A/V Test

See my screen? Hear the bell?

Coding Machine Learning

Spandan Madan
Neuro 140/240

Learning Outcomes

Main Goal: Get you set up and running for coding ML projects.

- How to set up your code?
- What tools/languages/frameworks can help make our life easier?
- How does it connect with the Theory?

Learning Outcomes

My Amazing ML Project

Files in that project

```python
import torch
from torchvision import datasets

def train_model():

def evaluate_model():
```

Code in these files

Learning Outcomes

My Amazing ML Project

How is this box organized?
Where is it stored?
How do I track changes?
How do I share it with someone else?

Files in that project

What kind of files would I have?
How would these be opened?
Which ones need to be shared and how?

```
import torch
from torchvision import datasets

def train_model():

def evaluate_model():
```

Code in these files

What language/frameworks should I use?

Part 1: The Box of Code

My Amazing ML Project

- **How is your project organized?**
- **Where is it stored?**
- **How do you track changes?**
- **How do you share it with someone else?**

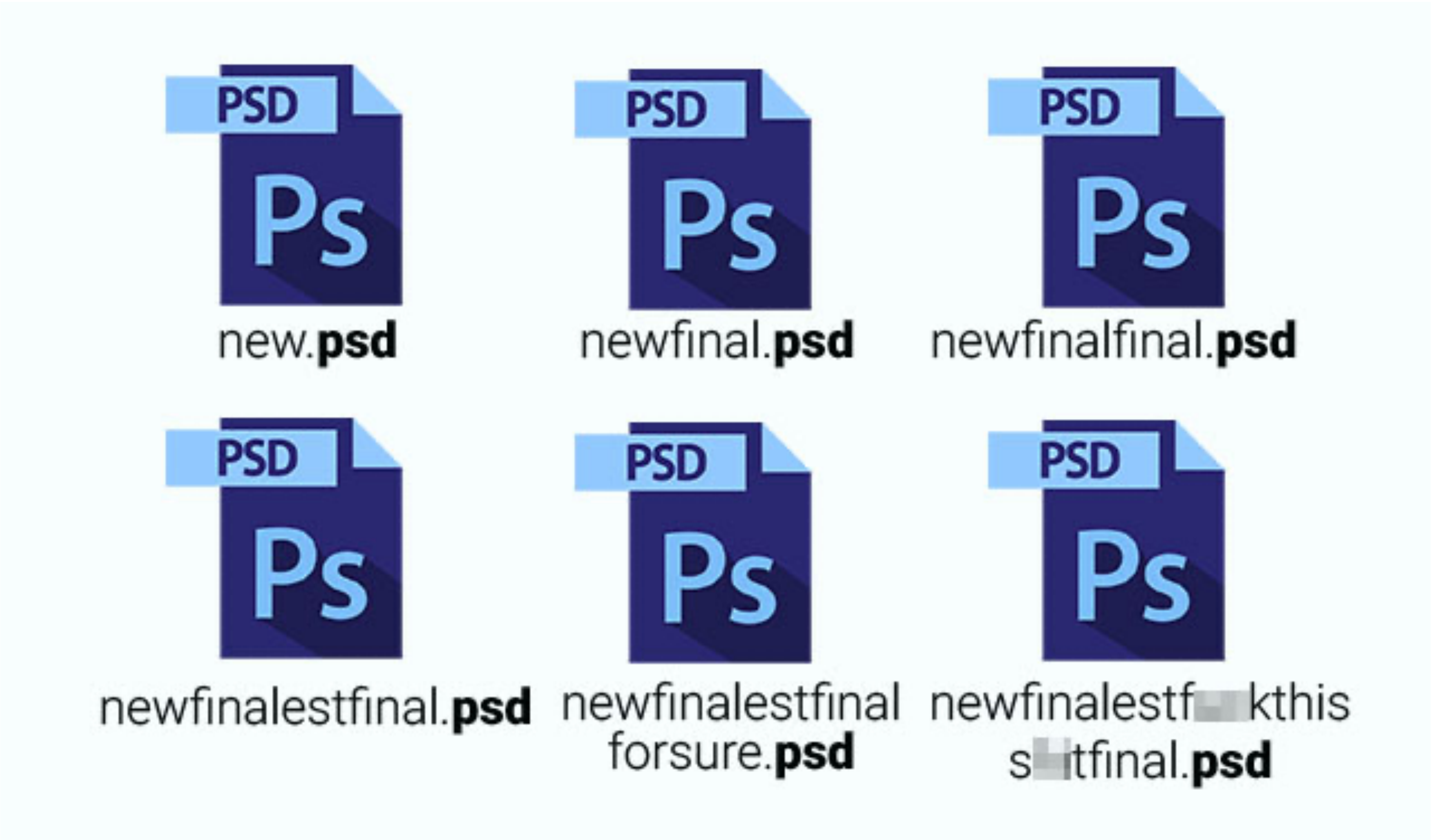
Github

GitHub = Git + Hub

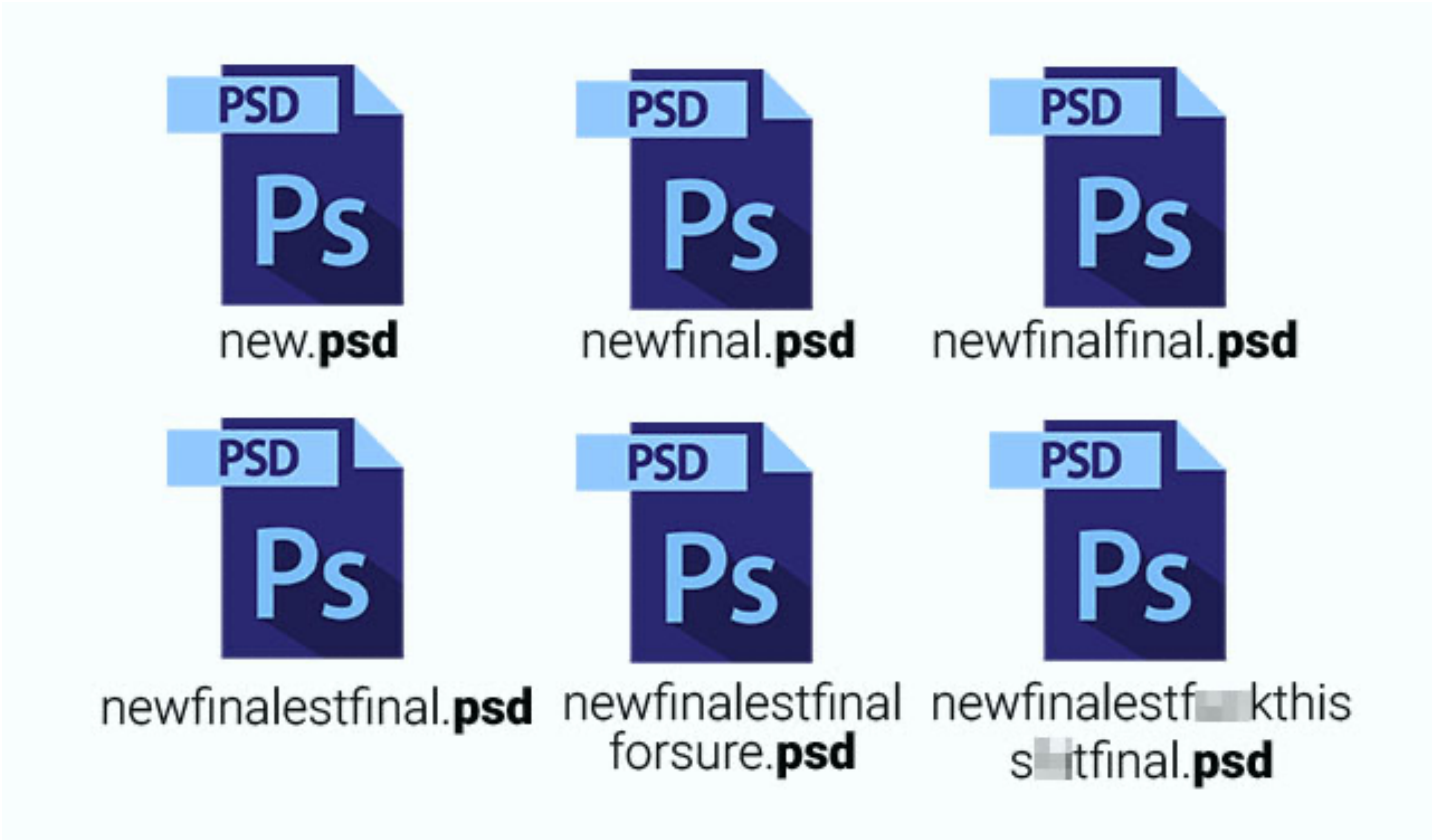
- Git: A system for tracking versions of files (Version Control)
- GitHub: Store your code + it tracks changes using git

Why do we need Git?

CENTER FOR Brains Minds+ Machines

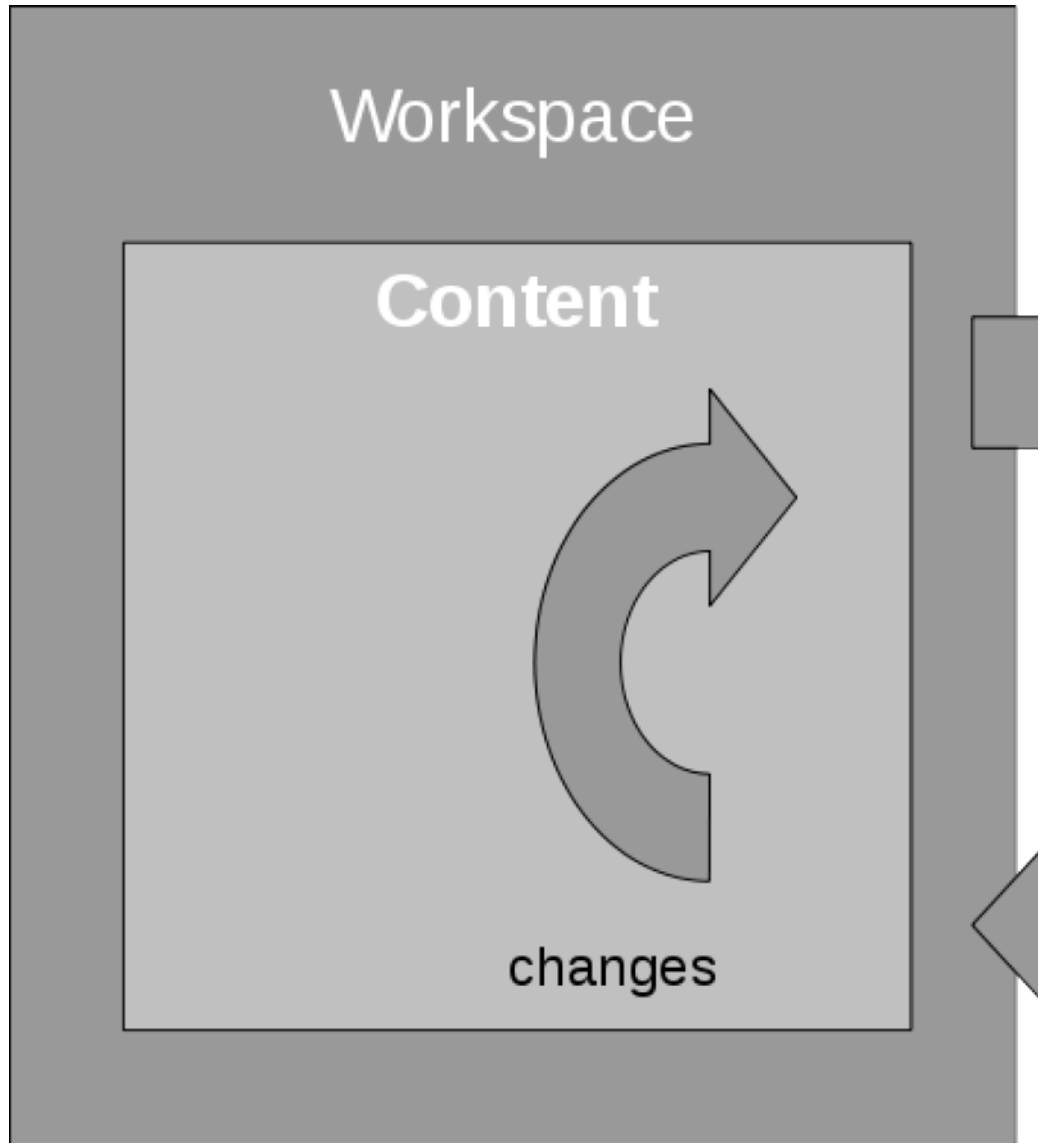


Why do we need Git?

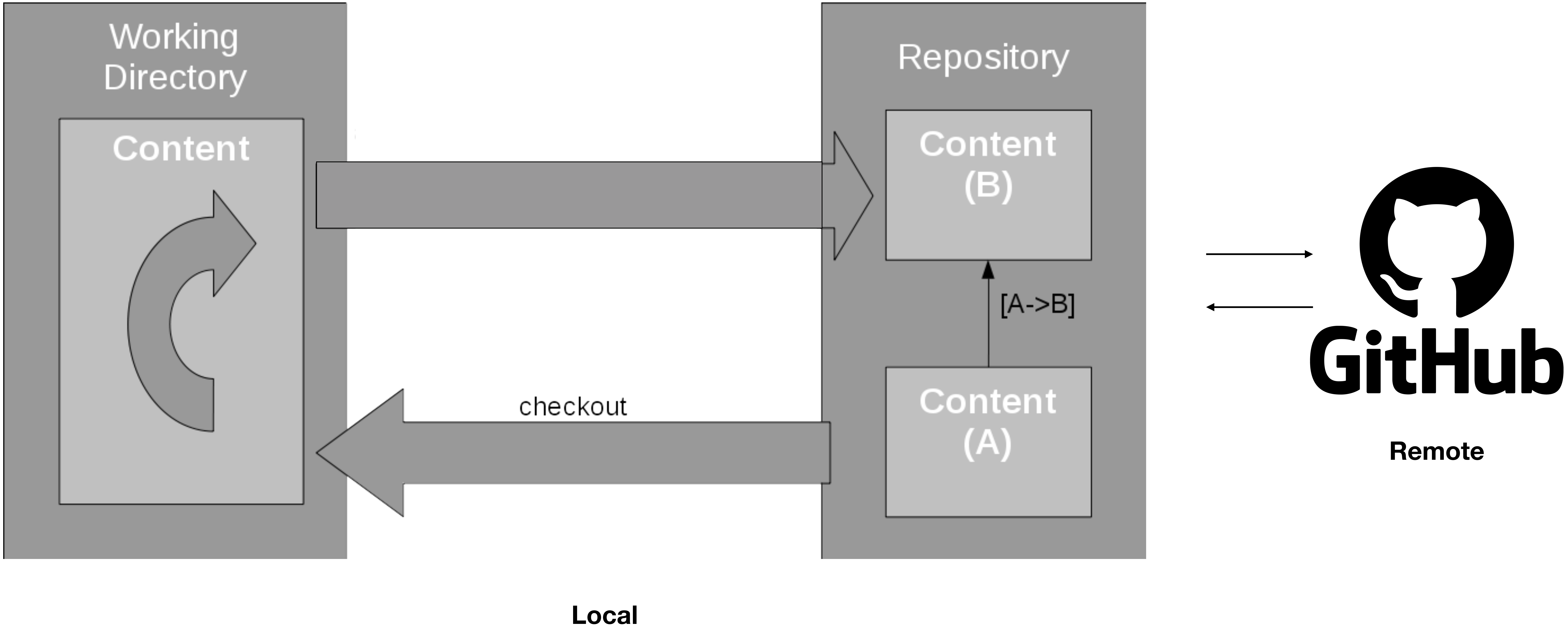


- Managing versions is important.
- Especially when multiple people are working on the same code-base.
- Combined with GitHub - Great way to store/manage/share your projects.

How version control works



How git works



Few useful commands

<u>Group 1: Getting stuff from github</u>
git clone - Copy from GitHub to local.
git fetch - get changes from remote
git merge - merge changes into local
git pull = git fetch + git merge

<u>Group 2: Reflecting your local changes on github</u>
git add - add a changed file to staging area
git commit - commit the change i.e. create snapshot
git push - upload new snapshot to github

<u>Group 3: Branches etc</u>
git checkout - switch or create new branch (Assignment 2)

More advanced git

CENTER FOR Brains Minds+ Machines

- How to resolve merge conflicts.
- Pushing too large a file - undoing added files.
- Undoing a commit i.e. moving to previous commit.

Summary Part 1

- Git = Management system for your files
- GitHub = a place where code is stored and managed using git snapshots

My Amazing ML Project

Files in that project

```
import torch
from torchvision import datasets

def train_model():

def evaluate_model():
```

Code in these files

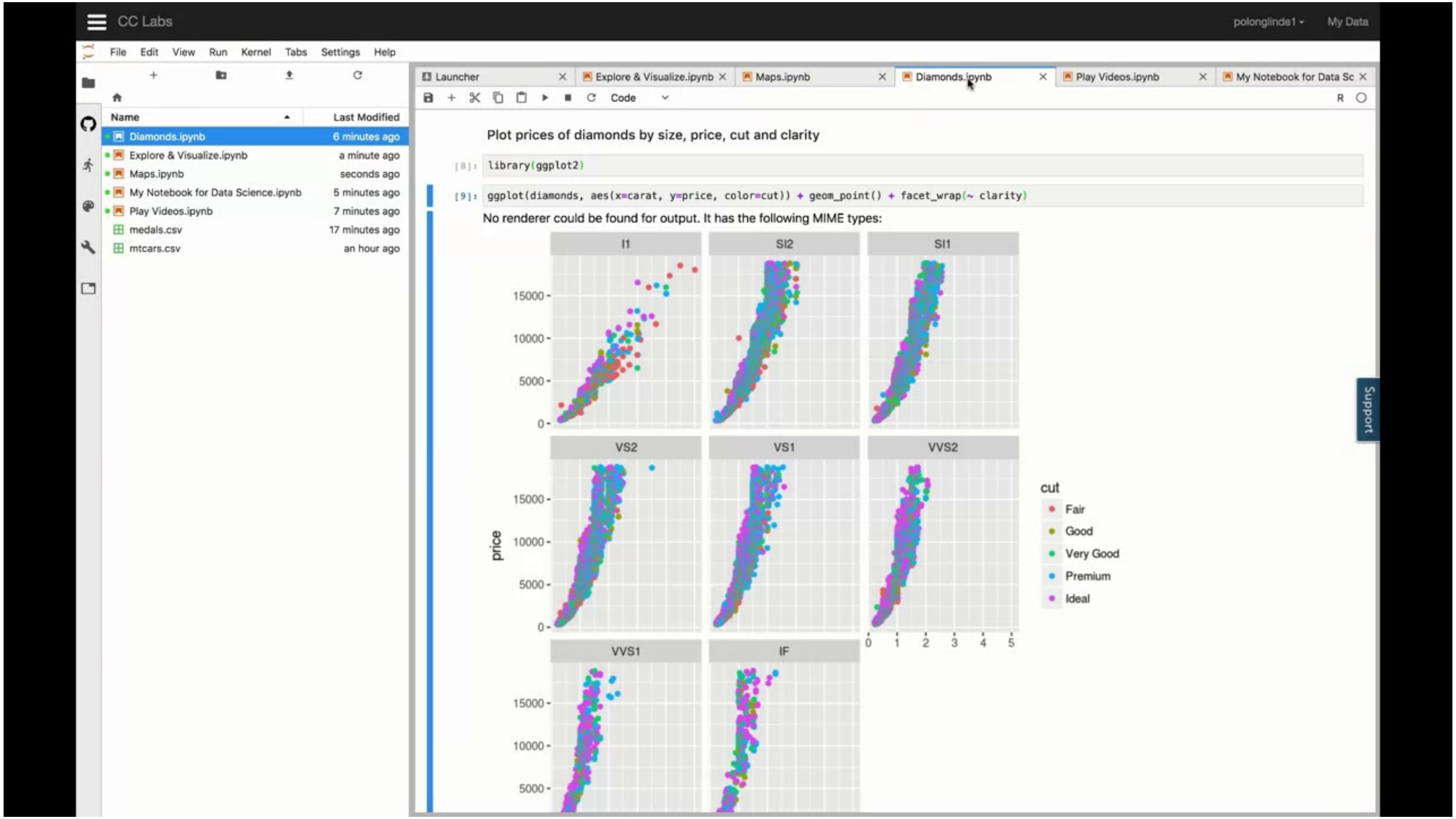
Part 2: Files in the Box

Files in that project

- **What kind of files would I have?**
- **How would these be opened?**
- **Which ones need to be shared and how?**

Jupyter

What is Jupyter?



Google Colab

Summary of Part 1 and 2

GitHub

Jupyter Notebooks

Part 3

```
import torch
from torchvision import datasets

def train_model():

def evaluate_model():
```

- What language/framework should we use to write code in our files?
- We will use PyTorch.

PyTorch

- Facebook's framework for ML/DL research.
- Easy to install.
- Easy to prototype.
- Resource for PyTorch: https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html

A very brief introduction to ML/DL

...and how PyTorch allows us to code for ML

What is Machine Learning?

- Very elusive to define.

- Tom Mitchell (CMU): “A machine for which <u>*performance improves*</u> for some <u>*specific task*</u> with <u>*experience*</u>”.


the world of
TOTAL

All About The Company
Global Activities
Corporate Structure
TOTAL's Story
Upstream Strategy
Downstream Strategy
Chemicals Strategy
TOTAL Foundation
Homepage

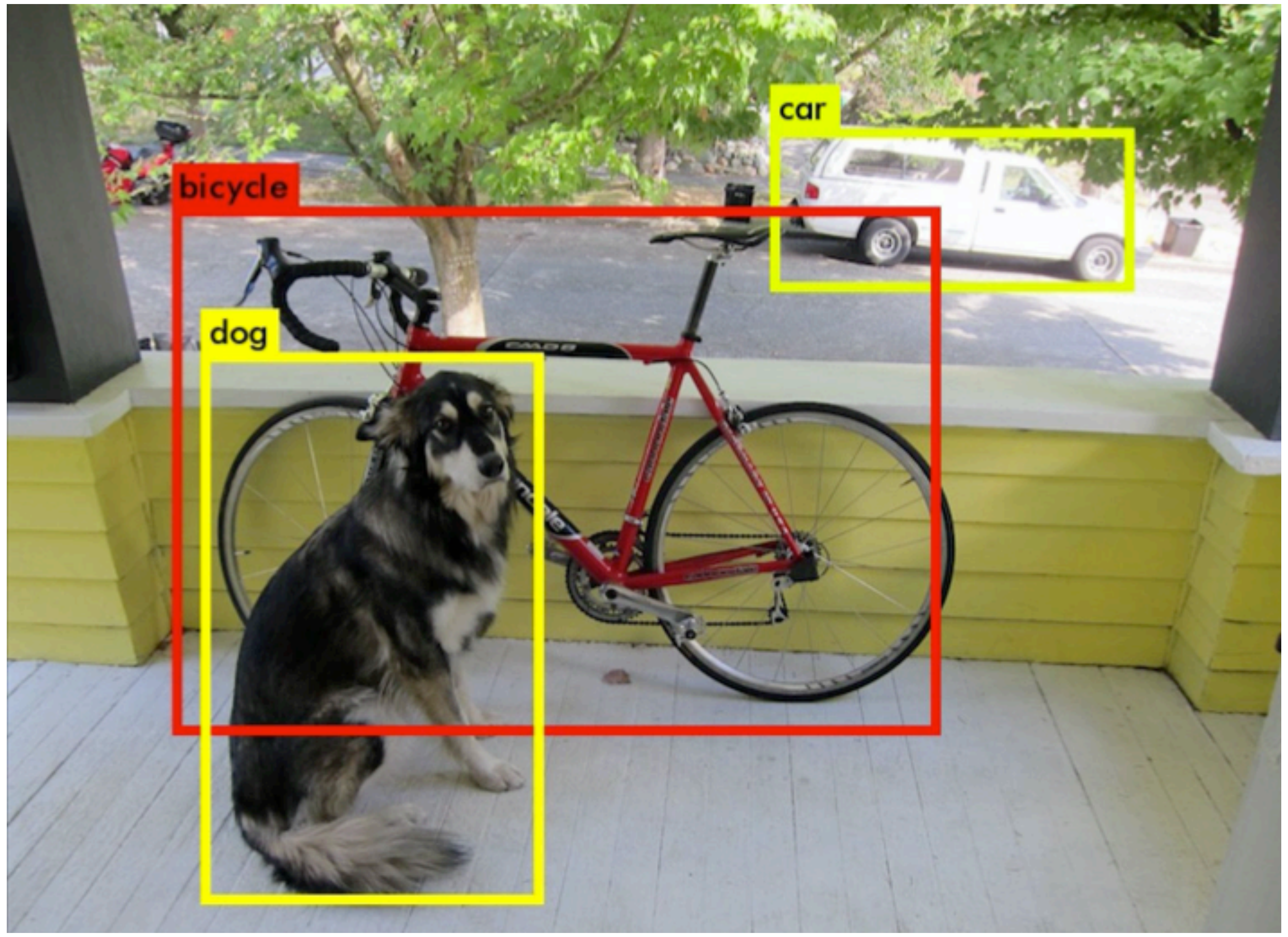
all about the
company

Our energy exploration, production, and distribution operations span the globe, with activities in more than 100 countries.

At TOTAL, we draw our greatest strength from our fast-growing oil and gas reserves. Our strategic emphasis on natural gas provides a strong position in a rapidly expanding market.

Our expanding refining and marketing operations in Asia and the Mediterranean Rim complement already solid positions in Europe, Africa, and the U.S.

Our growing specialty chemicals sector adds balance and profit to the core energy business.



Why is ML everywhere?

Machine Learning

Unprocessed things

The toolkit perspective

CENTER FOR
Brains
Minds+
Machines

INFORMATION GOES IN → MACHINE LEARNING MODEL → MORE USEFUL INFORMATION COMES OUT

x_i f y_i

CENTER FOR Brains Minds+ Machines

EXAMPLES

→ Machine Learning System → Chair YES/NO

What is f?

- Assumption: Underlying relationship. We have reason to believe that:

$$y_i = g(x_i) \quad \forall x_i \in X$$

- But, g is very, very complex/hard to pin point. Ex: protein folding, music preference, voting preference, and so on.

- Solution: Let’s approximate g with f.

Typically learning setups

- **Supervised:** Both y (labels) and x (inputs) are given.
    - Ex: Is this a picture of chair? Can you translate this from English to Hindi/Mandarin/French?
- **Unsupervised:** Discovering patterns in x, independent of a y.
    - Ex: Anomaly detection in a subset of pictures, or predicting next word in a sequence.

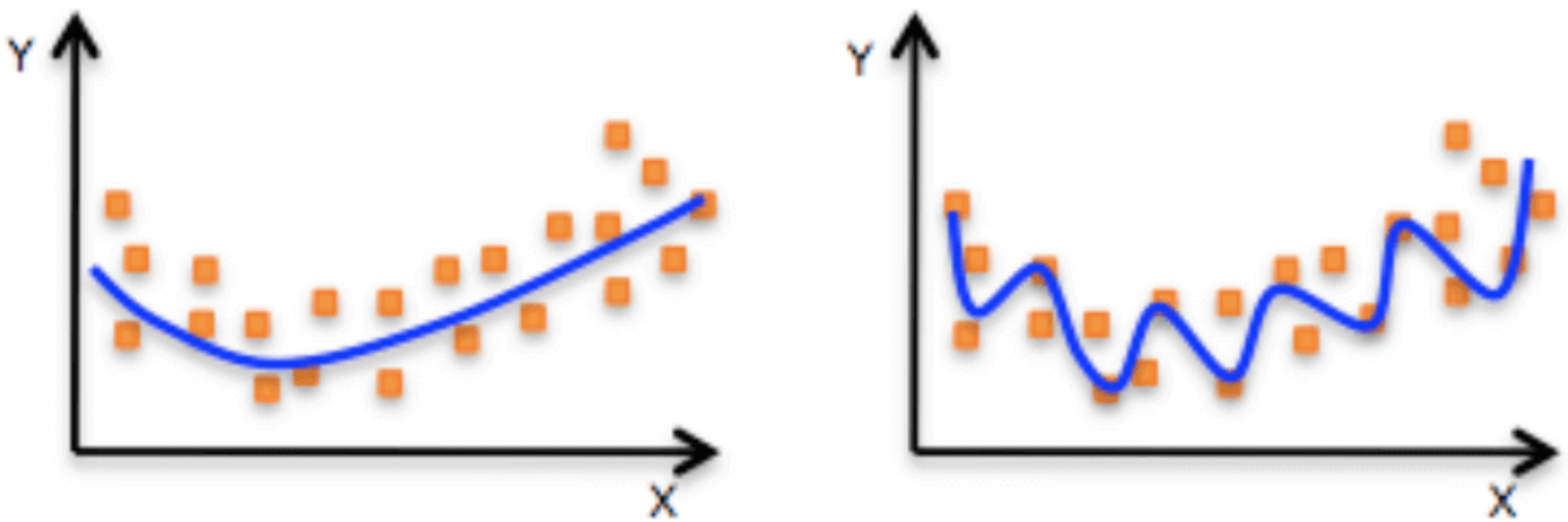
Typically learning setups

- **Semi-supervised:** Supervised + Unsupervised components.
  - Ex: MRI, CT scans - mixing vision + small, labeled doctor feedback.
- **Reinforcement:** Really shouldn't be on the same list. Specifically cast for settings where multiple actions are taken in a sequence.

Supervised Learning

$$y_i = g(x_i) \sim f(x_i)$$

- Goal, finding f which best approximates g.
- Let $x_i \in X_{train}, y_i \in Y_{train}$, is the goal to find the best fit f on (X_{train}, Y_{train})?



Overfitting

Empirical Risk Minimization

- f and g should be similar on all possible X,Y you will get.

- This includes - (1) Train set, (2) Test set, **(3) Samples you don't have**

$$\epsilon(f) = \int_{X \times Y} f(x) \sim g(x)$$

- How should we quantify f ~ g?

$$L(f(x), y) \rightarrow [0, \inf)$$

Empirical Risk Minimization II

$$\epsilon(f) = \int_{X \times Y} f(x) \sim g(x)$$

$$\epsilon(f) = \int_{X \times Y} L(y, f(x))$$

$$L(f(x), y) \rightarrow [0, \inf)$$

The goal of supervised learning: $$\min_{f} \epsilon(f)$$

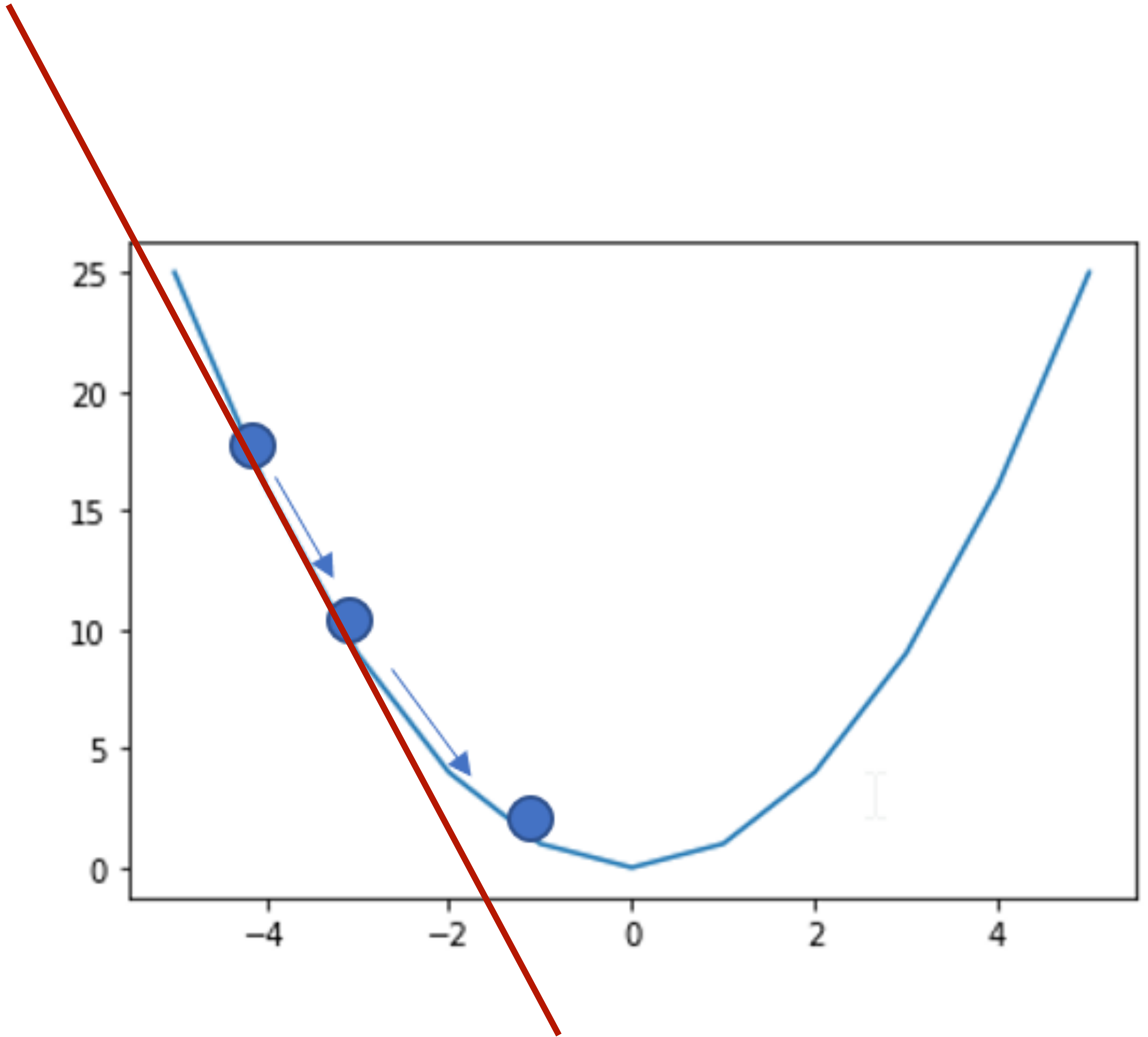
What all factors did we see?

- We are trying to minimize, the loss function (L) incurred by our chosen function f on dataset space (X,Y)
- Let's go over each one of these components

Minimization: How do we do it?

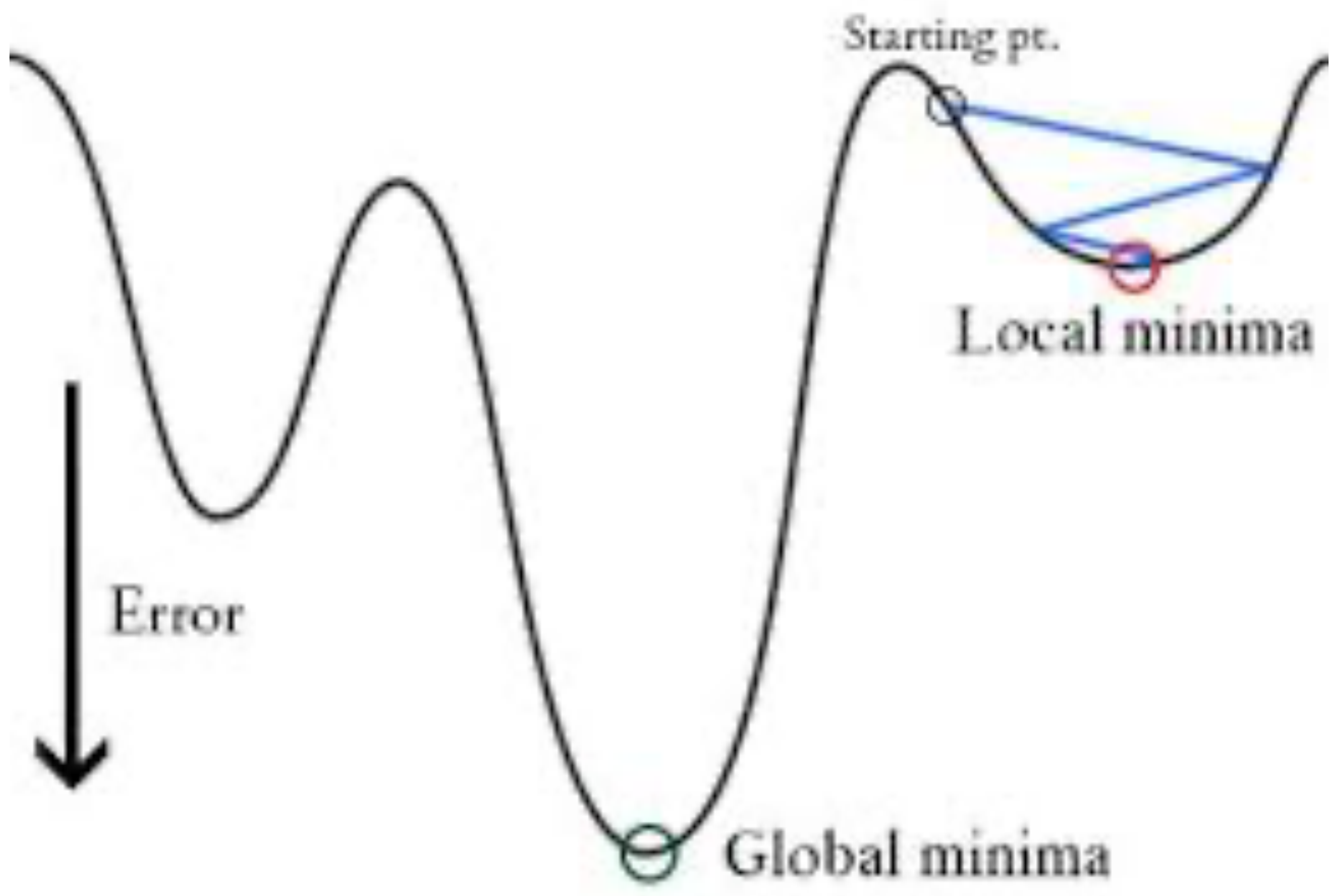
- Gradient Descent: (1) Measure gradient of loss w.r.t. parameter (say α),

  (2) change parameter as: $\alpha = \alpha + r * \dfrac{\partial L}{\partial \alpha}$



How about a more complex function?

- Gradient descent should not work.
- We have other variants which work better in practice.



So, we need to make the job easy.

- Components to play with - Loss function L, Data (X,Y), and architecture of the function f.
- Let’s see how we can modify each of them.

What is a good loss function?

- It should reflect what is important to us. For classification:- don’t penalize if prediction is same, and do penalize if classification is wrong.

- Should help optimization. Best to keep it continuous.

- For ex: Misclassification loss does not have good gradients. Hence, we use CrossEntropy loss for classification.

- If we have a custom use case, we can design a custom loss function.

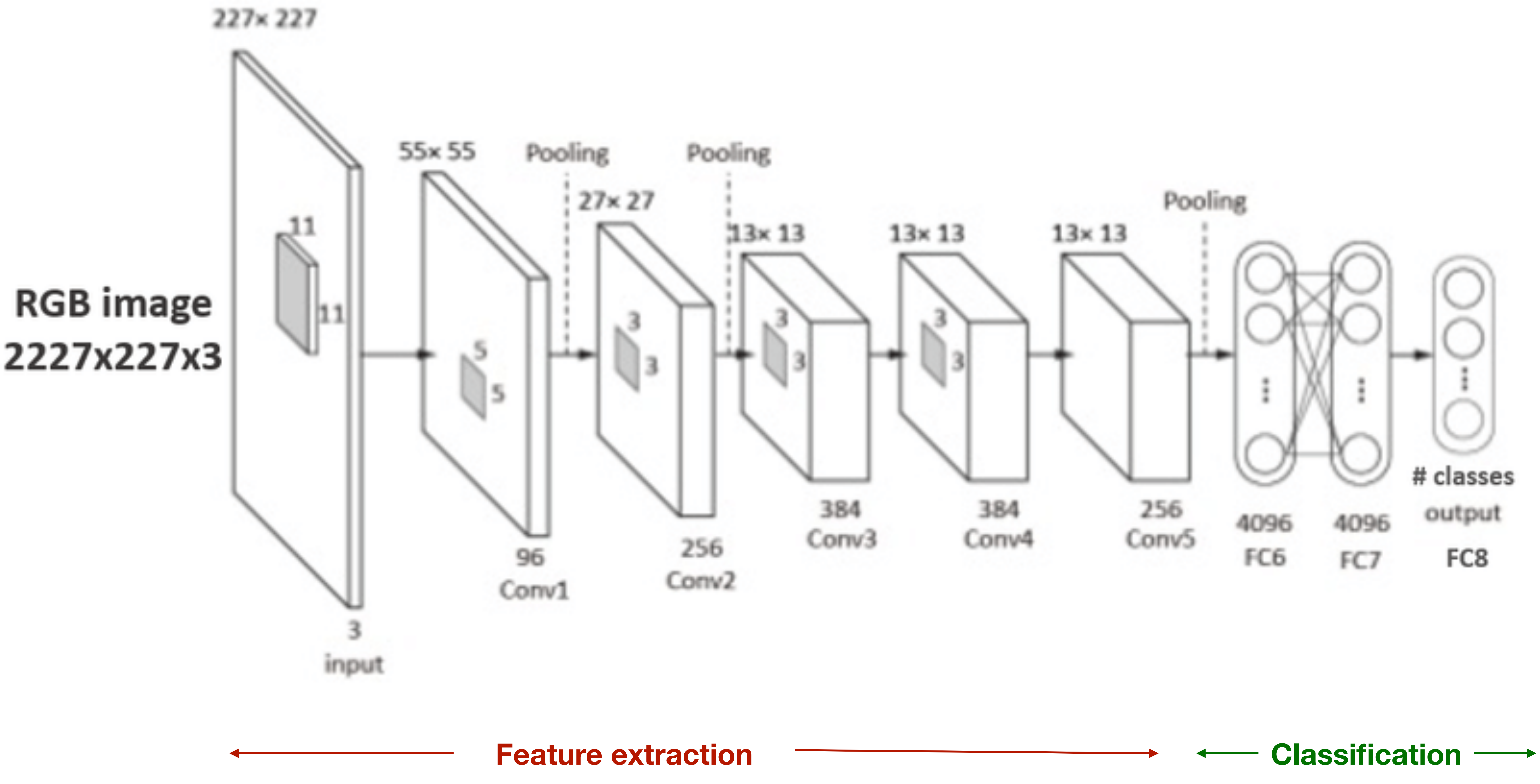
Making (X,Y) most useful

- Features = facets of your data. Ex: For picking best advertisement - age, shoe preference, favorite color.
- Conventional ML: features are everything.
- Finding the right features is very, very hard.

Making (X,Y) most useful

- Typical things done:- PCA, mean normalization. More fancy ones - whitening, sketching/streaming, random projections.
- Deep learning automates this search. First few layers = non-linear projection of your data.
- First few layers combine parts of input in non-linear way till they find something which is a good feature.

Deep Learning - automatic feature extraction



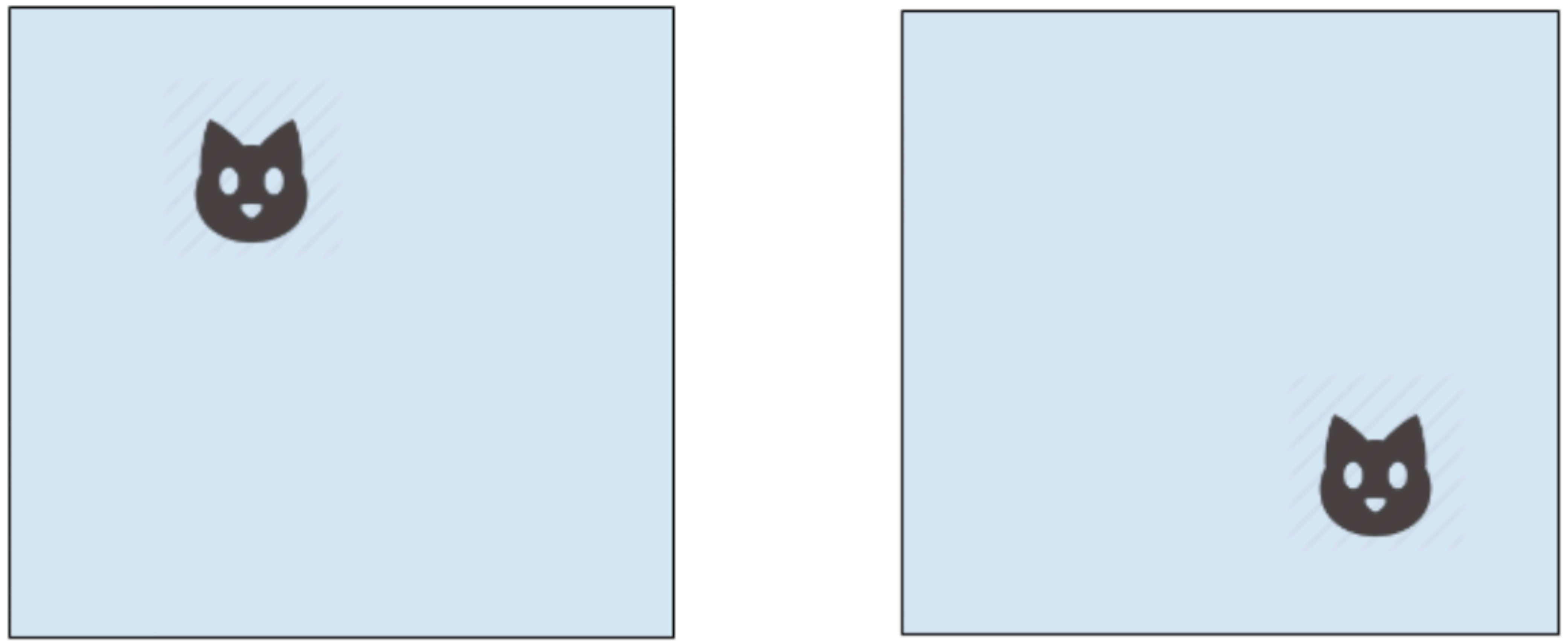
Important questions worth asking

- Are we sure a CNN can be the mathematically best f?
    - Yes, universal approximation theorem proves that.
- If we’re doing automatic search, why do certain architectures work better? Why CNNs, and not normal feed forward architectures?

Architectures to help automatic search

- Addition of inductive bias - i.e. exploiting the structure of the problem.

- For ex:



- CNNs, RNNs are designed so this structure can be exploited.

I am going to the _____ because I am hungry.

I am hungry, so I am going to the _____.

So, how do we code all this?

We were introduced to 4 components of Deep Learning today. These are:

1. Dataset: We want to load + Pre-Processing easily + fast.

2. Architectures: We may want to include specific structure of our problem.

3. Loss functions: Custom loss functions for our task are important.

4. Optimization: Variant of gradient descent used. So, gradient of loss w.r.t. NN parameters needs to be calculated.

PyTorch returns

PyTorch allows great control over each of these aspects

- Data-Loading: Automatically manages issues of memory and speed on GPU.
- Architectural expressivity: If you’ve ever used CAFFE, you would know.

libstdc++ installation

This route is not for the faint of heart. For OS X 10.10 and 10.9 you should install CUDA 7 and follow the instructions above. If that is not an option, take a deep breath and carry on.

- Easy definition of losses and custom losses.
- Autograd.
- Easy implementation of different gradient algorithms.

CENTER FOR
Brains
Minds+
Machines

Let’s see it in code?

https://colab.research.google.com/drive/125blGZwFZUm8g6YsRntzyJrrp-NjWkuW?authuser=1