Bangkok | Manila | Singapore

GeoML 101

Introductory concepts for Machine Learning in Python

January 11, 2023

About Our Instructors

Joshua Cortez
Machine Learning Consultant

- Developed and productionalized geospatial machine learning models for social development and sustainability applications
- Built geospatial data pipelines for sustainability and telecommunication organizations
- Worked on customer analytics for banks

GeoML 101

Day 2 Overview

Instructor: Joshua Cortez

1. Introduction to Classical Machine Learning
   a. Overview of ML
   b. Data Exploration
   c. Model Development
2. Introduction to Computer Vision
   a. Computer vision use cases
   b. Types of computer vision tasks
   c. Land Cover Land Use Classification

Workshop Objectives

By the end of the workshop, participants should be able to:

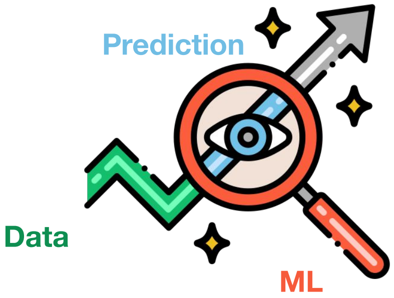
- Define ML, example use cases, and explain the steps in an ML workflow
- Understand the relevance of computer vision and its geospatial applications

Classical Machine Learning

What is Machine Learning?

What is it?

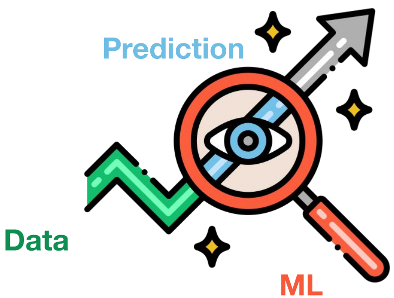
A tool for making inferences and predictions from data



What is Machine Learning?

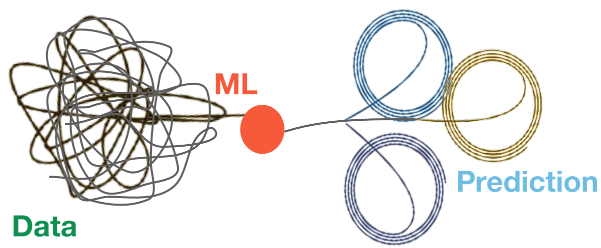
What is it?

A tool for making inferences and predictions from data



What can it do?

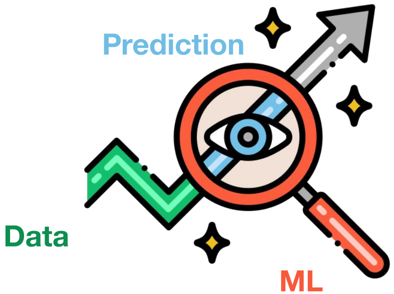
It can predict outcomes from the data based on patterns humans may not see



What is Machine Learning?

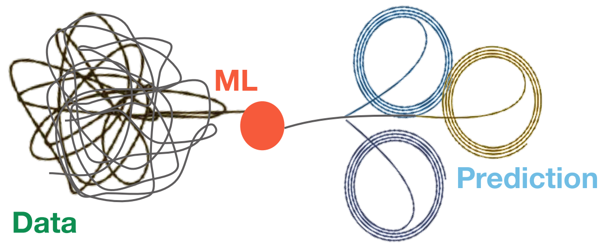
What is it?

A tool for making inferences and predictions from data



What can it do?

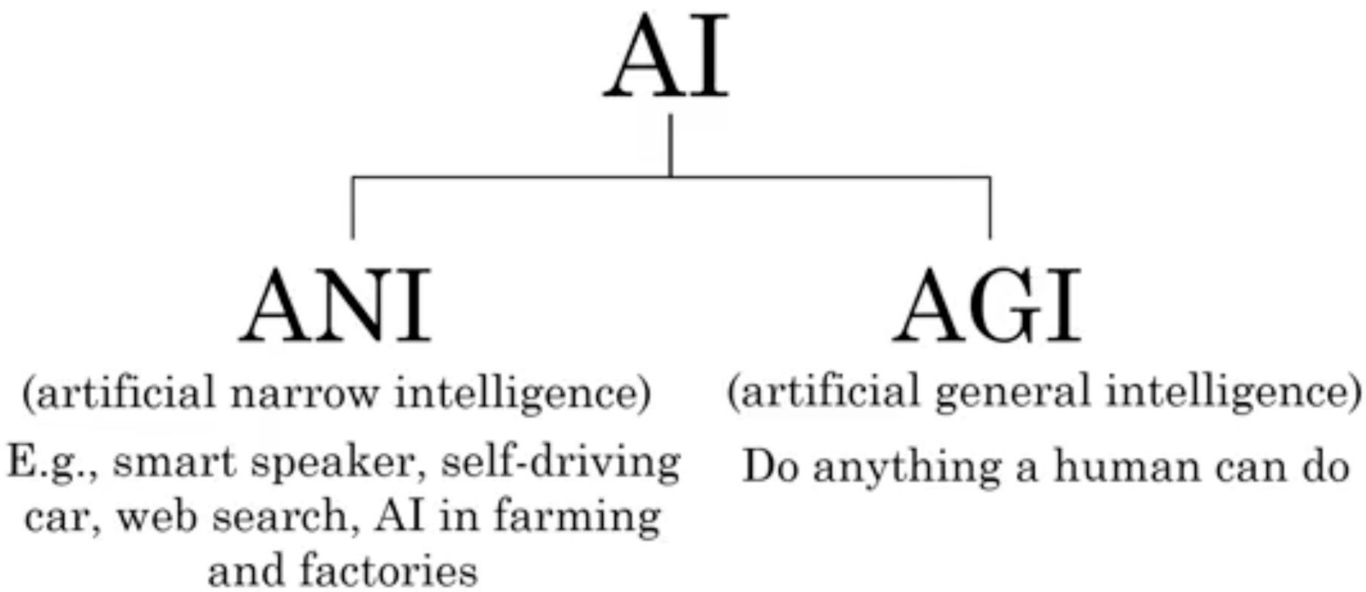
It can predict outcomes from the data based on patterns humans may not see



How does it work?

Using methods from statistics and computer science, the machine learns the patterns from existing data and applies it to new data

What’s the difference between AI and ML?

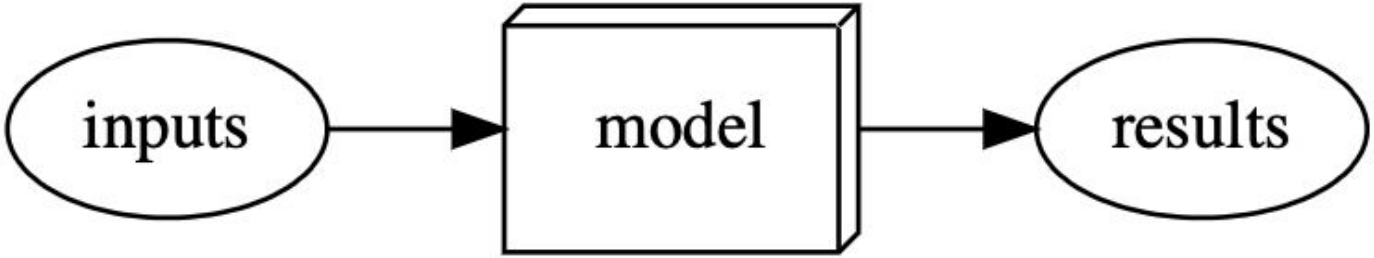


Lots of progress and applications here
Generally called as Machine Learning

Computers learn specific tasks by training on data

An active area of research but no real AGI yet

In ML, you typically use give input data to a trained model to generate a result or prediction



Application	Input	Results/Output
House Price Prediction	Size of house, # of bedrooms, etc	Price of the house
Spam Filtering	Email	Classification spam or no spam?
Machine Translation	English text	Filipino text

Machine Learning

A field of study that gives computers the ability to **learn from examples** without being explicitly programmed.

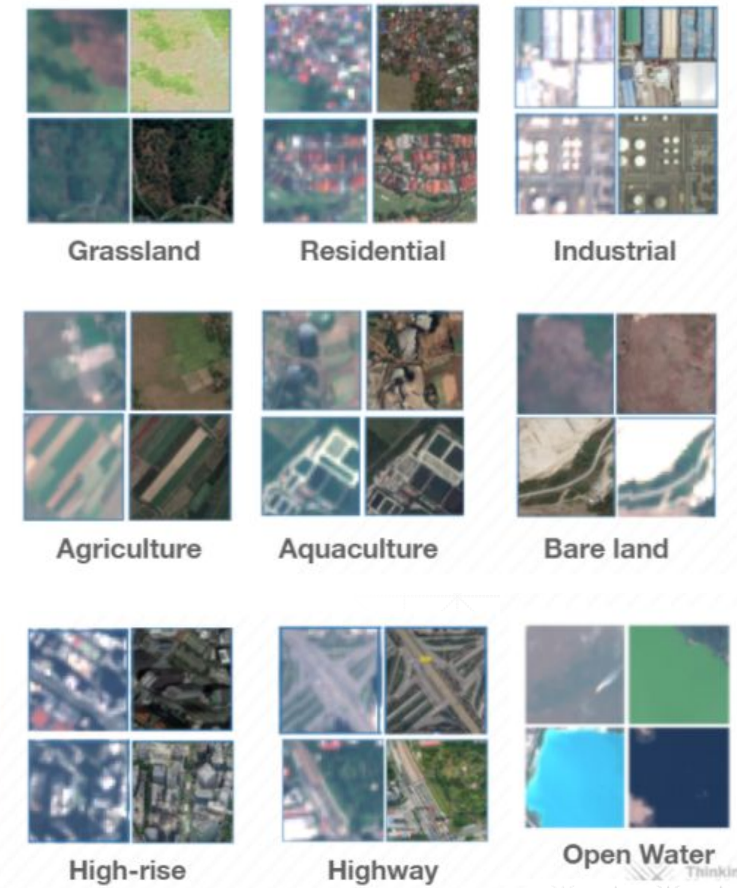
Example: Classifying land use and land cover

```
def detect_color():
    #lots of code here

def d          ):
    #l        de here

def a         re():
    #1     of     e here

def analyze_texture():
    #lots of code here
```



Machine Learning

We often need a lot of labeled data to train a complex machine learning model.

ML Model

Land Use & Land Cover Classification

Residential, Industrial, Agricultural, Forest, Bare Land, Open Water, etc.

Typical ML Project Development Flow

Data Exploration				Model Development		
Problem Definition	**Data Wrangling**	**EDA**	**Model Design**	**Feature Engineering**	**Experiment-ation and Tuning**	**Model Evaluation**
Identifying the dependent variable and problem type	Getting data Sanity checking	Extracting initial insights Identifying possible predictors Checking difficulty of problem	Model types Table structure Splitting data	Data pre-processing Data transformation Feature selection	Tuning hyperparameters Feature selection	Quality of results Stability of results Deciding on “best” model

In reality, there are lots of iterations in ML

**Problem is too hard?
Refine the definition!**

Data Exploration

Model Development

Problem Definition	Data Wrangling	EDA	Model Design	Feature Engineering	Experiment-ation and Tuning	Model Evaluation
Identifying the dependent variable and problem type	Getting data Sanity checking	Extracting initial insights Identifying possible predictors Checking difficulty of problem	Model types Table structure Splitting data	Data pre-processing Data transformation Feature selection	Tuning hyperparameters Feature selection	Quality of results Stability of results Deciding on “best” model

In reality, there are lots of iterations in ML

If you need more data...

Data Exploration				Model Development		
Problem Definition	**Data Wrangling**	**EDA**	**Model Design**	**Feature Engineering**	**Experiment-ation and Tuning**	**Model Evaluation**
Identifying the dependent variable and problem type	Getting data Sanity checking	Extracting initial insights Identifying possible predictors Checking difficulty of problem	Model types Table structure Splitting data	Data pre-processing Data transformation Feature selection	Tuning hyperparameters Feature selection	Quality of results Stability of results Deciding on "best" model

In reality, there are lots of iterations in ML

The model is not doing well? Try other model types and hyperparameters!

Data Exploration | **Model Development**

Problem Definition	Data Wrangling	EDA	Model Design	Feature Engineering	Experiment-ation and Tuning	Model Evaluation
Identifying the dependent variable and problem type	Getting data Sanity checking	Extracting initial insights Identifying possible predictors Checking difficulty of problem	Model types Table structure Splitting data	Data pre-processing Data transformation Feature selection	Tuning hyperparameters Feature selection	Quality of results Stability of results Deciding on "best" model

In reality, there are lots of iterations in ML

The model is still not doing well? Do more feature engineering!

Data Exploration

Problem Definition
- Identifying the dependent variable and problem type

Data Wrangling
- Getting data
- Sanity checking

EDA
- Extracting initial insights
- Identifying possible predictors
- Checking difficulty of problem

Model Design
- Model types
- Table structure
- Splitting data

Model Development

Feature Engineering
- Data pre-processing
- Data transformation
- Feature selection

Experiment-ation and Tuning
- Tuning hyperparameters
- Feature selection

Model Evaluation
- Quality of results
- Stability of results
- Deciding on "best" model

Problem Definition

- Clearly define the problem / use case
- Identify the problem type

Poverty Estimation

Can we locate the most vulnerable communities?

Photo source: Chitto Cancio via Unsplash

Poverty estimation in the Philippines

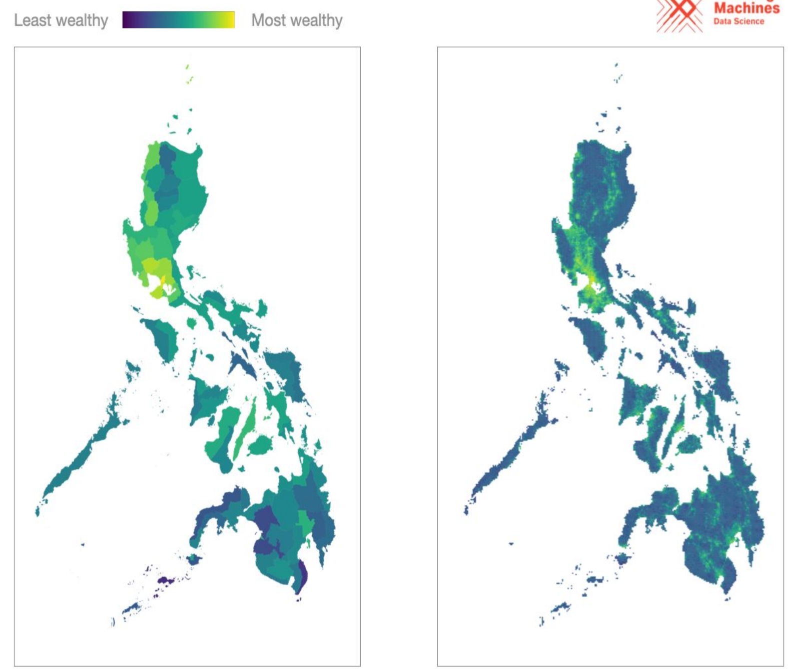
Challenge

Without access to high resolution data, organizations risk investing limited resources in the wrong priorities

Opportunity

Use ML and open data to produce:

- Interpretable ML model trained on the Demographic Health Survey that generates detailed poverty estimates across the country
- Poverty map can help develop data-driven social policies and programs

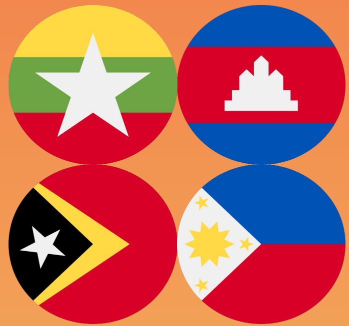


Actual Wealth

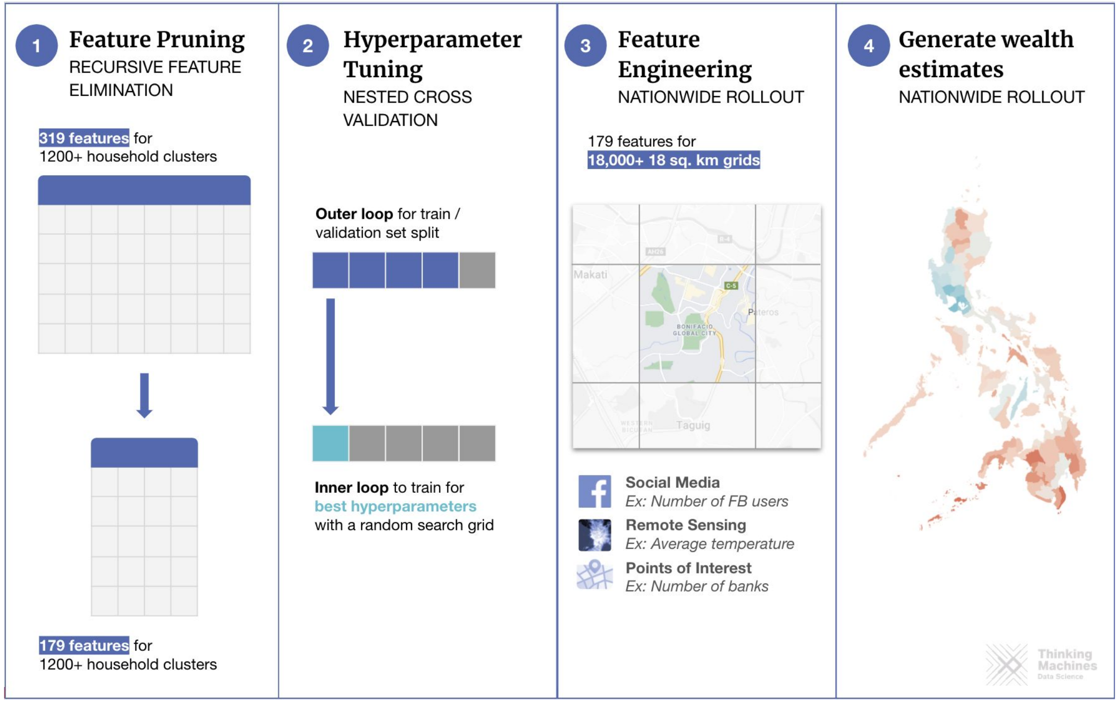
ML-Generated Wealth Estimates

Key open data and tools:

From our previous poverty mapping for the Philippines in 2022, we're working on developing the models across Southeast Asia



We leveraged openly available datasets and DHS wealth data to map poverty across the country



Source: Thinking Machines website

Hands-on Exercise

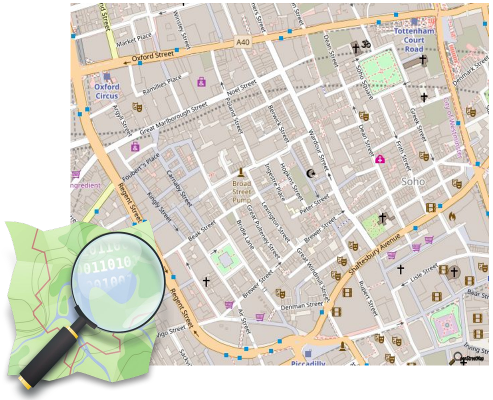
Exercise 4 to 6

Please make your own copy by clicking File > Save a Copy in Drive

Data Wrangling

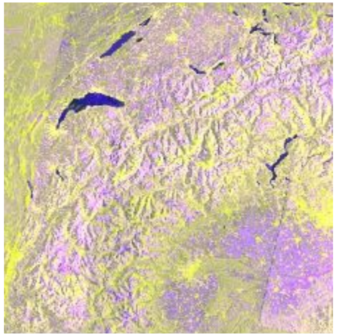
- Obtain raw data and do some pre-processing
  - Some early data transformations (e.g., aggregating to the appropriate level)
  - Combining data across different sources
- Get an overall “feel” of the data
- Sanity checks: Does the data make sense?
- Data completeness checks: Are there a lot of NULLS?

Examples of geospatial data



Open Street Map (OSM)

- The "Wikipedia" of maps. Has data on roads and buildings by type
- A global community of volunteers update and review this data



Satellite Imagery

- Different kinds of images captured by satellites orbiting the globe.
- Some satellite imagery are open, such as Sentinel-2 from the European Space Agency

What do the table/s and fields represent? [DHS]

- The data comes from a DHS 2014 on-the-ground survey conducted in Cambodia regarding the households' socio-demographic information
- Each row is DHS cluster
- Each column is some information about that DHS cluster
- Wealth Index: derived feature (using PCA) where higher values mean more wealthy areas
- Coordinates (lon, lat): are jittered to preserve privacy of survey respondents

DHSCLUST	DHSID	Wealth Index	Longitude	Latitude
1	KH201400001	-2355	103.5409	13.3958
2	KH201400002	6789	102.0353	13.9542

What do the table/s and fields represent? [OSM]

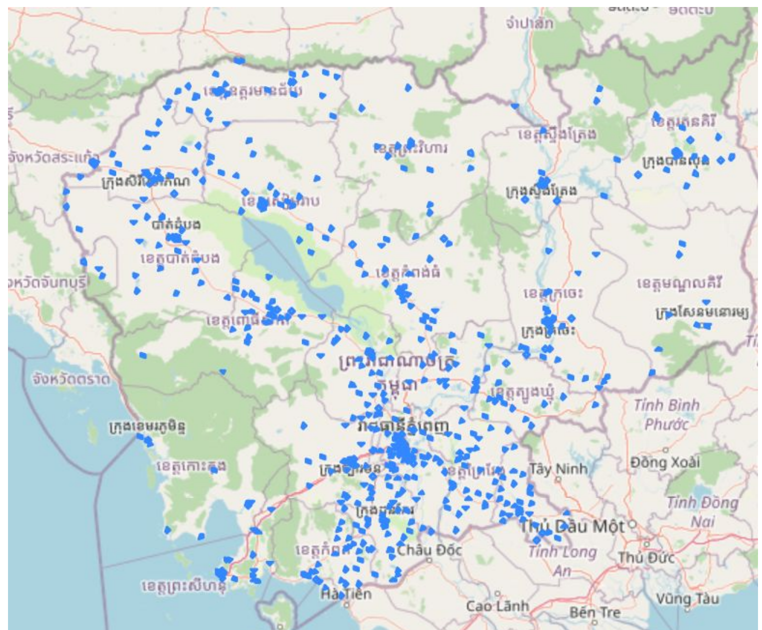
- The data comes open street map, prepacked from the Geofabrik website
- Each row is a point of interest (POI) and each column describes the POI
- The fclass column describes the type of POI (here they are supermarkets)
- The geometry is concise and machine readable way of storing the geometric information. Here we have POINT, but there are also POLYGON, and LINESTRING geometries

osm_id	code	fclass	name	geometry
182561	2501	supermarket	AEON Madxvalu Express Takhmao	POINT(104.94791, 11.47526)
182562	2501	supermarket	AEON Maxvalu Express Bodaiju	POINT(104.84801, 11.55945)

Data Checking

- Do the values make sense? Do they fall within expected ranges?
- Are there lots of missing values?
- Can you quickly visualize the data?

Visualized DHS Clusters

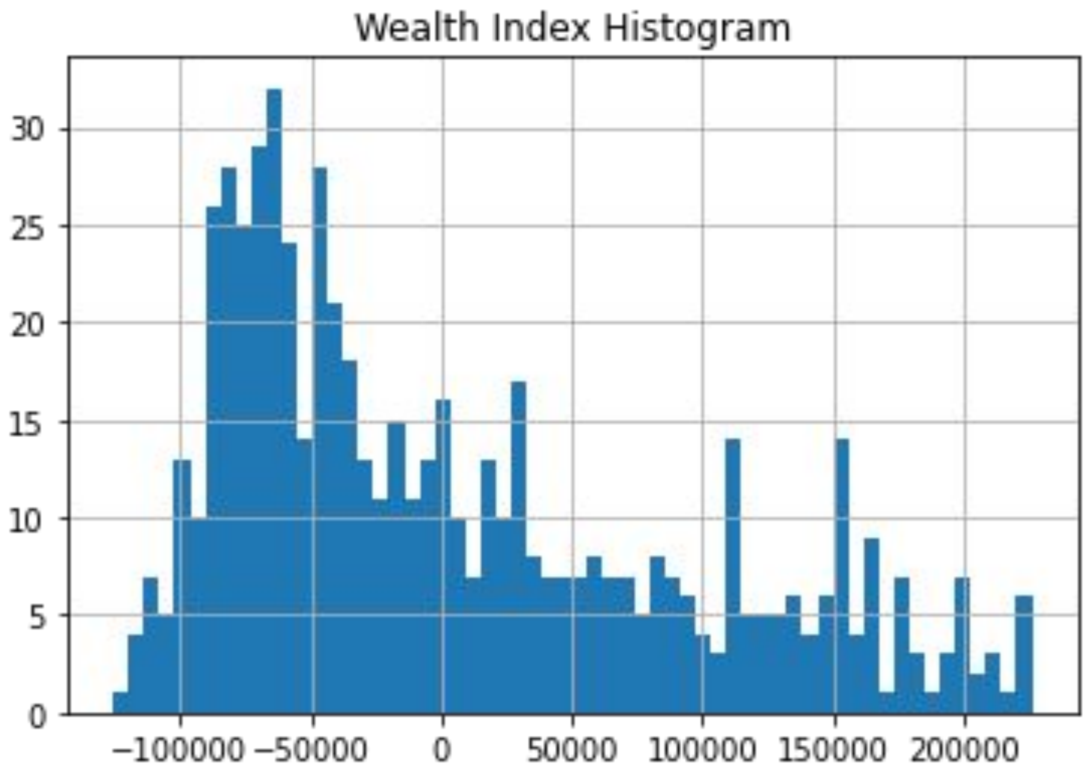


EDA (Exploratory Data Analysis)

- Get an even *better* overall feel for the data
- Look for relationships between variables
- Draw some early insights

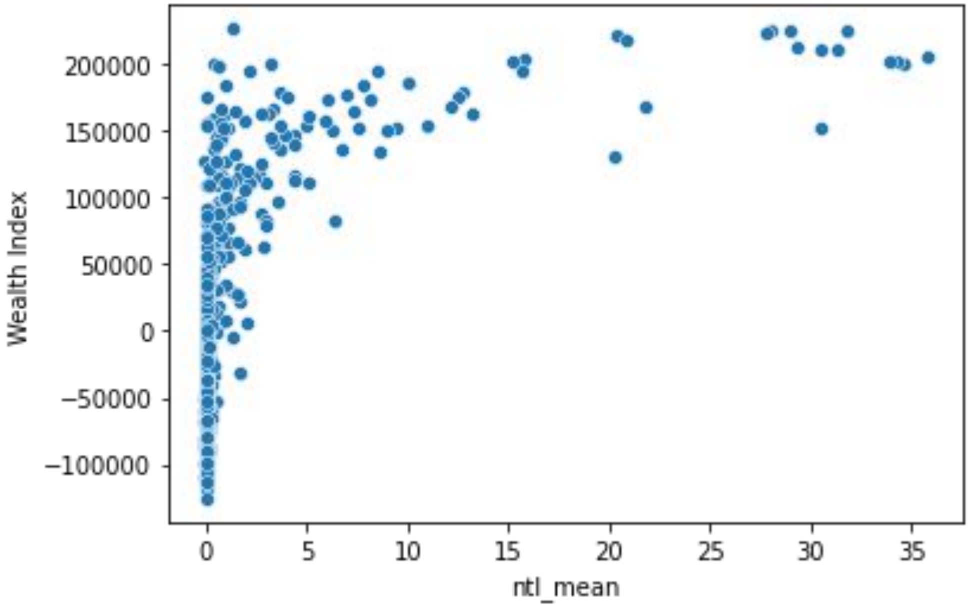
How is the target variable distributed?

- We can use the histogram plot to visualize the distribution



Is there signal in the data?

- Purpose is to check for relationships between variables
- Specifically, the relationship between your dependent variable (the thing you're trying to predict) and the independent variable **("features")**



Sample scatterplot between wealth index and average Nighttime Lights

Model Design

- Determine what the final table for model development looks like
  - Sometimes called the “ABT” or “Analytical Base Table”
- Decide on model types to test
- Split validation into train-validation-test sets
- Plan experimentation workflow

Table Structure

DHS Cluster Radius Info

Dependent Variable

Features

Each row is a 2km radius around each DHS cluster

DHSID	geometry	Wealth Index	OSM Features (POI count, distance to nearest POI, etc)	Ookla Features (Avg Upload Speed, Avg Download Speed, etc.)	Nighttime Light Features (min, max, etc)
KH201400000001	POLYGON(...)	-2355	...	...	...
KH201400000002	POLYGON(...)	6789	...	...	...
KH201400000003	POLYGON(...)	12325	...	...	...

Choosing Model Types

- Nature of the data
  - Dependent variables
  - Are there lots of features? What types of features?
- Check the literature: track record of handling particular types of problems well
- Use case for projects:
  - Performance
  - Speed
  - Interpretability
  - Ease of implementation in the end user's systems / environment

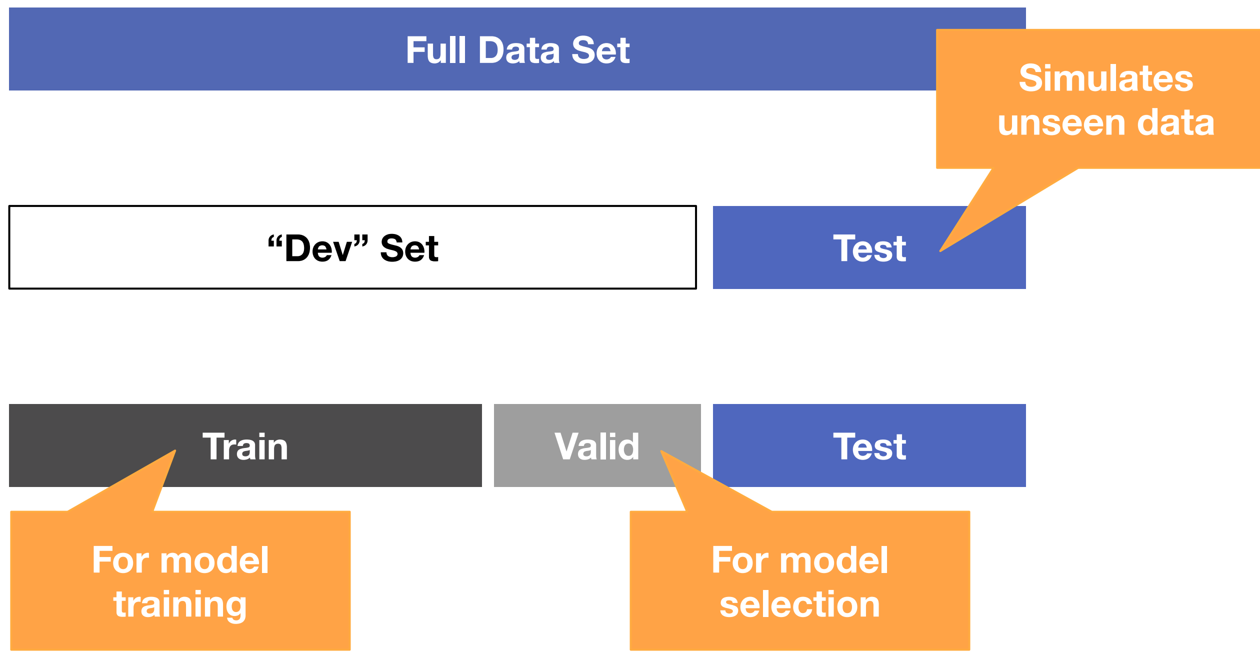
Factors	Logistic Regression (LR)	Random Forests (RF)	Light Gradient Boosting Machines (LGBM)
Number of estimators	One	Many	Many
Speed	Fast	Slow	Fast, typically used for larger data sets
Data pre-treatment	Treatment of NULLS, One-hot encoding, Standardization	Treatment of NULLS, One-hot encoding	Does not need one-hot encoding
Hyperparameters to consider	Few	Many	Many
Can handle non-linearity?	It's complicated	Yes	Yes
Explainability	High	Medium	Medium

Data Splitting

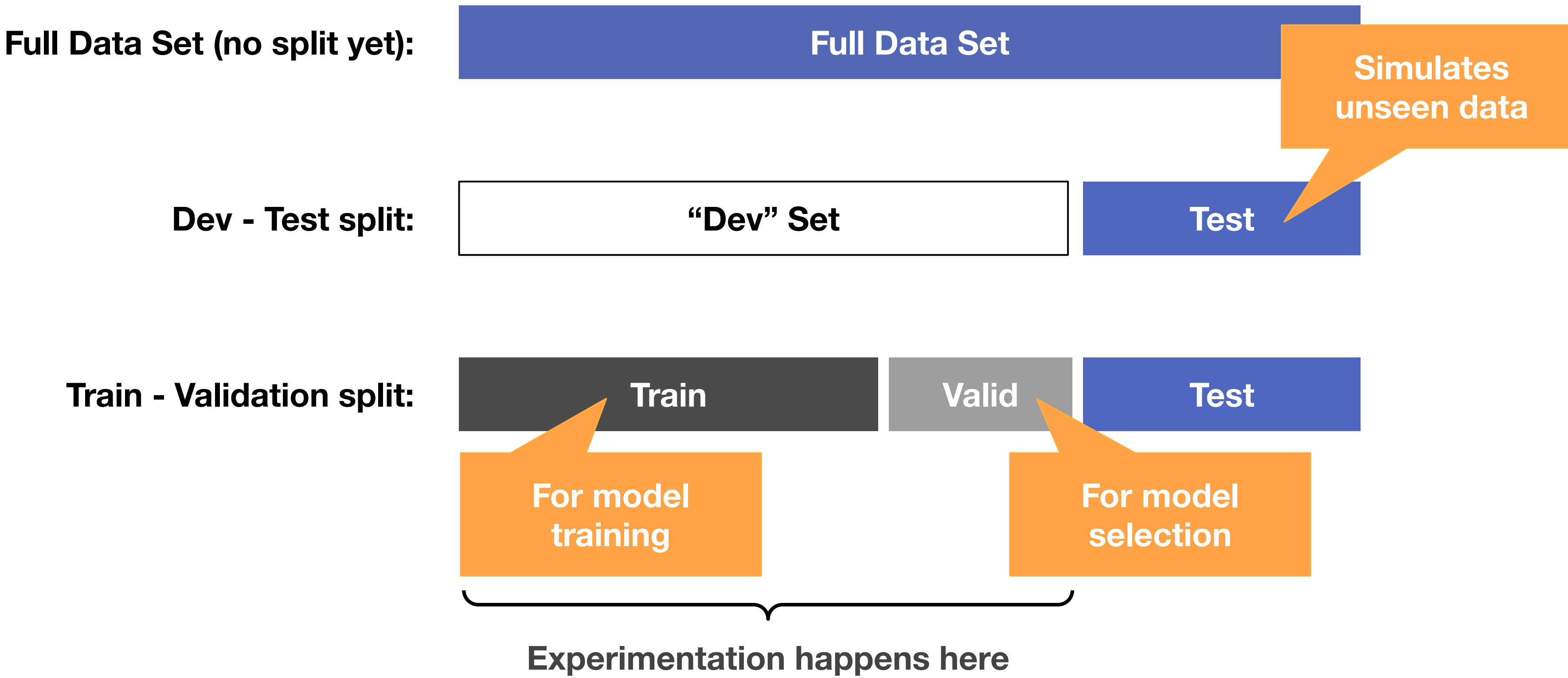
Full Data Set (no split yet):

Dev - Test split:

Train - Validation split:



Data Splitting



04

Model Development

04

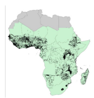
Model Development

1. Feature Engineering
2. Experimentation and Tuning
3. Model Evaluation

Feature Engineering

- Transform features into the values that will be used for experimentation
- Treatment of NULL values when needed
- Determine which features to include in experimentation

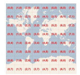
Feature Engineering Recap



DHS Clusters AOI Generation

Aggregate DHS data into cluster-level, and generate AOI geometries for each

Cluster Geometries →



Country Grid Tile Generation

Split up a country into equally sized grid tiles

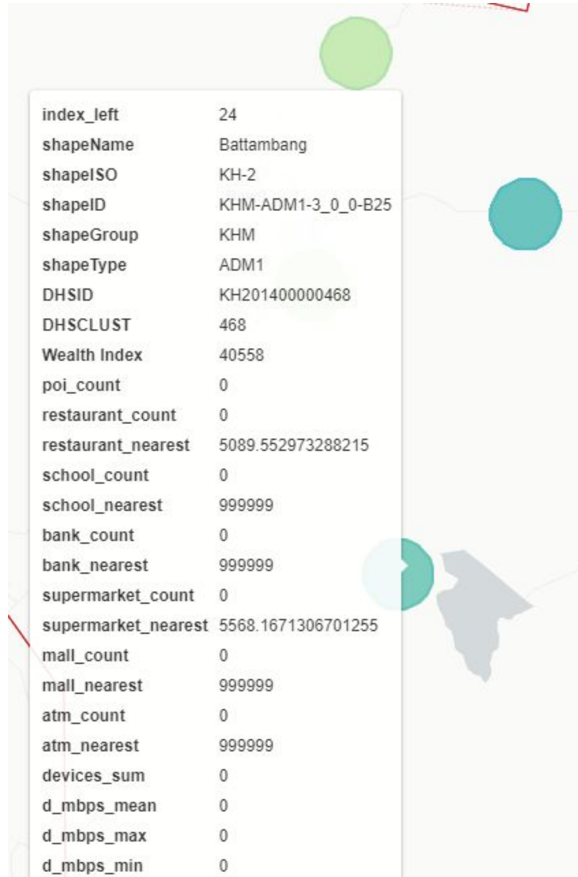
Tile Geometries →

Feature Engineering

Generate features for the AOIs/tiles from various datasets:

- Ookla Internet Speeds
- OSM POIs
- Rasters (e.g. Night Lights)

We computed features for every 2km around each DHS cluster



Options for dealing with NULLS

1. Imputation
    a. Fill in NULLS with values of “best guesses”
    b. For spatial data, we can do spatial imputation. I.e. get weighted average value of neighbors
        i. Ok especially for rasters and there are only small gaps in the data. Less ok if there are big clusters of NULLS.

5	6	7	8	9	10	11
6	7	8	9	10		12
7		9	10	11	12	13
8	9	10	11	12	13	14
9	10	11	12		14	15
10	11	12	13	14	15	16

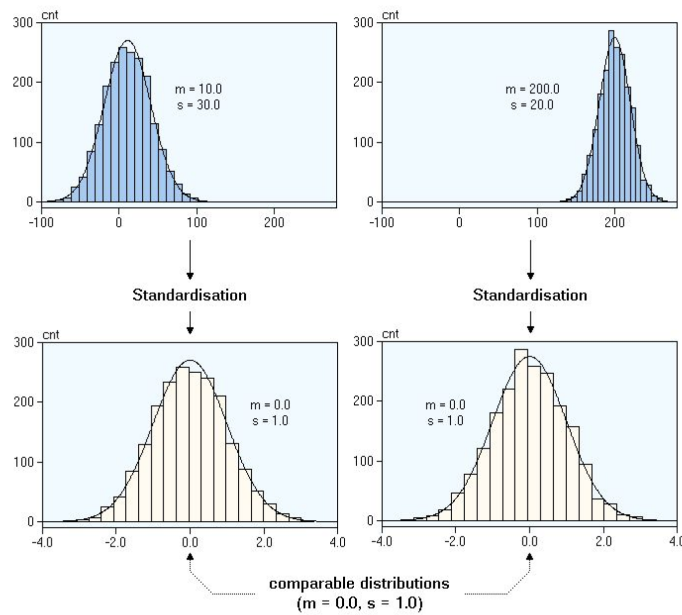
5	6	7	8	9	10	11
6	7	8	9	10	11	12
7	8	9	10	11	12	13
8	9	10	11	12	13	14
9	10	11	12	13	14	15
10	11	12	13	14	15	16

Options for dealing with NULLS

1. Imputation
    a. Fill in NULLS with values of "best guesses"
    b. For spatial data, we can do spatial imputation. I.e. get weighted average value of neighbors
        i. Ok especially for rasters and there are only small gaps in the data. Less ok if there are big clusters of NULLS.
2. Drop them entirely
    a. Rationale: Imputed values might reduce model performance
3. Assign a separate value for NULLS
    a. Some geospatial data might have this by default (e.g. -99999 value means NULL)
    b. Ok approach if using a tree-based model like random forest. Not ok if using a linear model or neural network!

Transformations

- Typically refers to scaling the features using either:
  - Min-max scaling
  - Z-scoring (e.g., mean-centering and standardization)



Rationale: make the features have comparable distributions so the model doesn't bias towards one feature over another

Transformations

- Typically refers to scaling the features using either:
  - Min-max scaling
  - Z-scoring (e.g., mean-centering and standardization)
- A necessary step in some model types (e.g., Logistic Regression, Neural Networks)
- For tree based models such as random forest, this is usually not necessary
- Note: The transformations on the dependent variable is handled separately from the features!
- For deployment:
  - Fit a transformer on the training set
  - Use the fitted transformer to transform the test set

Let’s try one experiment!

Select Features : Select basic features first

↓

Separate X’s and y’s : Necessary step for model input

↓

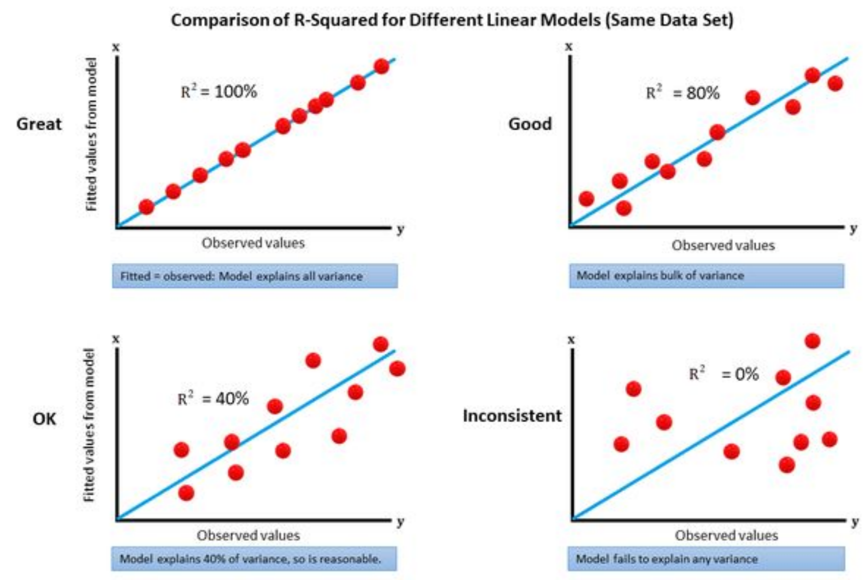
Train Model : Use default hyperparameters

↓

Evaluate Results : How well did our model do?

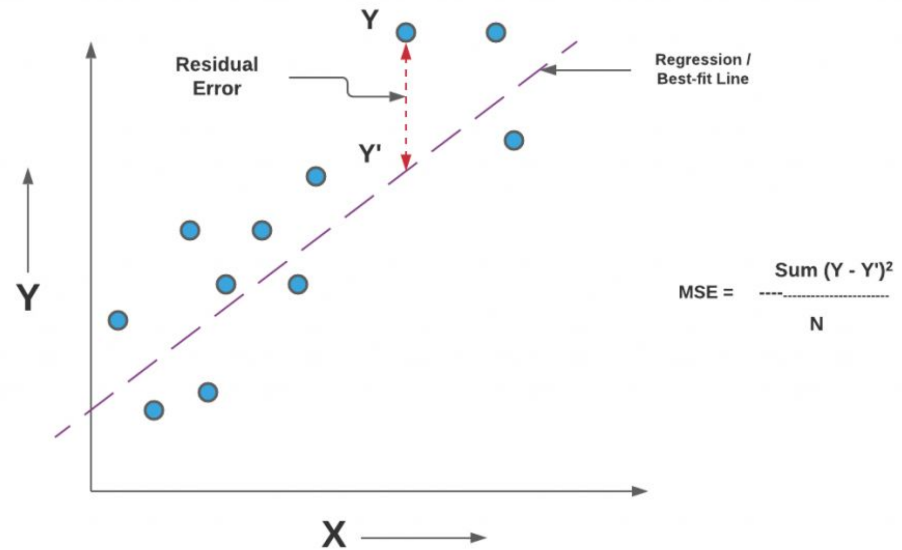
Regression Performance Metrics: MSE, MAE, R^2

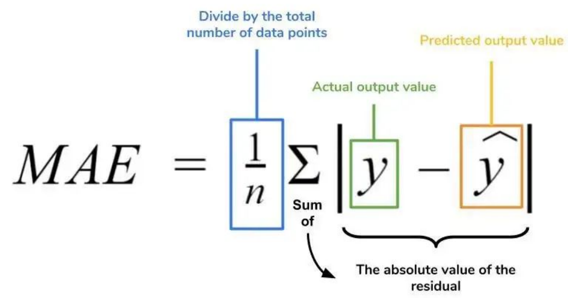
- **R^2:** it is the "percentage of variance explained by the model". The closer the value is to 100%, the better the model is at predicting the target variable.
- This metrics is independent of the original units. Can be useful for benchmarking how good a model is overall without context of the units.



Regression Performance Metrics: MSE, MAE, R^2

- **RMSE (Root Mean Squared Error):** Average error between target and prediction
- **MAE (Mean Absolute Error):** Mean error between target and prediction
- Both metrics quantify the error in terms of the original units.
- The MAE is preferred when dealing with skewed data since MAE is robust against exceptionally large values

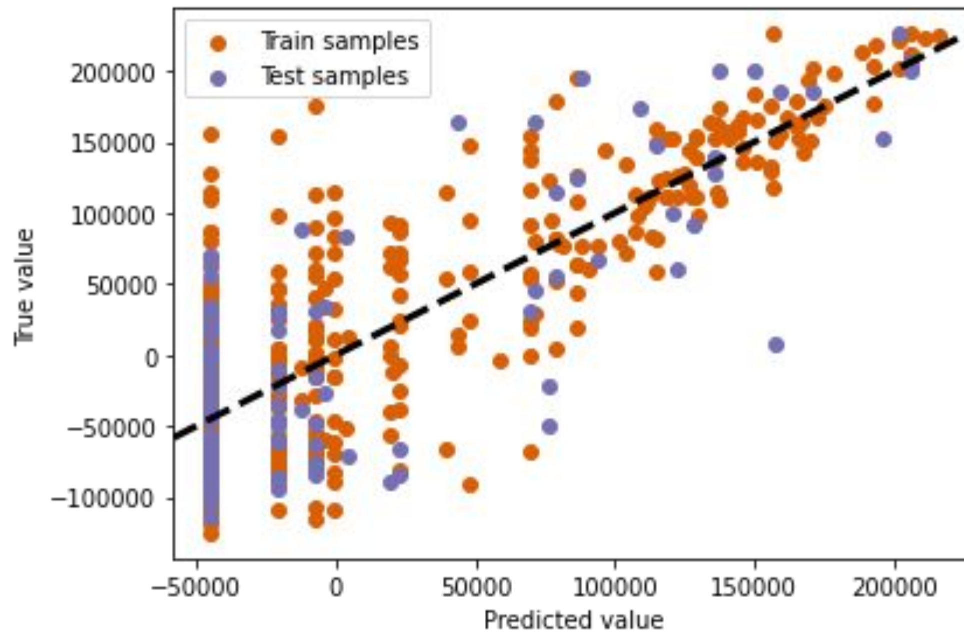




Initial Model Results [Regression]

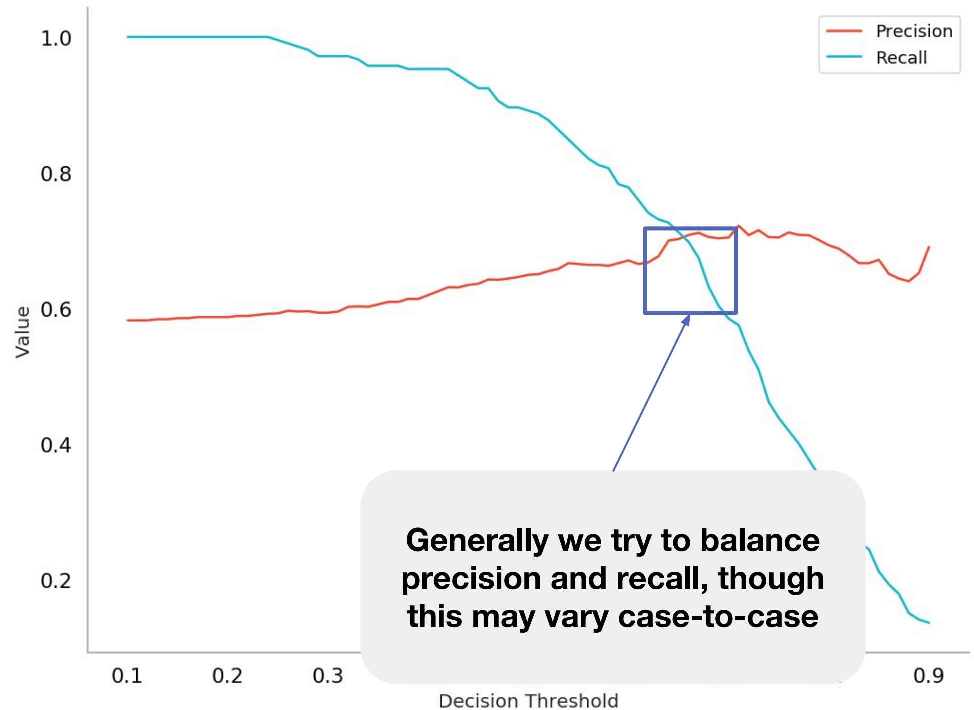
A common plot to evaluate a model is the predicted vs actual plot

Metrics for initial run:
Train R^2: 69%
Test R^2 62%



Classification Performance Metrics: Accuracy, Precision, Recall

- **Accuracy:** How many did we get right?
- **Precision:** Of those we tagged as positive, how many were true positives?
- **Recall:** Of the true positives, how many did we correctly tag
- All three can change depending on the **decision threshold** that you set



Generally we try to balance precision and recall, though this may vary case-to-case

Initial Model Results [Classification]

	value
train precision	1.000000
train recall	1.000000
train f1	1.000000
valid precision	0.675393
valid recall	0.708791
valid f1	0.691689
test precision	0.637615
test recall	0.655660
test f1	0.646512

These results are not good!

Overfitting

“Overfitting”

- Model is basically “memorizing” the training set
- The model does not actually learn any generalizable patterns
- Shows up as a large difference in performance metrics between training and test sets

What can we do to improve the model?

- Try removing outliers
- Try alternative transformations (e.g., min-max scaling)
- Try a different set of features or engineering new features
- Try tuning the model's hyperparameters
- Try different model types

Let’s try again, but changing a few things...

Select Features : Select a different set of features

↓

Separate X’s and y’s : Necessary step for model input

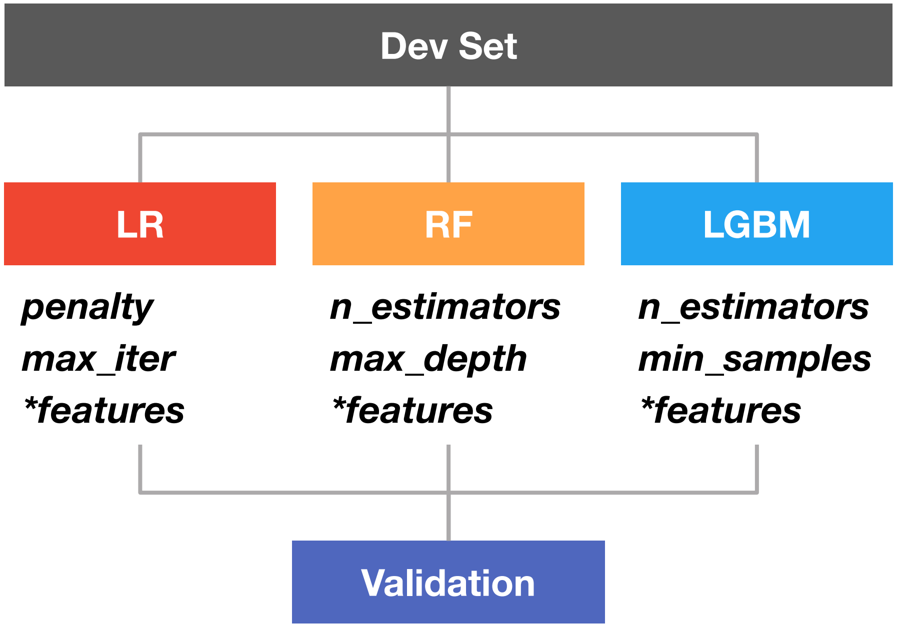
↓

Train Model : Try different hyperparameters

↓

Evaluate Results : How well did our model do?

Doing Multiple Experiments



Record of Results

	n_estimators	min_child_samples	feature_set	train size	test size	valid size	runtime	train precision	train recall	train f1	train roc
0	100	20	feature_set_01	438	318	364	0.222378	1.000000	0.984962	0.992424	0.992481
1	100	20	feature_set_02	383	318	364	0.203456	1.000000	0.982979	0.991416	0.991489
2	100	20	feature_set_03	383	318	364	0.229387	1.000000	1.000000	1.000000	1.000000
3	100	20	feature_set_04	383	318	364	0.178523	1.000000	0.978723	0.989247	0.989362
4	100	100	feature_set_01	438	318	364	0.161076	0.820611	0.808271	0.814394	0.767507
5	100	100	feature_set_02	383	318	364	0.154586	0.810345	0.800000	0.805139	0.751351
6	100	100	feature_set_03	383	318	364	0.172539	0.877729	0.855319	0.866379	0.833065
7	100	100	feature_set_04	383	318	364	0.172539	0.832599	0.804255	0.818182	0.773749

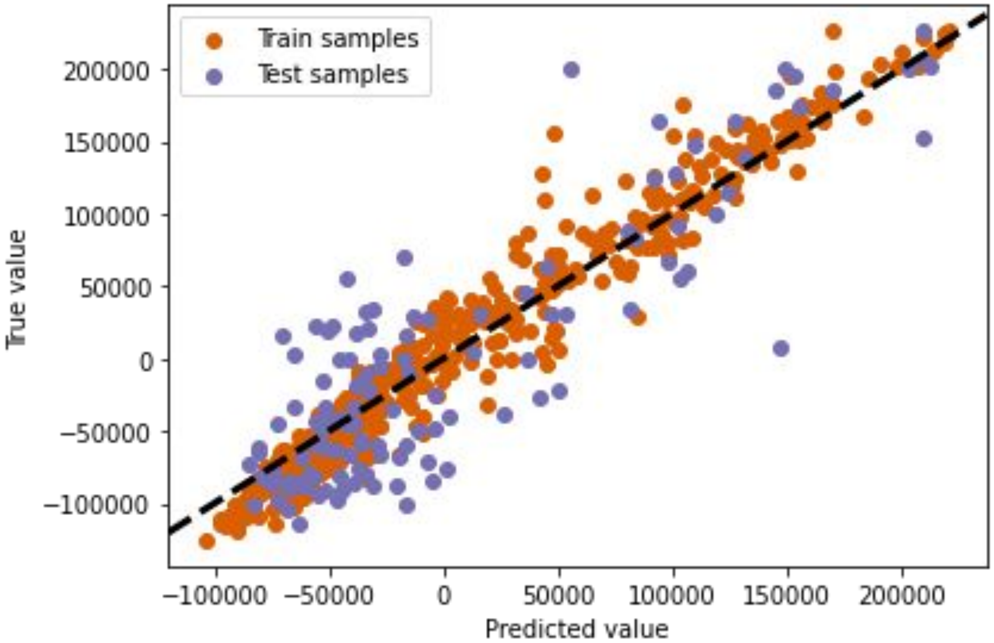
- At the end of all the experiments, select the “best” model

Model Results After More Features [Regression]

Metrics substantially increase after adding more features

Metric improvement:
Train R^2: 69% -> 95%
Test R^2 62% -> 76%

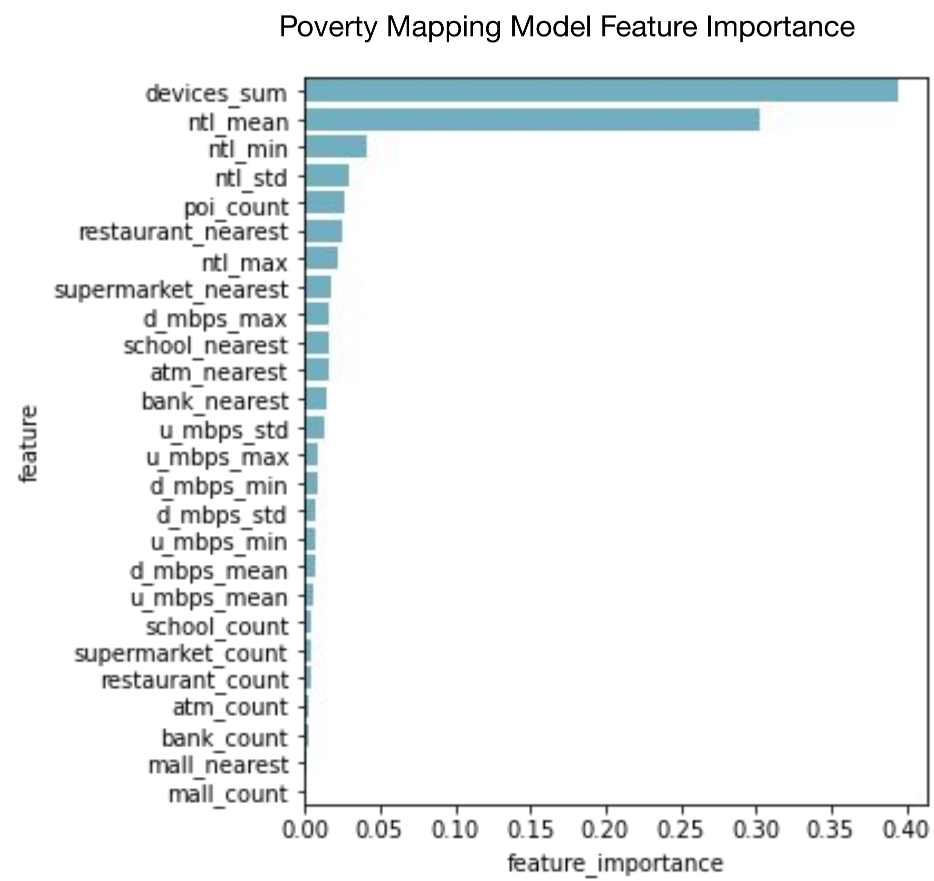
Note: test set R^2 is more representative of model performance. Could still improve through some hyperparameter tuning



Model Interpretability: Random Forest Feature Importance

We use feature importance metrics to determine which features are driving the predictions of the model.

Caveat of random forest feature importance: it doesn't show direction of impact



Model Interpretability using SHAP (SHapley Additive exPlanations)

Local Explainability

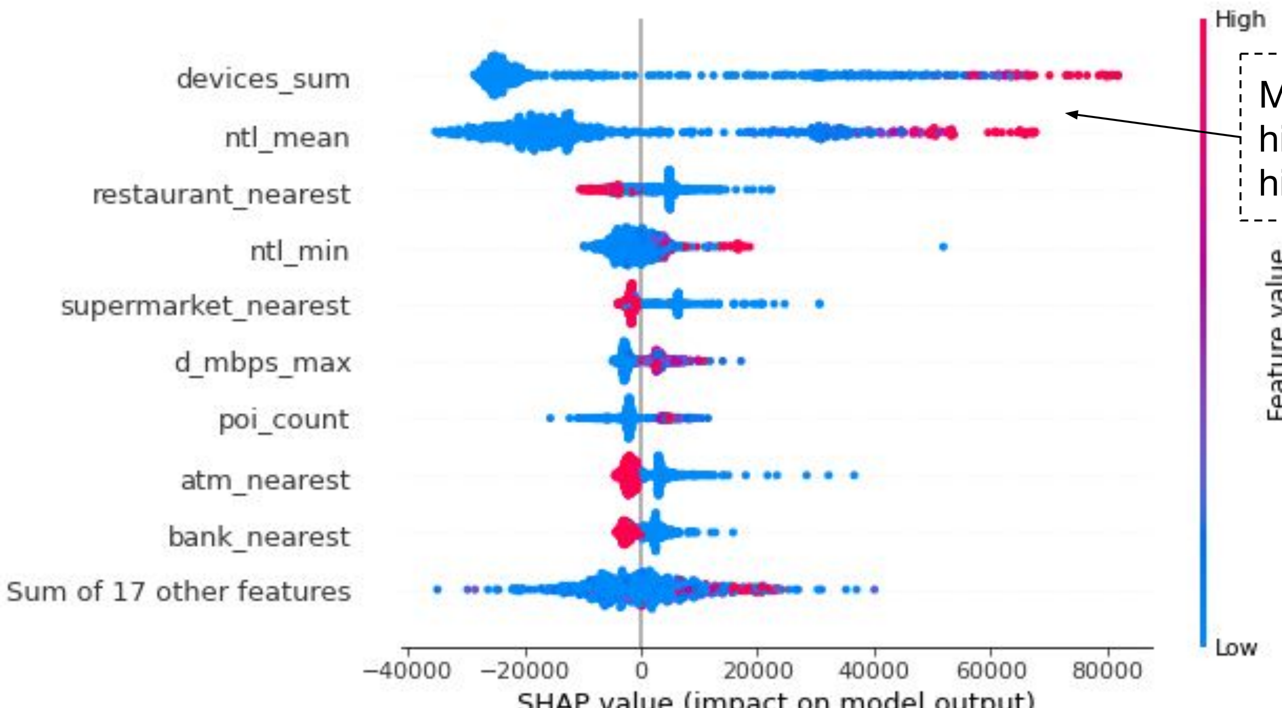
Model prediction

The values of these features are pulling up the prediction

These values of these features are pulling down the prediction

Model Interpretability using SHAP (SHapley Additive exPlanations)

Global Explainability using the Beeswarm Plot



More devices and higher NTL means higher wealth

05

Model Rollout

Overall Wealth Estimation Model Recap

DHS Clusters AOI Generation

Aggregate DHS data into cluster-level, and generate AOI geometries for each

Country Grid Tile Generation

Split up a country into equally sized grid tiles

Cluster Geometries

Tile Geometries

Feature Engineering

Generate features for the AOIs/tiles from various datasets:

- Ookla Internet Speeds
- OSM POIs
- Rasters (e.g. Night Lights)

Dataset for ML Training

Dataset for ML Inference

ML Modelling / Experiments

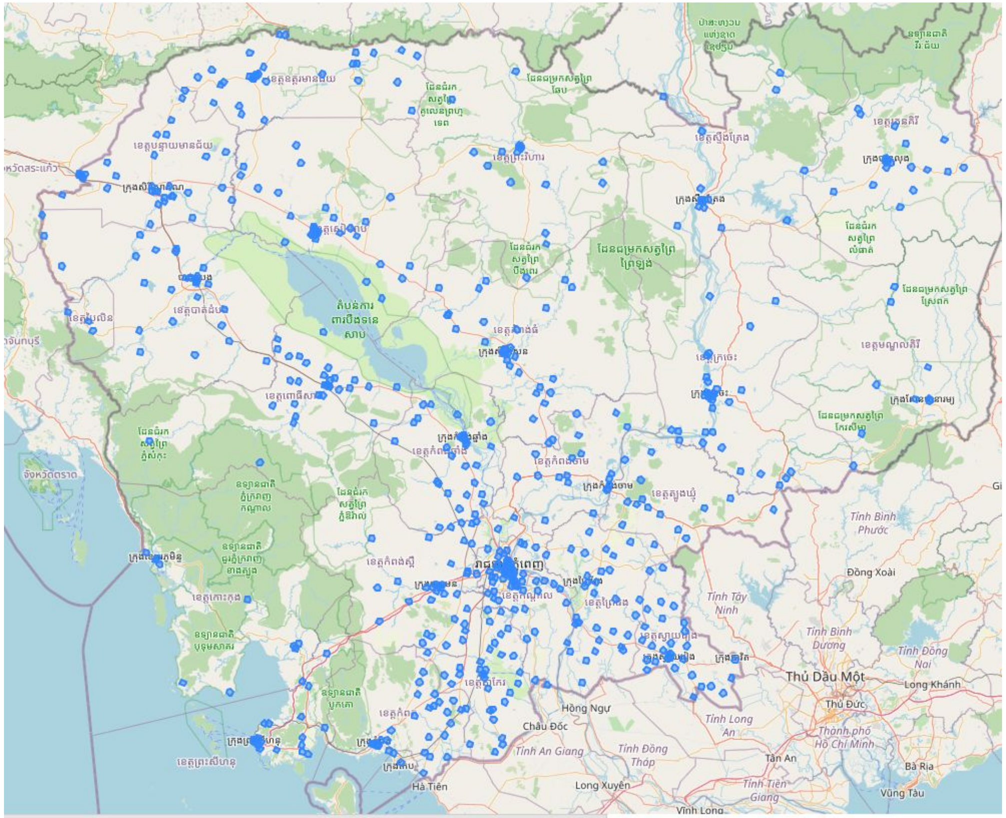
Train ML model using agreed-upon model (classification and/or regression) to estimate wealth for an area

Model

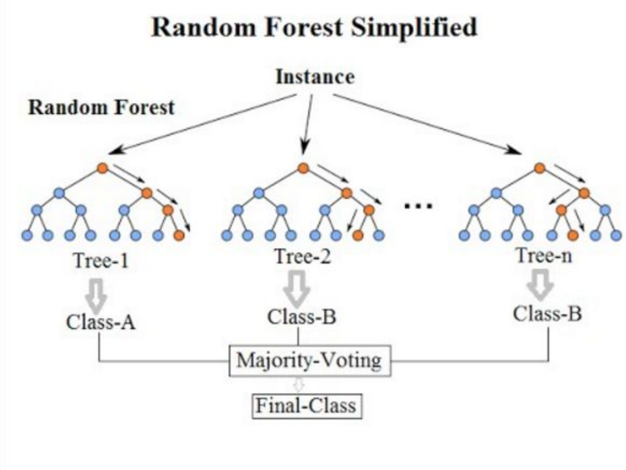
ML Rollout

Apply the model on country-wide grid tiles to estimate wealth

Recap: Given DHS cluster radius wealth + features, we trained a random forest wealth estimation model



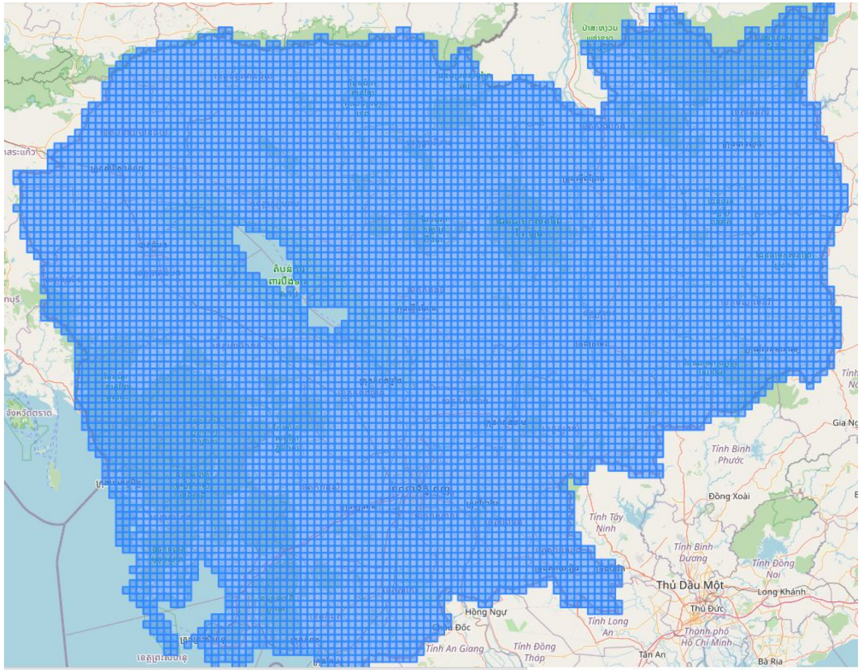
Model Training



Model Rollout

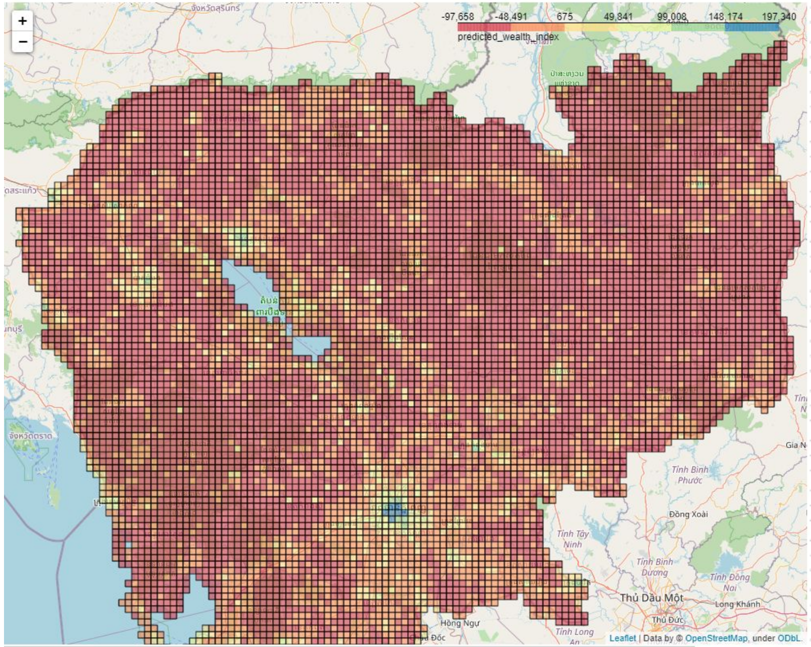
We can use our model to generate wealth estimates across the entire country

"Blank" grid tiles

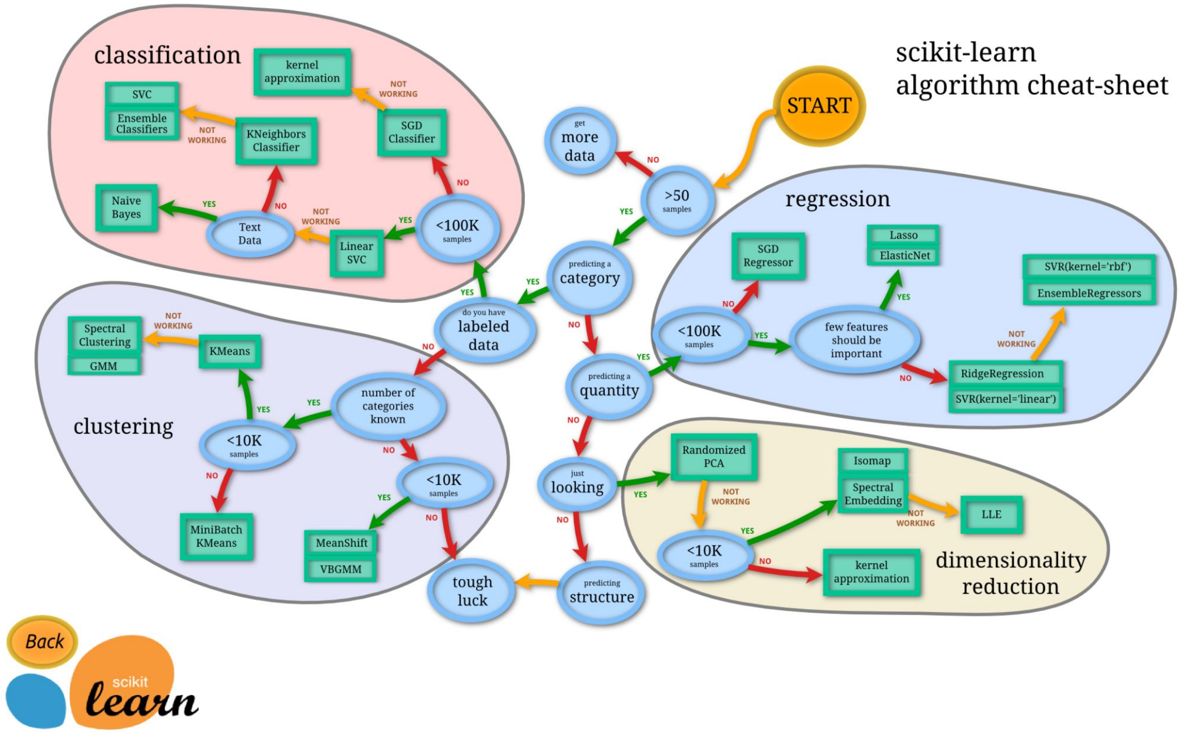


Model rollout

Country-wide grid tiles with wealth estimates



Machine learning is not limited to regression



- Recommendation Algorithms
- Deep Learning
- Reinforcement Learning
- Many more!

Computer Vision Use Cases

Computer vision mimics human vision to recognize patterns in images

Computer vision models can classify objects in an image after learning from human labeled examples.

Example: using satellite imagery to classify buildings into semi-detached, detached, and terrace house.

Thinking Machines'
Geospatial AI | Case Studies

Mapping informal settlements in Colombia

For the humanitarian agency iMMAP

CHALLENGE

Accelerate the **identification of new informal settlements** so humanitarian organizations can offer aid to refugees.

SOLUTION

Built an AI time series model to detect new informal settlements from satellite imagery

IMPACT

Detected **more than 350 new informal settlements** in 68 municipalities in just one month

USE CASE: GEOSPATIAL INTELLIGENCE FOR SELECTING RESTORATION SITES

Analyzing the suitability of shrimp ponds for climate smart aquaculture and mangrove restoration in ASEAN

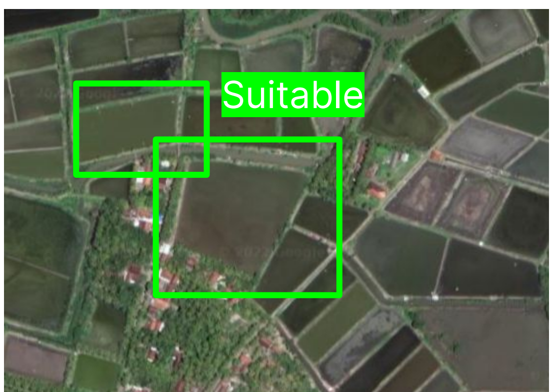
We are working with the global nonprofit **Conservation International** and **Arizona State University** to use computer vision and satellite imagery to identify fish and shrimp ponds in Indonesia and the Philippines that can be targeted for converting to sustainable aquaculture practices and mangrove restoration.

2021 Climate Change AI
Innovation Grant Winner

CONSERVATION INTERNATIONAL

Climate Change AI

ASU SCHOOL of SUSTAINABILITY
ARIZONA STATE UNIVERSITY



USE CASE: GEOSPATIAL INTELLIGENCE FOR ANALYZING FOREST COVER CHANGES

Using computer vision to map forest loss and direct investments in nature restoration

The Gerry Roxas Foundation is granting $16M to CSOs working on ecosystem restoration efforts in 30 bioregions of the Philippines. They want to use data to fund projects that directly address the main deforestation drivers in their areas of concern.

- Slash and burn
- Agricultural Expansion
- Biophysical Factors
- Mining
- Product Extraction

Time series (2016-2021) of greenery loss due to agriculture expansion in Cabanglasan, Bukidnon (Philippines)

Integrity Service Excellence
GERRY ROXAS FOUNDATION

Types of Computer Vision Tasks

There are four main computer vision tasks

- Image classification
- Object detection
- Semantic segmentation
- Instance segmentation

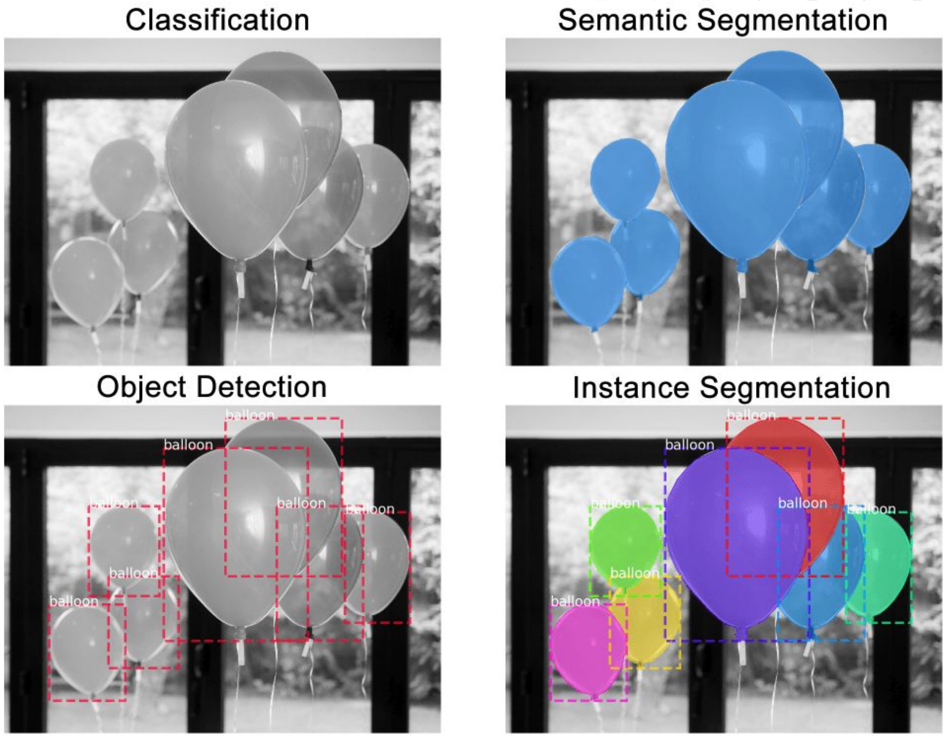


Image Classification

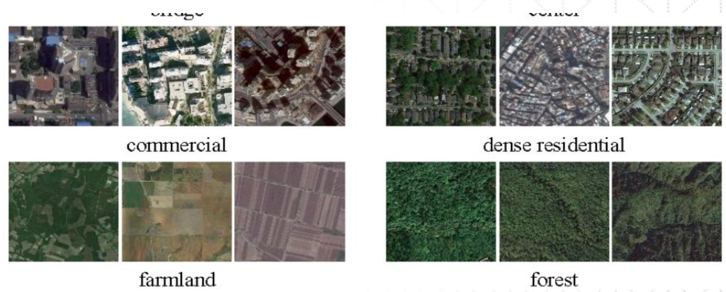
- One of the simplest computer vision tasks
- Determines the type or class label of objects in an image
  - **Input**: Image
  - **Output**: Class label (with probability score)

INTRODUCTION TO COMPUTER VISION

Image Classification

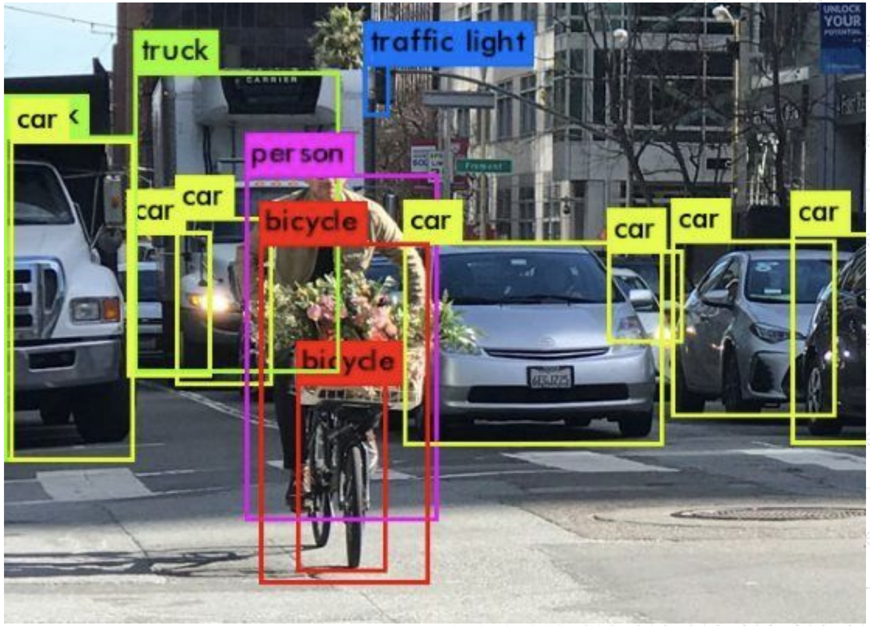
Applications in Earth Observation Data

- Image classification is often used for satellite image scene classification, e.g.
  - Aerial scene classification
  - Land use and land cover classification
  - Local climate zone classification



Object Detection

- Locates the presence of objects, with bounding box and class labels of the objects in the image
  - **Input**: Image with one or more objects
  - **Output**: Bounding boxes (defined by the bounding box coordinates) and labels



Semantic Segmentation

The process of classifying each pixel in an image belonging to a certain class label, a.k.a pixel-wise classification

- **Input**: Image
- **Output**: Prediction mask

Input

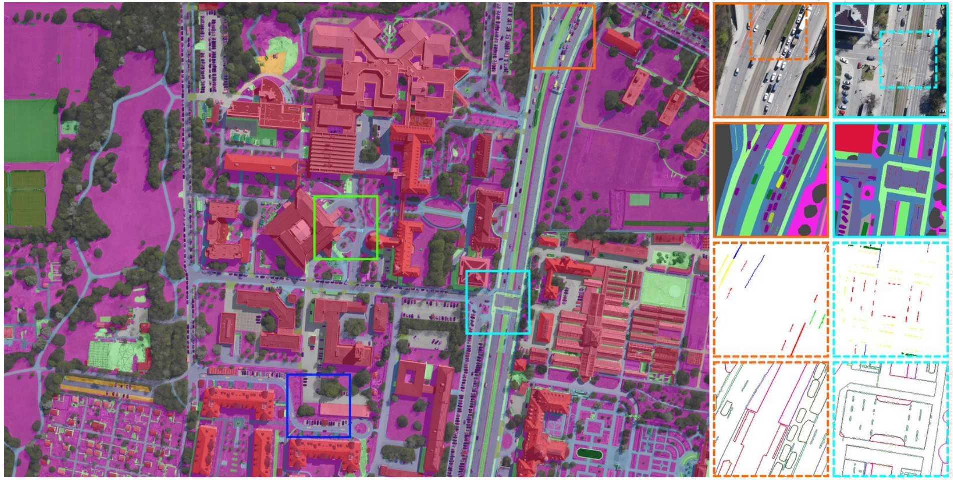
Prediction

Semantic Segmentation

Applications in Earth Observation Data

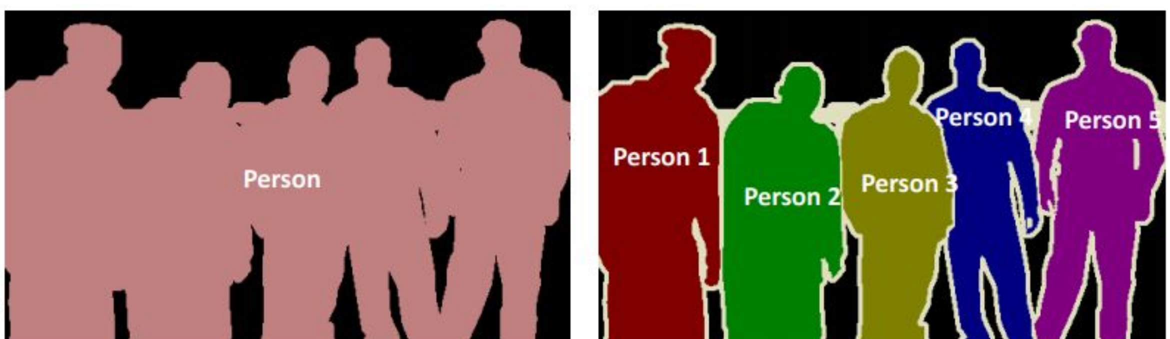
Commonly used for segmenting large land masses into different categories:

- Pixelwise land cover classification
- Agricultural crop classification
- Road network detection
- Crop field delineation
- Aerial scene segmentation



Instance Segmentation

- The problem with semantic segmentation is that it cannot distinguish between individual instances that are very close together.
- **Instance segmentation** remedies this by combining the benefits of both Object Detection and Semantic Segmentation in that it not only identifies the location of the object, but also the pixel mask of each particular object[+]



Semantic Segmentation **Instance Segmentation**

Instance Segmentation

Applications in Earth Observation Data

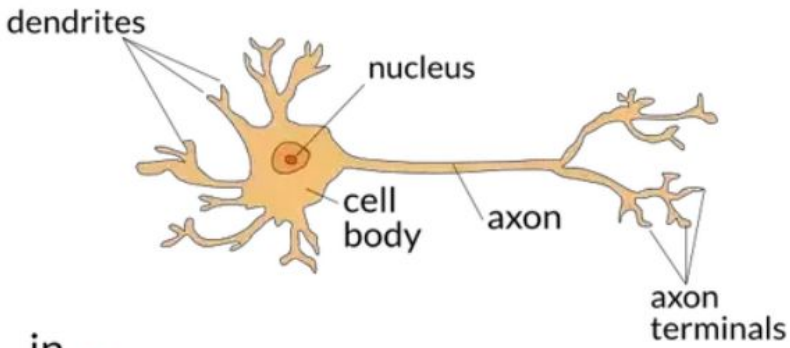
- Useful for Building Footprint Segmentation
    - Especially in cases where the buildings are in close proximity to one another (e.g. terraced buildings, dense subdivisions, etc.)

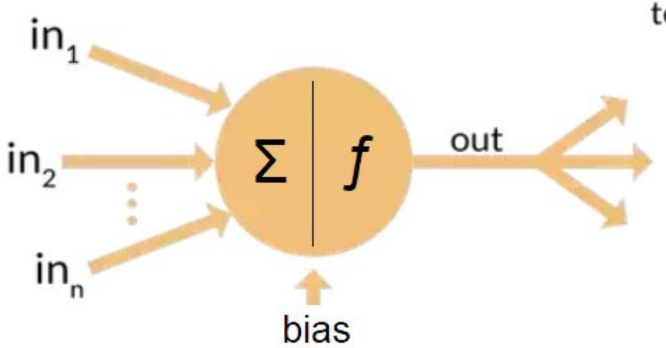
[Extra Materials] Convolutional Neural Networks

The building blocks of
modern computer vision

Neural Networks

Or, **networks of neurons**





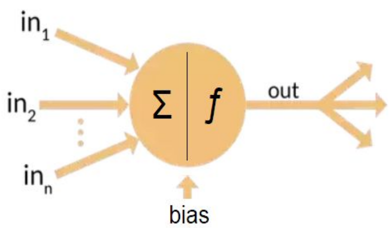
Biological Neuron
transmits information between different areas of the brain, enabling it to **form understanding and connections**

Artificial Neuron
Transmits information from the input to the output, in order to arrive at an optimal answer

https://towardsdatascience.com/the-differences-between-artificial-and-biological-neural-networks-a8b46db828b7

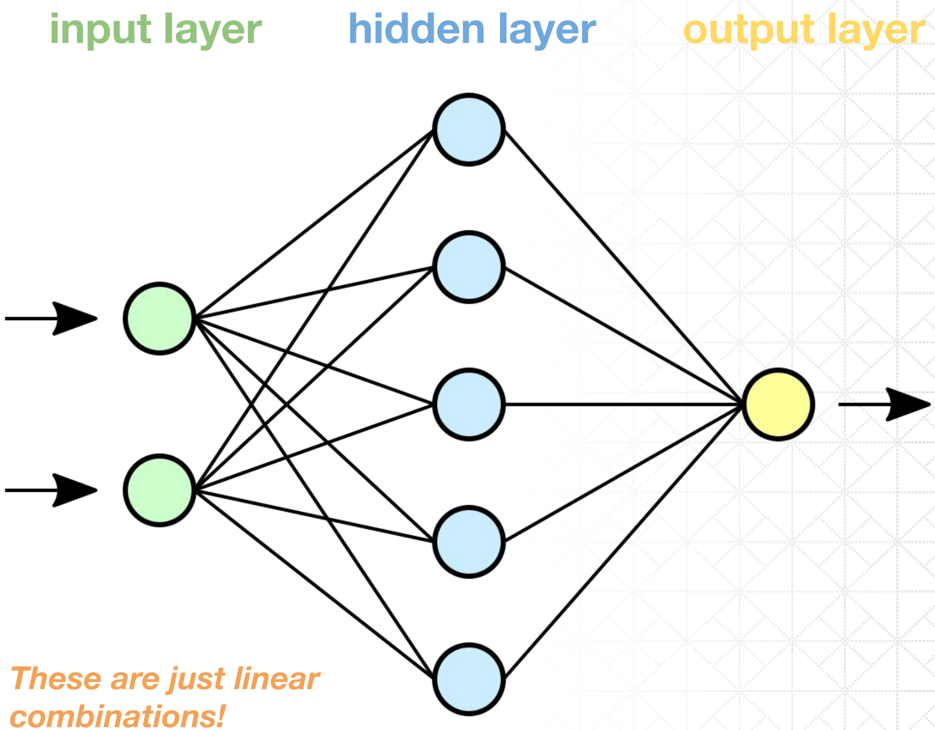
Neural Networks

Or, **networks of neurons**



Stacking and connecting many neurons = a neural network

Aka Multilayer Perceptron



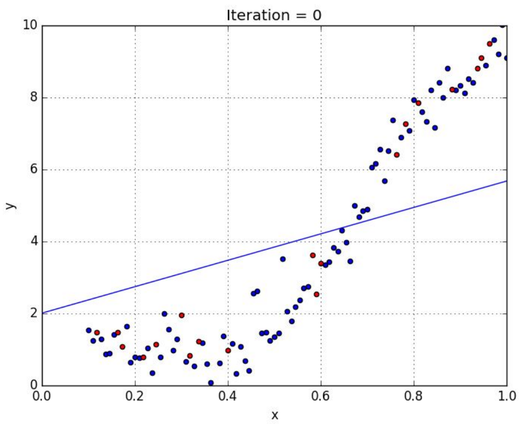
These are just linear combinations!

$y = x1w1 + x2w2 + \ldots + b$

How do Neural Networks learn?

An intuition of the training process

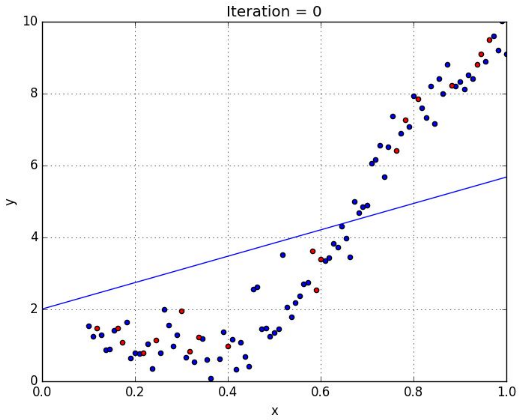
Recall: a linear model starts out randomly, and *iteratively* minimizes the error until it reaches the minimum.



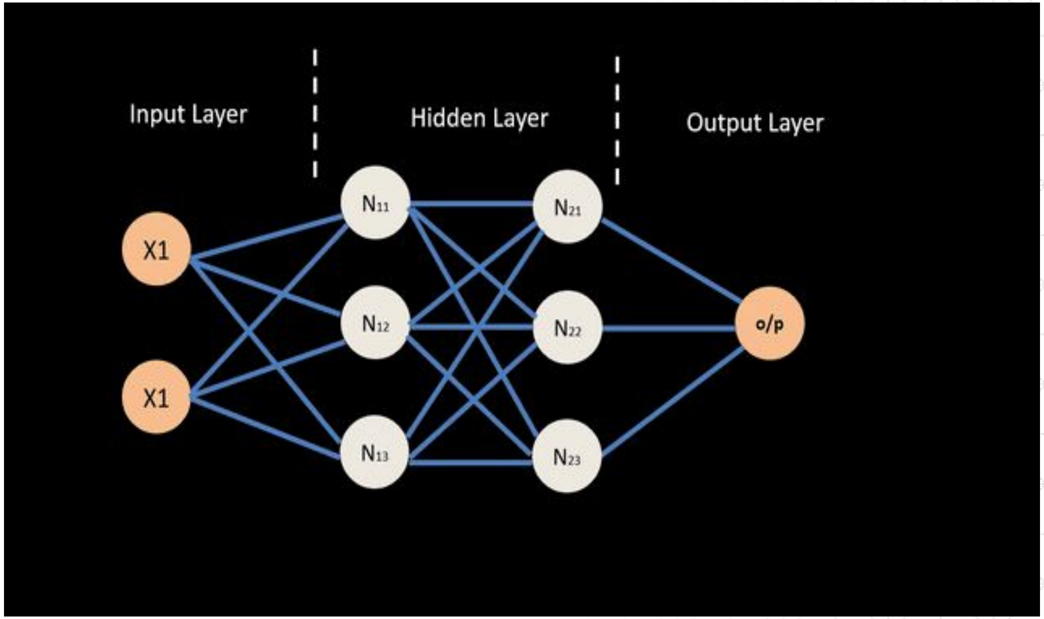
How do Neural Networks learn?

An intuition of the training process

Recall: a linear model starts out randomly, and *iteratively* minimizes the error until it reaches the minimum.



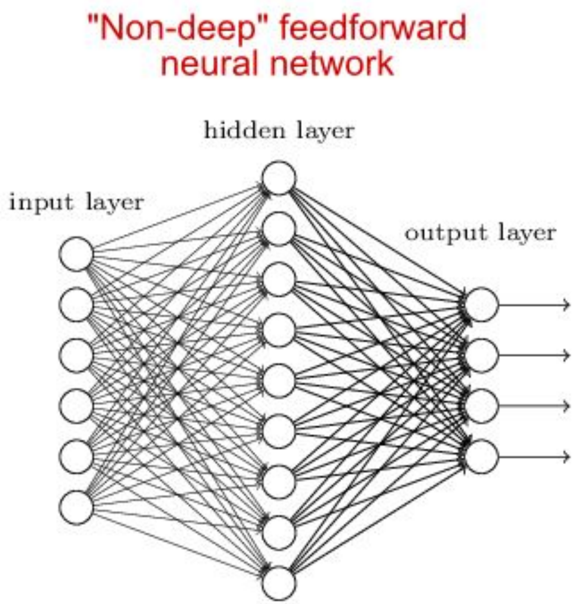
A neural network learns the same way. You can think of each node as just a linear model whose weights are iteratively updated to produce optimal outputs.
Here's how it looks for *one iteration*:

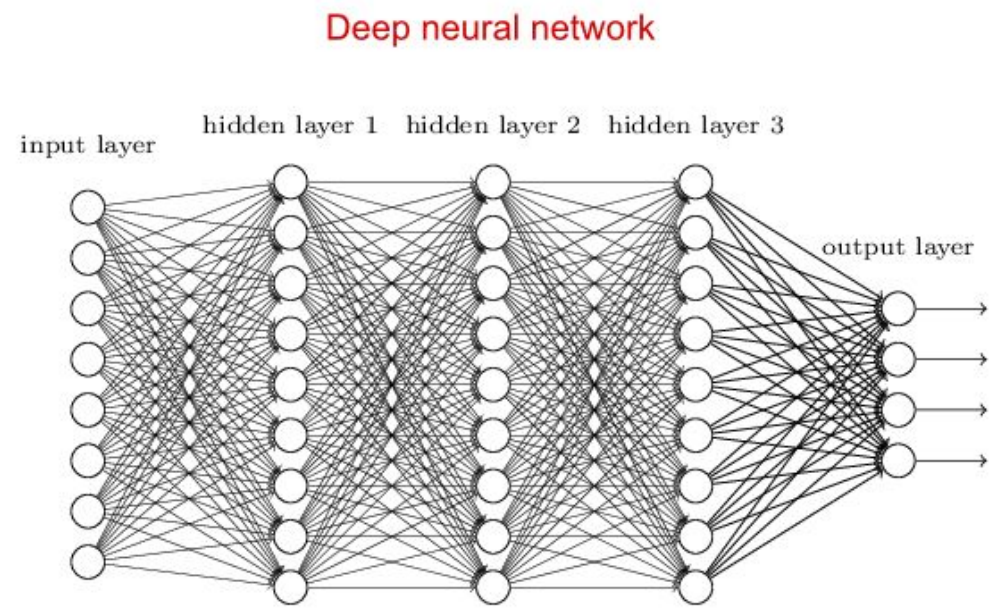


https://machinelearningknowledge.ai/animated-explanation-of-feed-forward-neural-network-architecture/

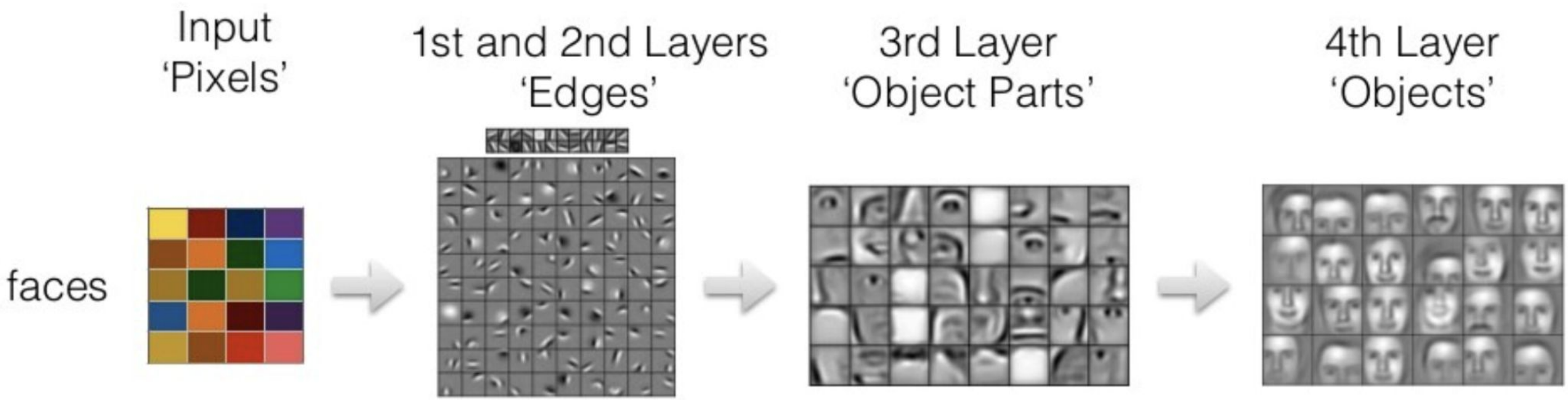
Deep Neural Networks

Simply means you're stacking many of the hidden layers





“Automatic” feature extraction



In early layers, these representations may be of edges or textures of the image.

As you go deeper, complex structures (e.g. ears and nose) start to be detected.
(which is why, in general, deeper networks are more powerful!)

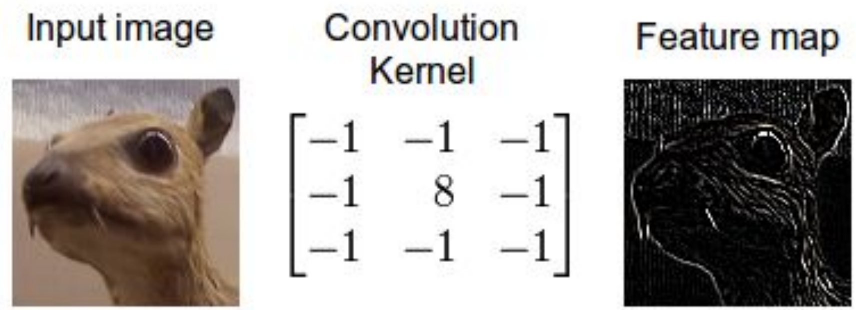
https://towardsdatascience.com/the-differences-between-artificial-and-biological-neural-networks-a8b46db828b7

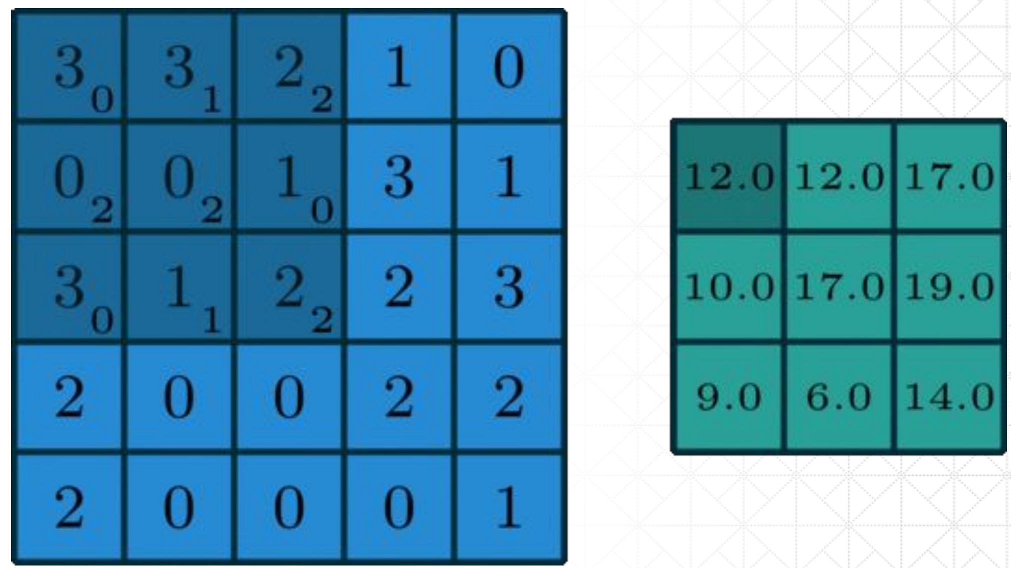
Convolutions

Operation that multiplies convolution kernel to the input image

Generates feature maps highlighting aspects of the image (such as edges in this example)

Derives from classical image feature extraction techniques such as image gradient vectors and histogram of gradients (HOG)

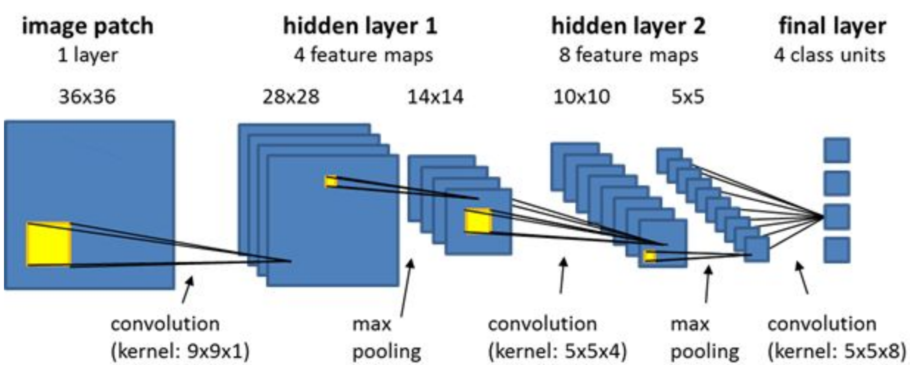




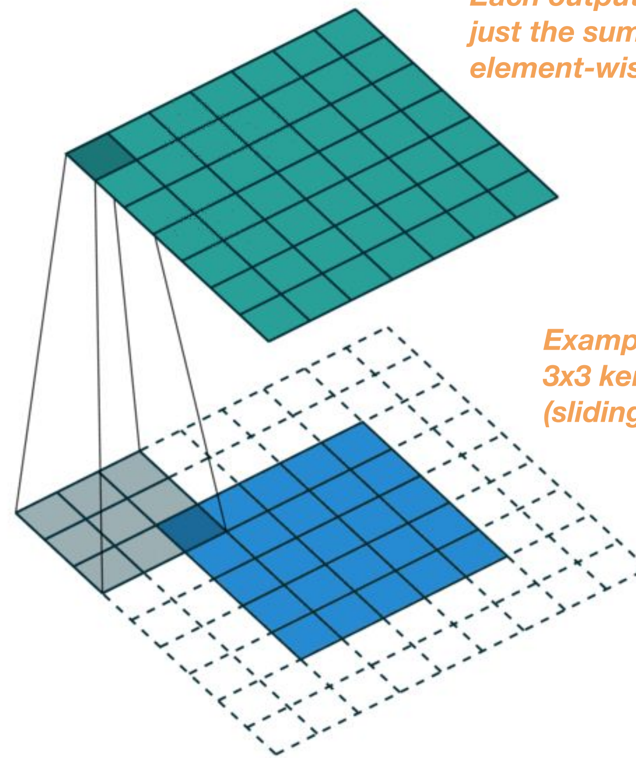
https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1

Convolutional Neural Networks?

CNNs are neural networks that make use of convolutional layers to learn smaller or more specialized patterns in an image.



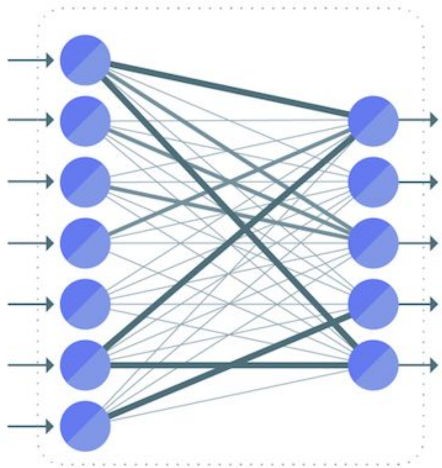
Each output neuron is just the sum of the element-wise product!



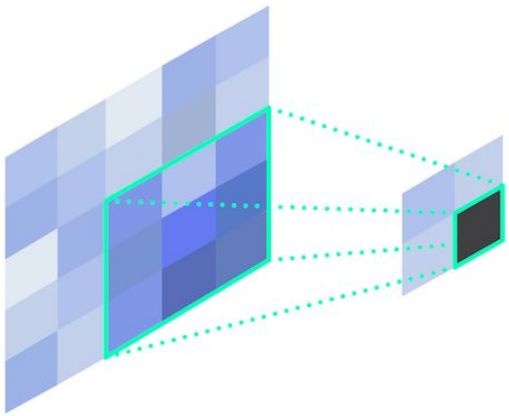
Example shows a 3x3 kernel (sliding window)

https://towardsdatascience.com/the-differences-between-artificial-and-biological-neural-networks-a8b46db828b7

Two most common layers



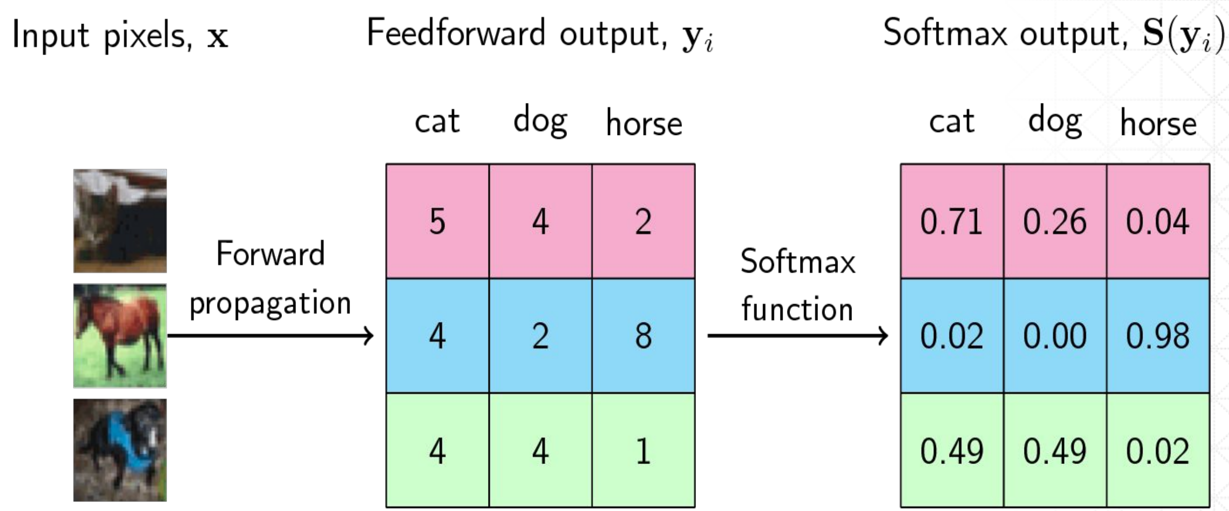
Fully-connected Layer (FC)
Each neuron in a fully-connected layer is *calculated using all of the neurons* of the previous layer



Convolutional Layer (Conv)
Each neuron in a convolutional layer is *calculated from a sliding window* of the previous layer

https://towardsdatascience.com/four-common-types-of-neural-network-layers-c0d3bb2a966c

Output classification as probabilities

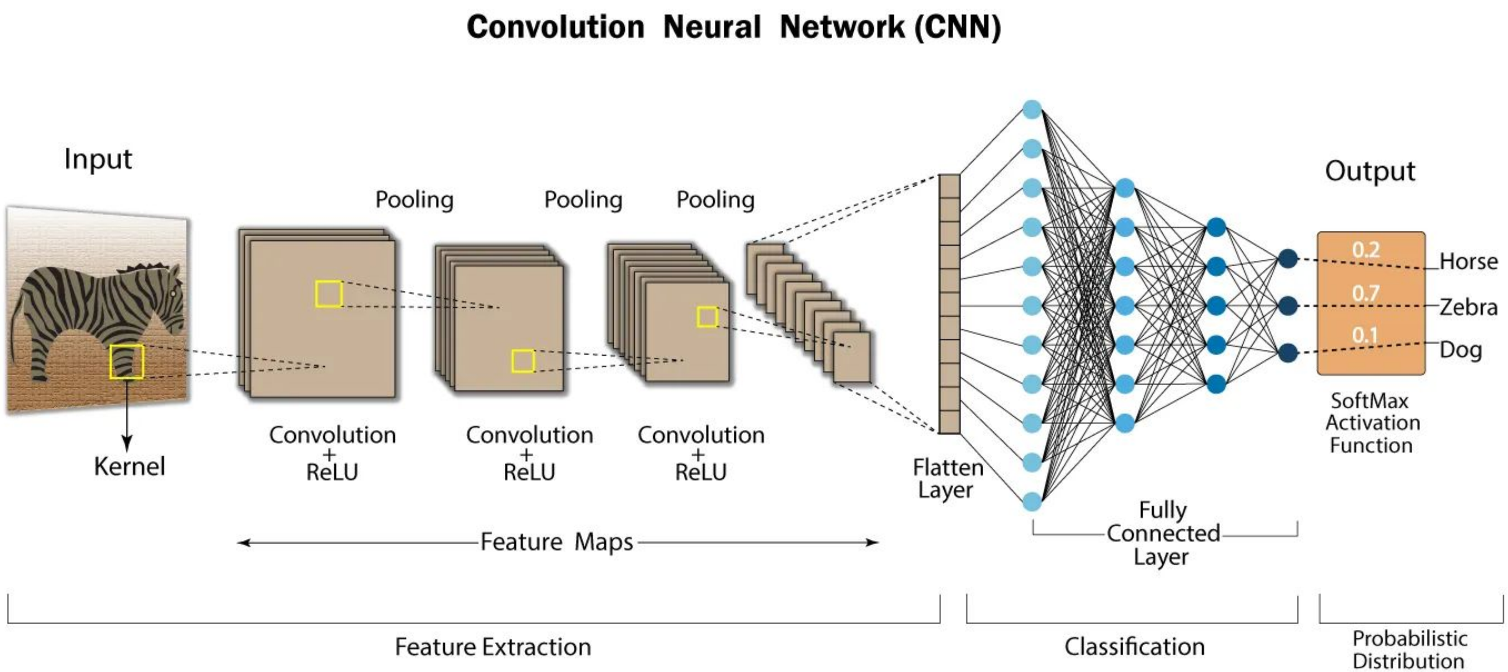


Softmax

Maps a set of numbers to probabilities, such that the probabilities sum up to 1.

https://ljvmiranda921.github.io/notebook/2017/08/13/softmax-and-the-negative-log-likelihood/

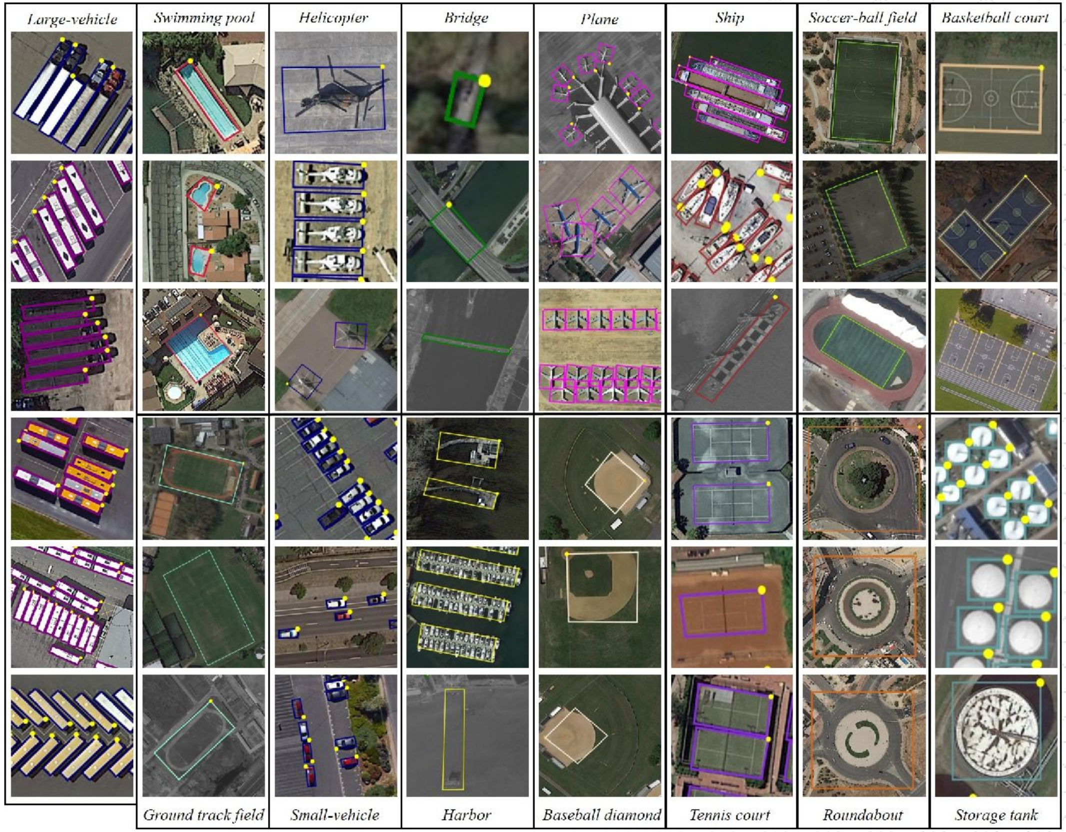
Common CNN architecture



Object Detection

Applications in Earth Observation Data

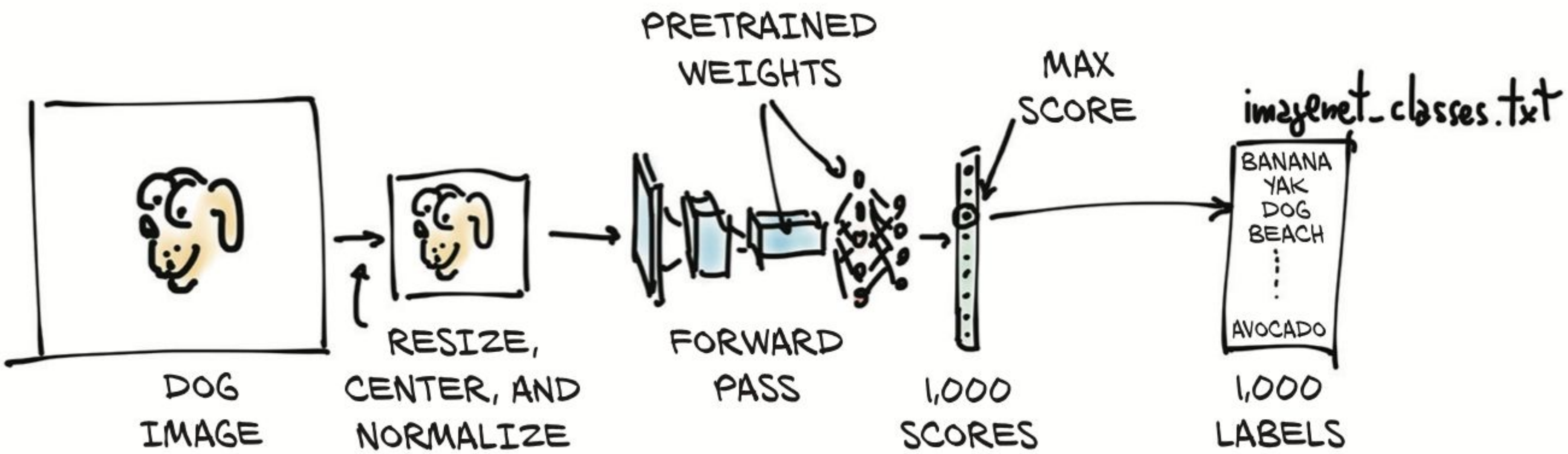
- Commonly used for locating different types of objects from aerial images
  - e.g. trees, cars, swimming pools, sports fields, oil tanks, etc.
- DOTA Dataset
  - Popular large-scale Dataset for Object Detection in high-resolution satellite imagery



Convolutional Neural Networks

Transfer Learning

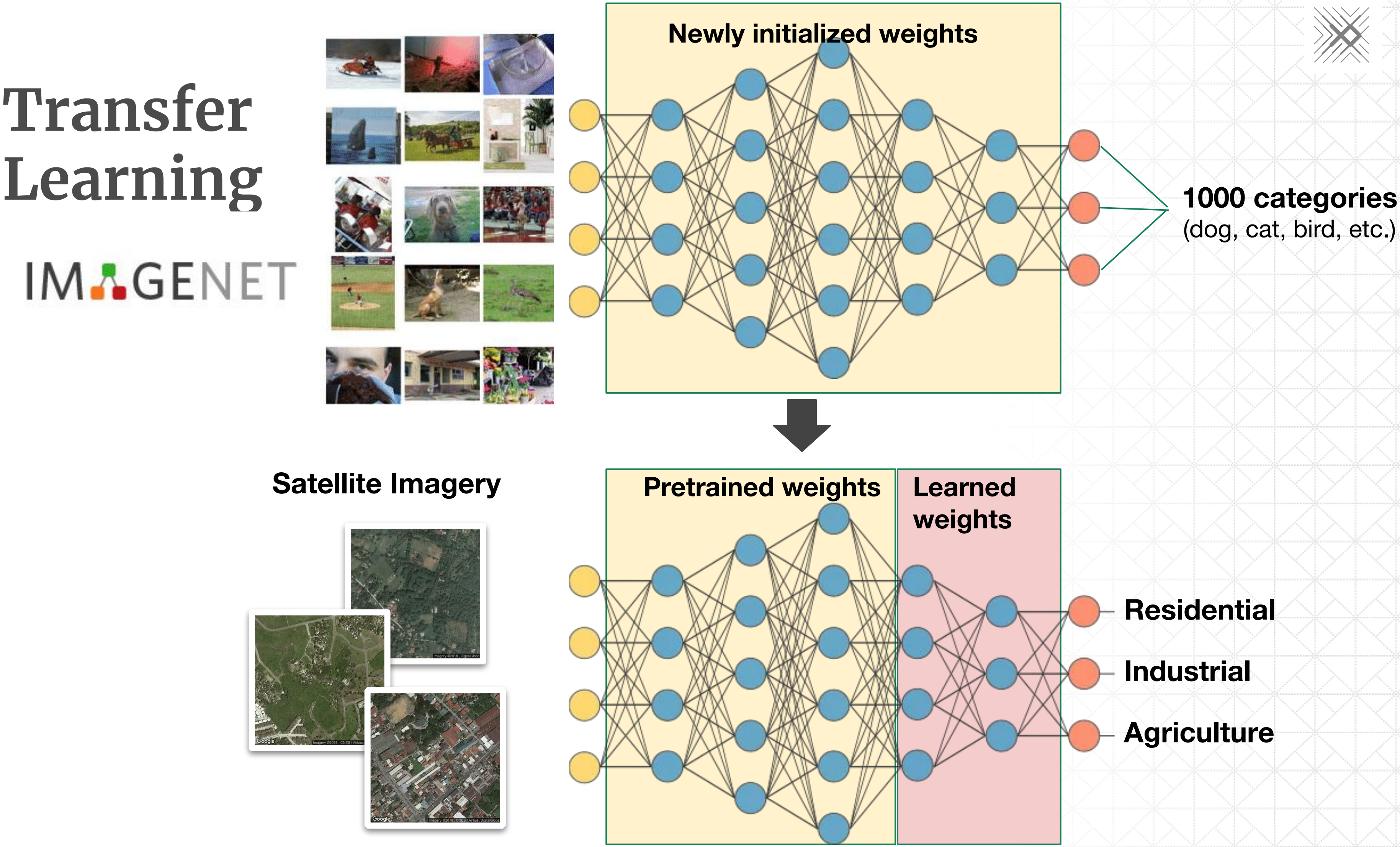
Using a pretrained model to fit the data



Most convolutional neural networks today are pretrained on the ImageNet Dataset

- The ImageNet dataset contains 14 million annotated images
- Pretraining: training a baseline model that has a general understanding of shapes, colors, objects, etc.
- The pretrained model can be fine tuned for more specific tasks

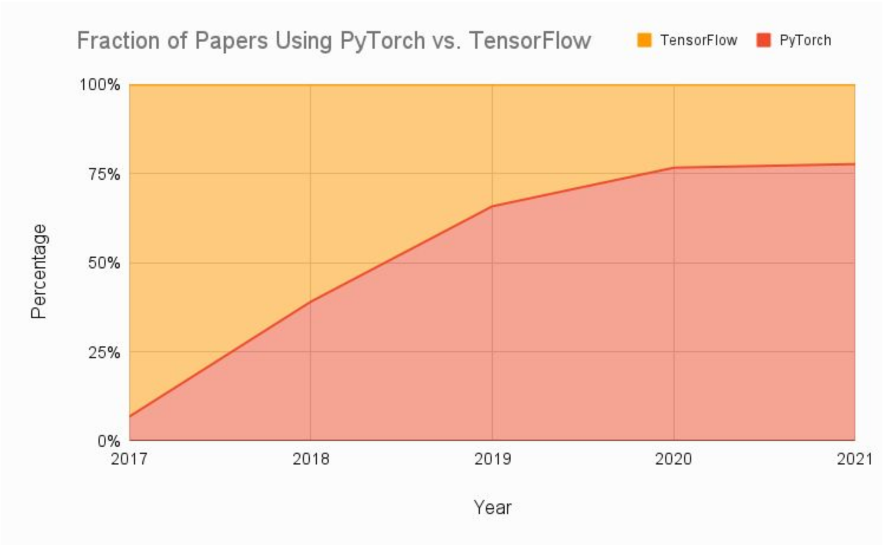
Transfer Learning



Intro to Pytorch

Pytorch is a the most used framework for machine learning today

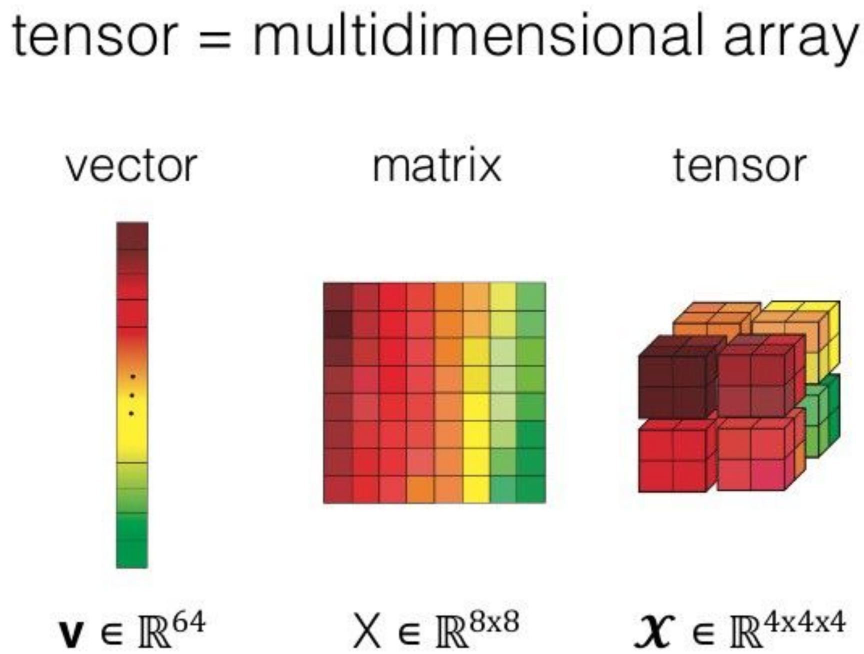
- "an optimized tensor library for deep learning using GPUs and CPUs."
- Written in C++ and CUDA (Nvidia)



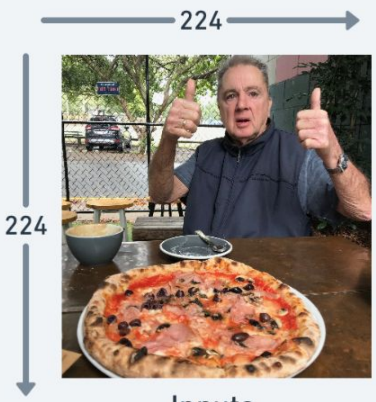
PyTorch Tensors

Tensors are simply **multidimensional arrays** (just like NumPy arrays!) with support for accelerated mathematical operations implemented in C++

These are **used to represent *all* kinds of data in PyTorch** (e.g. inputs, representations, weights, outputs)



PyTorch Tensors



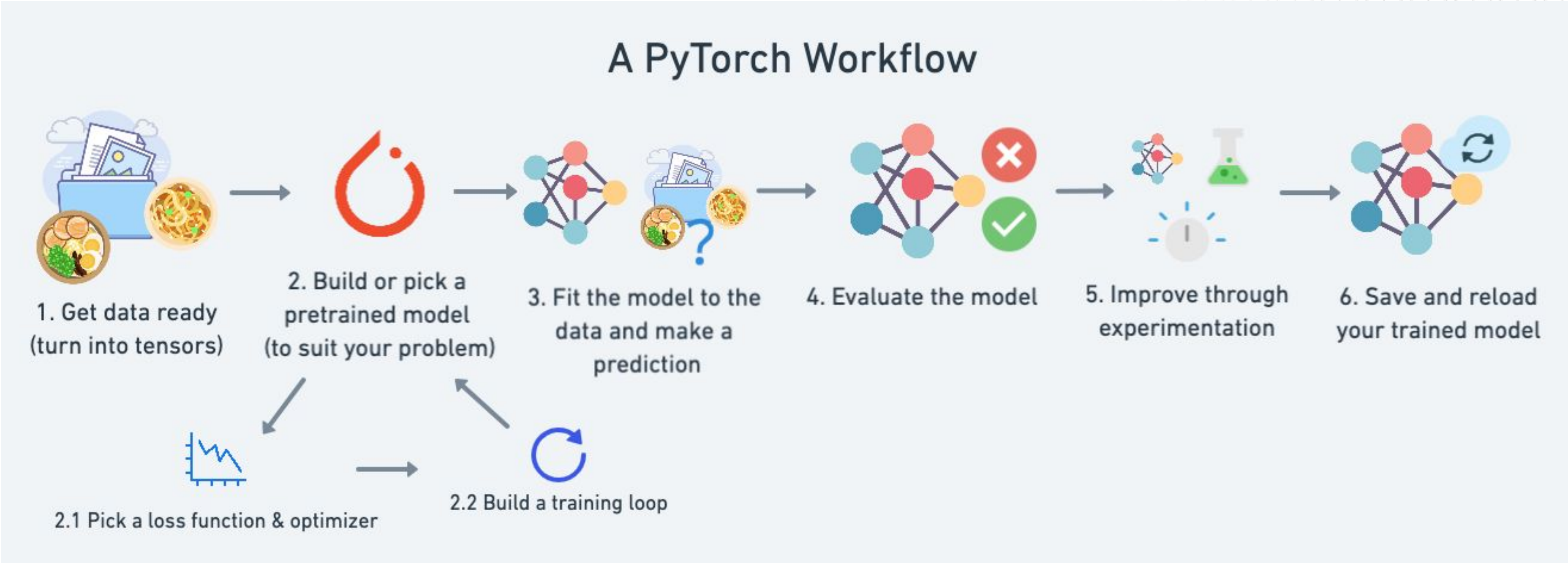
Inputs
(could be almost anything)

Processed inputs
(these get represented as a tensor, the processing will depend on the input)

[[[-0.01157, 0.02485, 0.02878...
-0.01271, 0.03971, 0.08827...
0.02680, 0.05589, -0.01068...
-0.00597, 0.00639, -0.01819...
...]]]

Tensor representation of image
with shape [3, 224, 224]

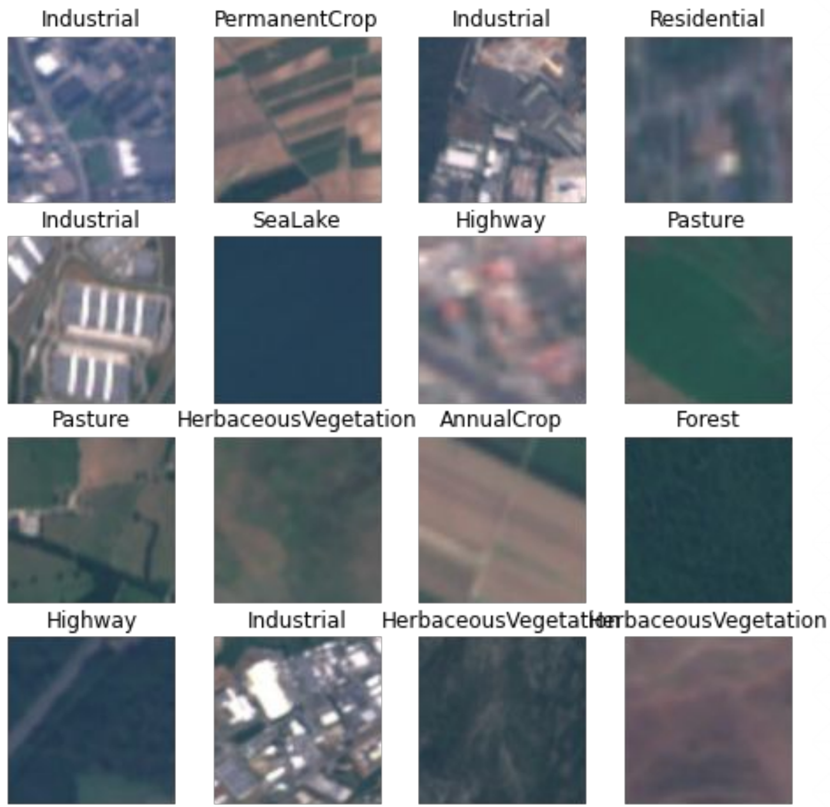
After we get data ready into tensors, we build a model



Land Cover Land Use Classification

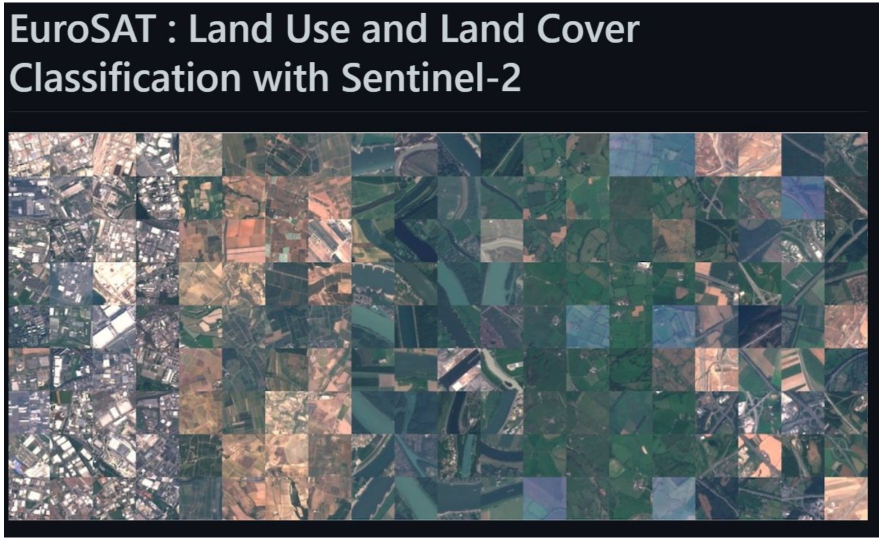
LULC: Land Cover and Land Use Classification

Objective: Classify tiles of land based on the satellite imagery

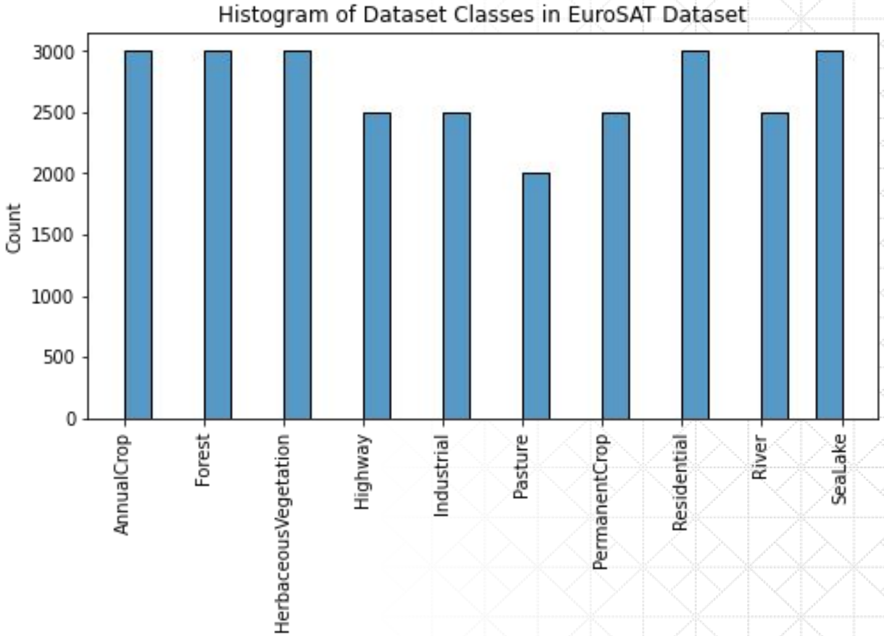


EuroSAT dataset

EuroSAT: open dataset of grid tiles across Europe with LULC classification



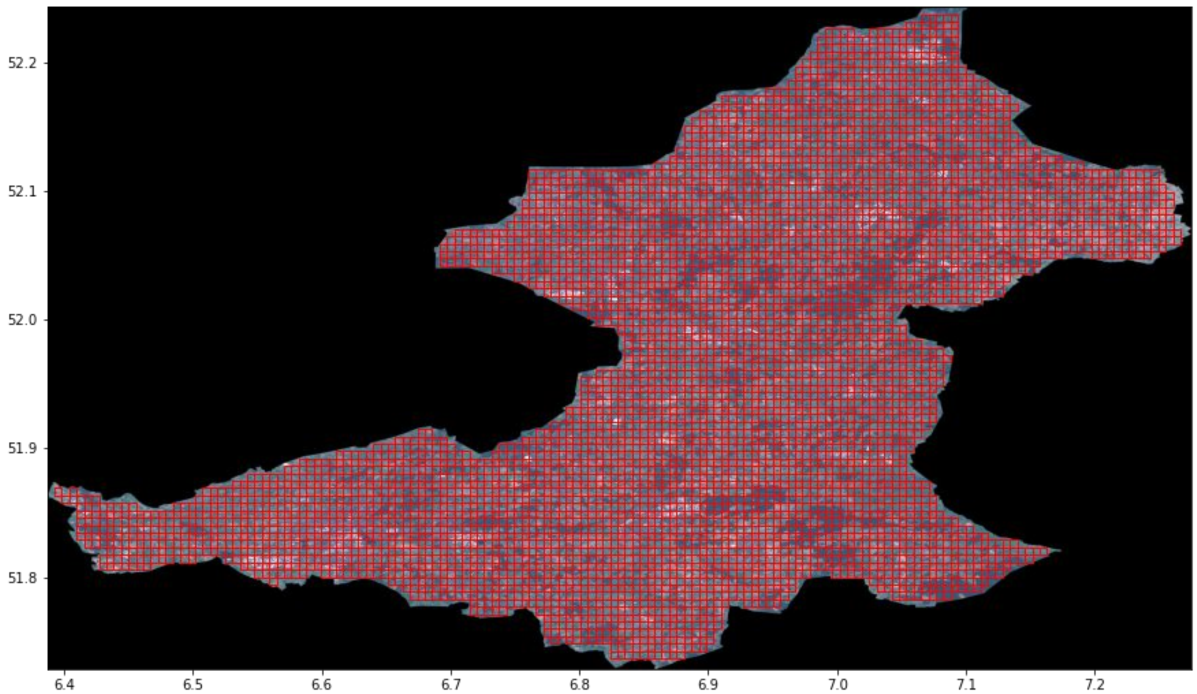
10 Classes in the EuroSAT dataset



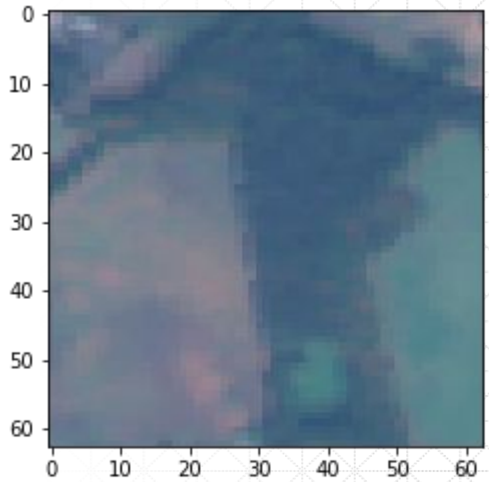
Sample use case:classify land in given study area

Example: Kreis Borken region in Germany

We first grid the entire region before deploying the model

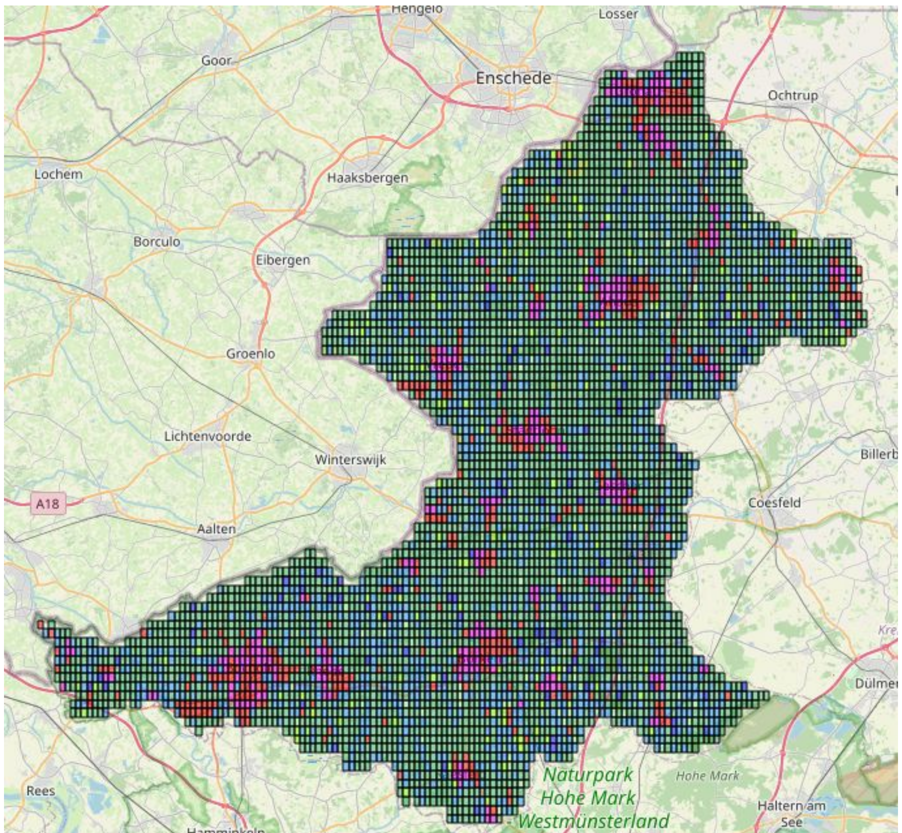


This is the zoomed-in image of 1 tile



Land Cover Land Use Classification

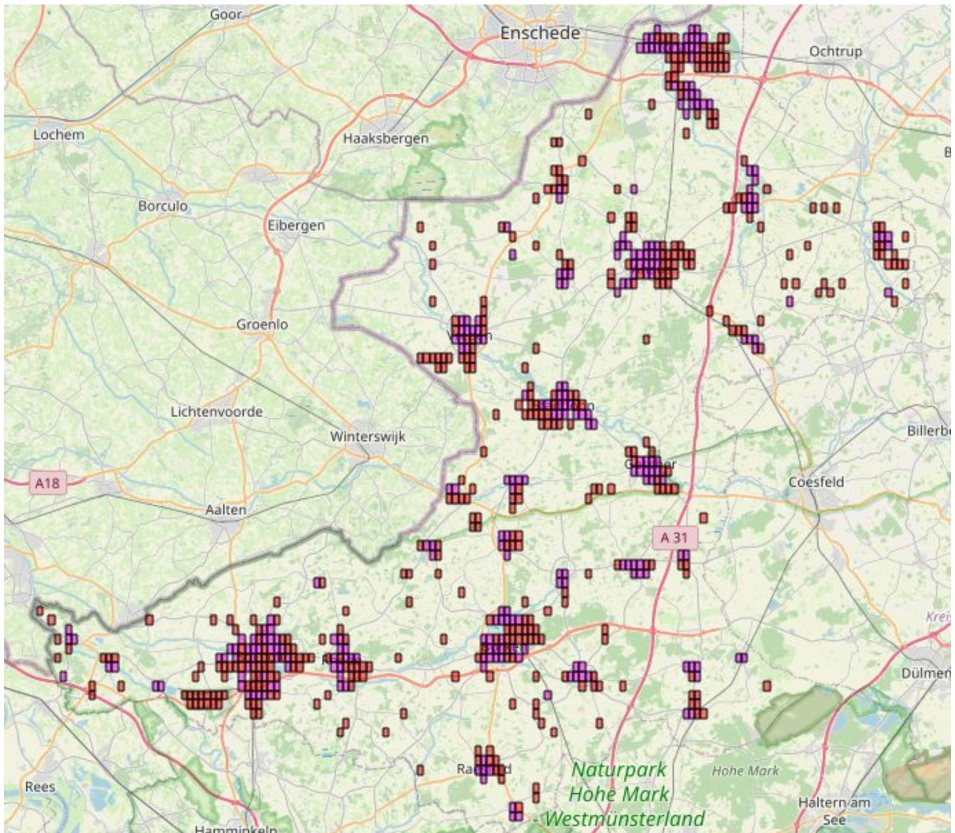
After getting a computer vision model ready, we can roll this out across the entire region



Legend for Classes

- AnnualCrop
- Forest
- HerbaceousVegetation
- Highway
- Industrial
- Pasture
- PermanentCrop
- Residential
- River
- SeaLake

We can filter the map for further analyses



Legend for Classes

- [ ] AnnualCrop
- [ ] Forest
- [ ] HerbaceousVegetation
- [ ] Highway
- [x] Industrial
- [ ] Pasture
- [ ] PermanentCrop
- [x] Residential
- [ ] River
- [ ] SeaLake

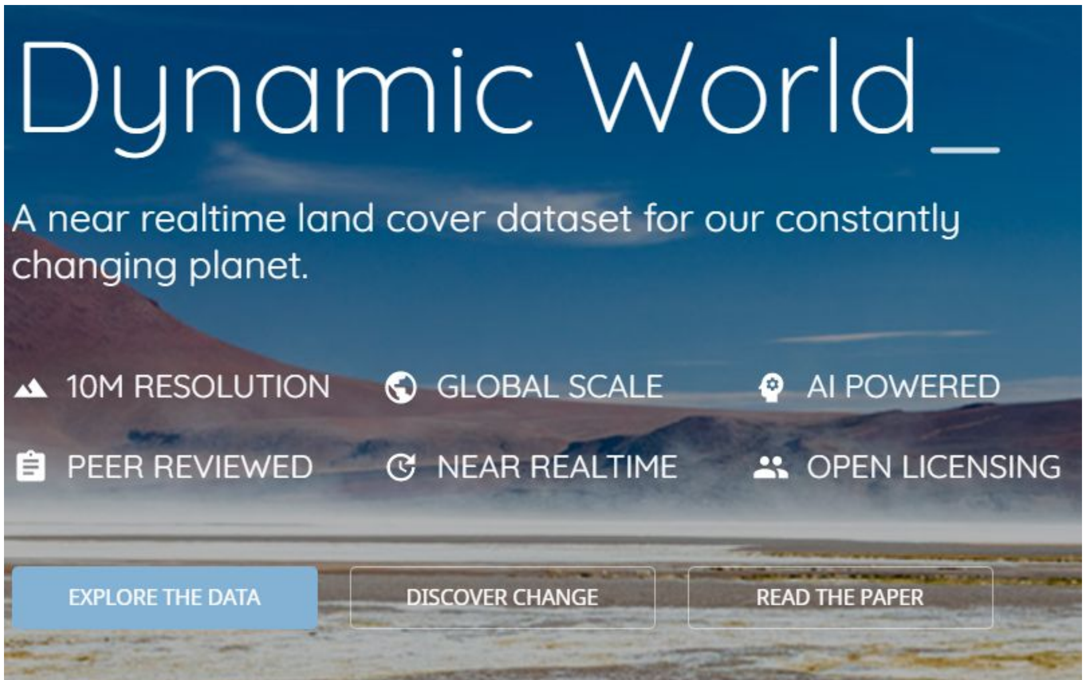
Land Cover Land Use Classification Walkthrough

You can run this example yourself in these Colab notebooks

Train the model [Notebook 1 Link]
Roll out the model [Notebook 2 Link]
More tutorials in Climate Change AI Tutorials

You can also try Google's Dynamic World dataset

"Near realtime LULC" model at 10 meter resolution. Check https://dynamicworld.app/

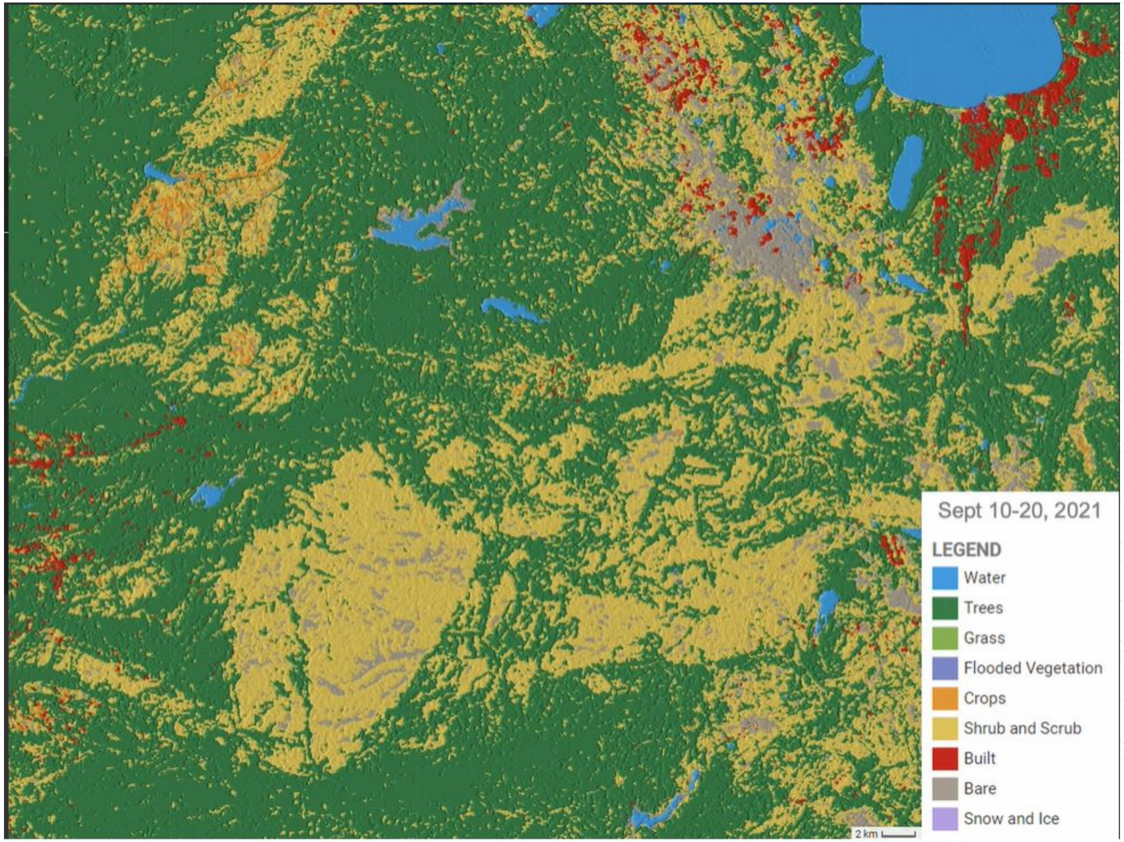


9 Land Use and Cover Types

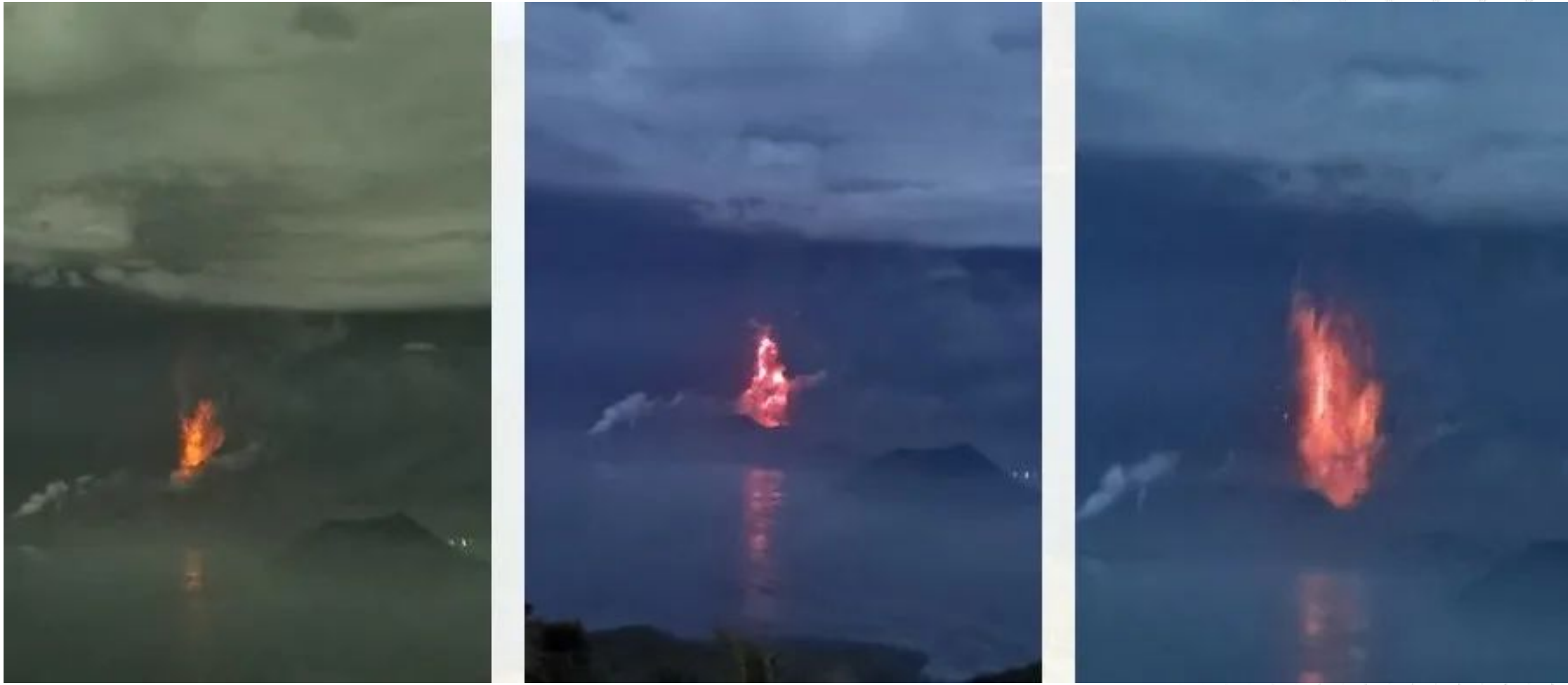
1. Water
2. Trees
3. Grass
4. Crops
5. Shrub and Scrub
6. Flooded Vegetation
7. Built-Up Area
8. Bare round
9. Snow and Ice

This is an area in California, USA averaged over Sept 10-20

Each pixel corresponds to a land cover/ land use



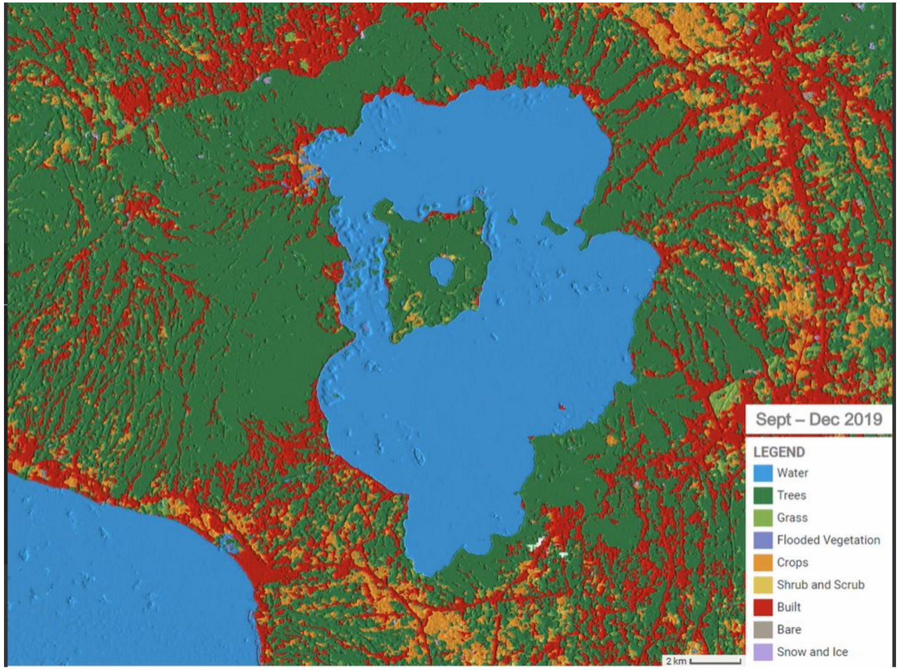
In Jan 2020, Taal Volcano in the Philippines erupted



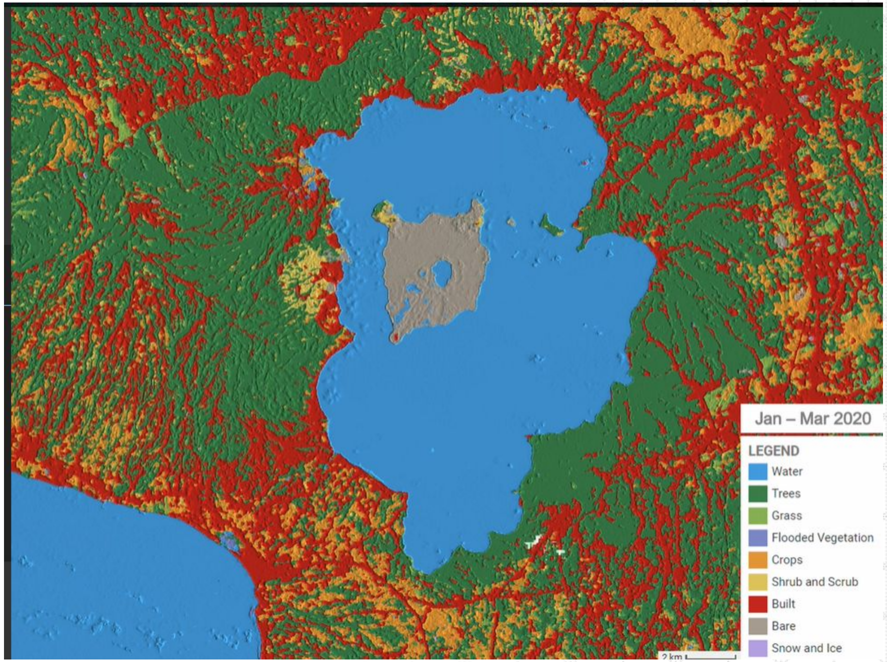
More than 24,000 people had to evacuate due to the ashfall

The Dynamic World dataset clearly shows the impact of the volcanic eruption

Before Eruption



After Eruption



Case Study: Pre-Assessment for Mangrove Restoration

How do we find the best shrimp ponds for replanting mangroves?

Mangrove forests are critical for slowing the effects of climate change

Fish and shrimp farming has destroyed **~38%** of mangroves globally

The **Climate Smart Shrimp** program allows farmers to produce more food with less land area, freeing space for replanting mangroves

Overview

Challenge

Up to date maps of fish farms are not easy to access. On top of this, the process of vetting if the fish farm qualifies for the requirements of the program is time consuming.

Solution

Use open geospatial data to identify the location of fish farms and identify which ones qualify for the program's ecological and operational requirements

Impact

Identified the top 40,000 hectares of fishponds that qualify for the Climate Smart Shrimp Program, shortening the selection period for the program's pilot sites

Where did we get the data?

Datasets used in ranking the sites

Points of Interest

- Road networks from OpenStreetMap
- Populated areas from Facebook and Center for International Earth Science Information Network (CIESIN)

Topographic Data

- Elevation from Copernicus GLO-30 DEM
- Slope from ALOS World 3D - 30m DEM

Current and Historical Presence of Mangroves

- Mangrove cover from 1999-2016 from Global Mangrove Watch

Case Study: Pre-assessment for Forest Carbon Projects

We ranked forests for suitability in terms of 3 frameworks

Different forest qualities require different frameworks to guide activities in the forest sector that reduces emissions

Afforestation/Reforestation

Find areas that have been deforested and has minimal population and no industrial use.

Revegetation

Find areas that have degraded forests that would benefit from assisted regeneration, has minimal population and no industrial use

REDD+

The criteria is optimized to find areas that have healthy forests at risk of degradation and deforestation

Overview

Challenge

Field-based carbon stock assessments are very **costly and time-consuming**. At this early stage of project planning, it is not cost-effective to commission on-the-ground feasibility studies without being highly selective

Solution

Use open data to map forest quality and restoration potential of unoccupied areas in the country

Impact

Identified the most suitable areas in the Philippines for protection which is 19% of pristine Philippine forests and suitable for restoration which is 10% of unoccupied land

Where did we get the data?

Datasets used

Global Forest Watch and Researcher Generated Datasets

- Carbon stock and potential sequestration to gauge carbon sink potential
- Deforestation risk to indicate level of protection and intervention needed

Demographic Data

- Population data to avoid displacement in favor of forestry activity
- History of insurgency to gauge safety

Land Use and Forest Cover

- Forest cover classification and canopy cover
- Land use classification

Recap

Data comes in many unconventional sources and novel open datasets

Location Information

Data based on geographic location, e.g. poverty estimates, points of interest, population count

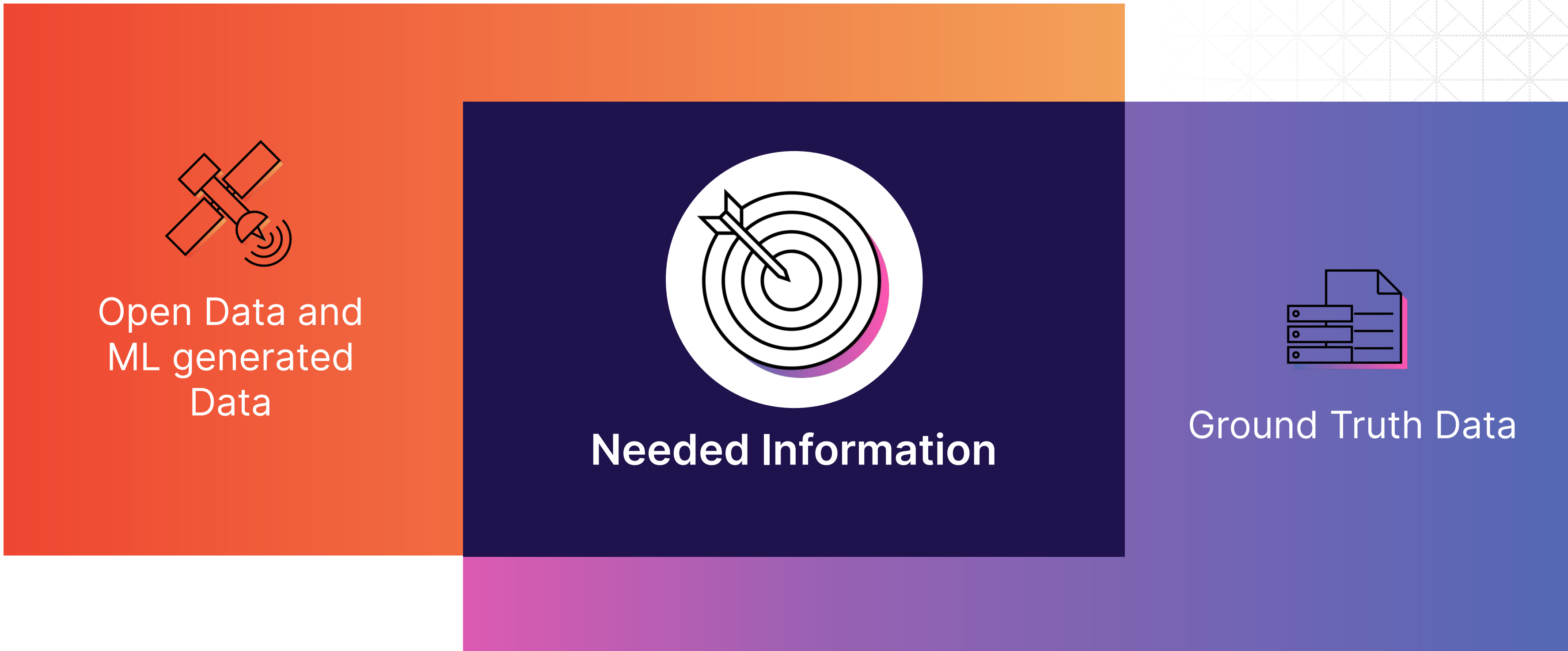
Satellite Imagery

Data collected by satellites, e.g. building footprints, land cover, nighttime lighting

Network Data

Data extracted from connected devices, e.g. signal coverage, internet speed, marketing data

We believe in the potential of open data to augment ground truth data and help fill out the gaps



Open Discussion

1. Thinking about your department's programmes, what is your **biggest goal** this year?
2. What **data do you need** to help you make decisions towards your big goal for the year?
3. Do you think **open data and ML models** (satellite imagery, crowdsourced data, etc) can support your department? If yes, how so?
4. What **resources** do you need to systematically use data in planning or implementing your programmes?

Instructor: Joshua Cortez

Day 2 Recap

1. Introduction to Classical Machine Learning
    a. Overview of ML
    b. Data Exploration
    c. Model Development
2. Introduction to Computer Vision
    a. Computer vision use cases
    b. Types of computer vision tasks
    c. Convolutional neural networks
    d. Land Cover Land Use Classification

Thank you!

Thinking Machines
Data Science

Data Stories
stories.thinkingmachin.es

Press
thinkingmachin.es/press-room

Follow us
/thinkdatasci
@thinkdatasci

Bangkok | Manila | Singapore

thinkingmachin.es

We’d love to hear your feedback!

Feedback Link:
https://forms.gle/NBjdsb9JEHV2C5kz9

Thinking Machines
Data Science

Data Stories
stories.thinkingmachin.es

Press
thinkingmachin.es/press-room

Follow us
/thinkdatasci
@thinkdatasci

Bangkok | Manila | Singapore

Appendix

References

- Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow
  - Practical application of ML models (examples, best practices)
  - Includes code snippets
- Elements of Statistical Learning
  - More “under the hood” details of models
  - Mathematical formulas and algorithm pseudocode
- Coursera Machine Learning
  - Building ML models from the ground up
  - Fundamentals + best practices