```
torch.tensor([...])
```

Parallelize computations on a
GPU (**super fast!**)

Arrays and Tensors



Part 2

sklearn and pytorch

sklearn

Most (if not all) sklearn ML models follow **this structure**:

```
model = MLModel(hyperparameters)
```

initialize

```
model.fit(X_train, y_train)
```

train

```
model.predict(X_test)
```

test

```
model.score(X_test, y_test)
```

evaluate

```
[82] lr_model = LogisticRegression(max_iter=1000)
      lr_model.fit(X, y)
```



```
LogisticRegression
LogisticRegression(max_iter=1000)
```

pytorch

Create custom neural networks with **pytorch**

```
[ ] class NeuralNetwork(nn.Module)
    def __init__(self):
        super().__init__()
        self.flatten = nn.Flatten()
        self.linear_relu_stack = nn.Sequential(
            nn.Linear(28*28, 512),
            nn.ReLU(),
            nn.Linear(512, 512),
            nn.ReLU(),
            nn.Linear(512, 10),
        )

    def forward(self, x):
        x = self.flatten(x)
        logits = self.linear_relu_stack(x)
        return logits

model = NeuralNetwork()
```

extend `nn.Module`

feedforward network

connect layers with
linear weight matrix

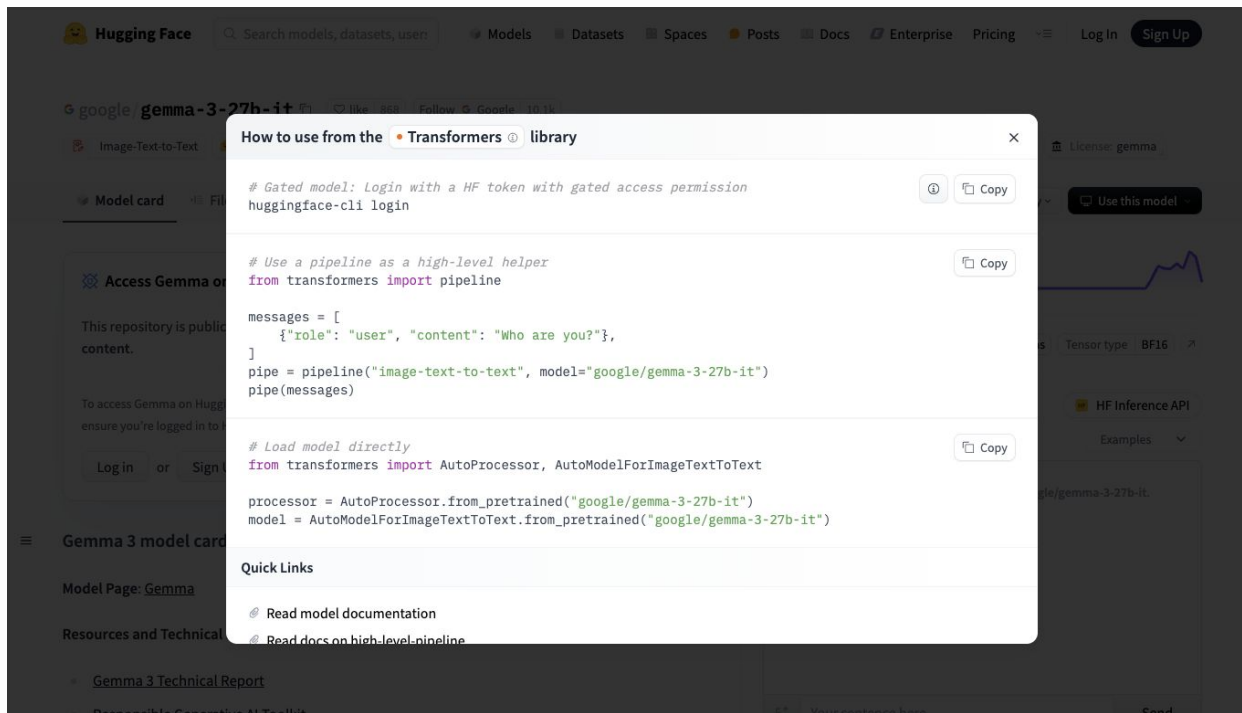
add activation functions

Part 3

Hugging Face 

Hugging Face 🤗

use the transformers library!



The screenshot shows the Hugging Face website interface. The top navigation bar includes the Hugging Face logo, a search bar, and links for Models, Datasets, Spaces, Posts, Docs, Enterprise, Pricing, Log In, and Sign Up. The main content area displays the 'Gemma 3' model card by Google. A modal window titled 'How to use from the Transformers library' is overlaid on the page, showing three methods to use the model:

- Gated model:** Login with a HF token with gated access permission
`huggingface-cli login`
- Use a pipeline as a high-level helper:**

```
from transformers import pipeline

messages = [
    {"role": "user", "content": "Who are you?"},
]
pipe = pipeline("image-text-to-text", model="google/gemma-3-27b-it")
pipe(messages)
```
- Load model directly:**

```
from transformers import AutoProcessor, AutoModelForImageTextToText

processor = AutoProcessor.from_pretrained("google/gemma-3-27b-it")
model = AutoModelForImageTextToText.from_pretrained("google/gemma-3-27b-it")
```

Each code block has a 'Copy' button. Below the code blocks is a 'Quick Links' section with links to 'Read model documentation' and 'Read docs on high-level pipeline'.

PYTHON FOR ML

Questions?